Volume

3

SBEM

Simplified Building Energy Model

Volume 3: C++ code

15 November 2010

BRE
JOSE ORTIZ
AZZA NABIL

SBEM SIMPLIFIED BUILDING ENERGY MODEL

Volume 3: C++ code

BRE
Sustainable Energy Group
Garston, Watford WD25 9XX

Jose Ortiz, ortizj@bre.co.uk
Phone +44 (0) 1923 664715 · Fax +44 (0) 1923 664097

Version History

Manual version	For SBEM version
15th November 2010	4.1.a

Key changes from the previous version in the order in which the Objects appear in this manual

1. This is the first version of this manual

Object	Keyword	Status

Table of Contents

Overview **5**
What is SBEM? 5
About This Manual 5
Version of the tool 5
Other documentation 5
SBEM Basics 5
SBEM Dictionary 6
BRUKL Basics & Dictionary 6
EPCgen Basics & Dictionary 6
BRIRL Basics & Dictionary 6
BERgen Basics & Dictionary 6
Installing and Running SBEM 6
Calculation Engine Structure 8
Uses for the program 9
Validation of SBEM 9
Input Hierarchy 10
Building Components **11**
CCHP-GENERATOR (*Renamed from CHP-GENERATOR*) 11
COMPLIANCE 16
CONSTRUCTION 34
DHW-GENERATOR 41
DOOR 76
DOOR-TM-BRIDGE 80
GENERAL 82
GLASS 122
HVAC-SYSTEM 125
PVS 190
REC-USER 193
SES 274
SMOKE-VENT (NO LONGER IN USE) 284
TSC 285
WALL 293
WALL-TM-BRIDGE 311
WIND-GENERATOR 313
WINDOW 316
WINDOW-TM-BRIDGE 326
ZONE 327
SBEM libraries: Matrix, BEEMLibDef, BEEMLibTab, ActlibraryTab **469**
Matrix 469
BEEMLibDef 511
BEEMLibTab 518
ActlibraryTab 555

Section

1

Overview

What is SBEM?

SBEM is a computer program that provides an analysis of a building's energy consumption. SBEM estimates the monthly energy use and carbon emissions of a building when given a description of the building's geometry, construction, and its building services equipment. SBEM was originally based on the Dutch methodology NEN 2916:1998, Energy Performance of Non-Residential Buildings. It has been developed in order to comply with the emerging CEN standards, where a full description of the methodology implemented in SBEM can be found. The purpose of the program is to aid in the analysis of energy usage in buildings through a simplified calculation that maintains simplicity and consistency for the user.

About This Manual

This manual includes c++ code for all the objects in the program SBEM (Simplified Building Energy Model). The code is organised by commands that are listed alphabetically. Keywords associated with commands are also listed, grouped according to function. Description of commands and keywords support the understanding of the c++ code for those not familiar with the intrinsic structure of the SBEM code. Additional comments and guidance will be added in future editions of this document. Code associated with general parts of the system, not relevant to the calculation of specific objects' performance, such as reporting, etc, has been excluded from this document.

Version of the tool

This manual refers to version **4.1.a** of SBEM.

Other documentation

This manual is the third volume in a set of three volumes. The other volumes in this set are:

SBEM Basics

This manual helps you learn the SBEM program by introducing you to the program's basic features.

SBEM Dictionary

This is a dictionary that defines all of the input commands (Objects) and Keywords in the program SBEM. It also gives examples of usage. It presents all of the output reports, together with explanations of many of the key report items. Additionally, it lists all the NCM activity database building types, activity types, and activity identification numbers.

The following manuals are also available:

BRUKL Basics & Dictionary

This manual defines all of the input commands (Objects) and Keywords in the compliance checking module for UK Building Regulations, BRUKL. It also presents a sample of the output compliance report, together with explanations of many of the key report items.

EPCgen Basics & Dictionary

This manual defines all of the input commands (Objects) and Keywords in the EPCgen module which generates the Energy Performance Certificate (EPC) for a building. It also presents a sample of the output reports (i.e., the Energy Performance Certificate and the Recommendations Report), together with explanations of many of the key report items.

BRIRL Basics & Dictionary

This manual defines all of the input commands (Objects) and Keywords in the compliance checking module for the Republic of Ireland Building Regulations, BRIRL. It also presents a sample of the output compliance report, together with explanations of many of the key report items.

BERgen Basics & Dictionary

This manual defines all of the input commands (Objects) and Keywords in the BERgen module which generates the Building Energy Rating (BER) for a building in the Republic of Ireland. It also presents a sample of the output reports (i.e., the BER Certificate and the Advisory Report), together with explanations of many of the key report items.

Installing and Running SBEM

The program is provided zipped in a file called **SBEM.zip**. There are no installation procedures apart from unzipping the file. The directory structure has to be kept as it is in order to allow the correct operation of the software.

Currently, SBEM runs under a DOS environment, using the Command Prompt of Windows. A Linux version of SBEM is also available. The program needs an input file (*.inp) to run the calculations, and it produces a coma-separated value file (*.sim) for results. It also displays some of the results in a (*_sbem.pdf) file.

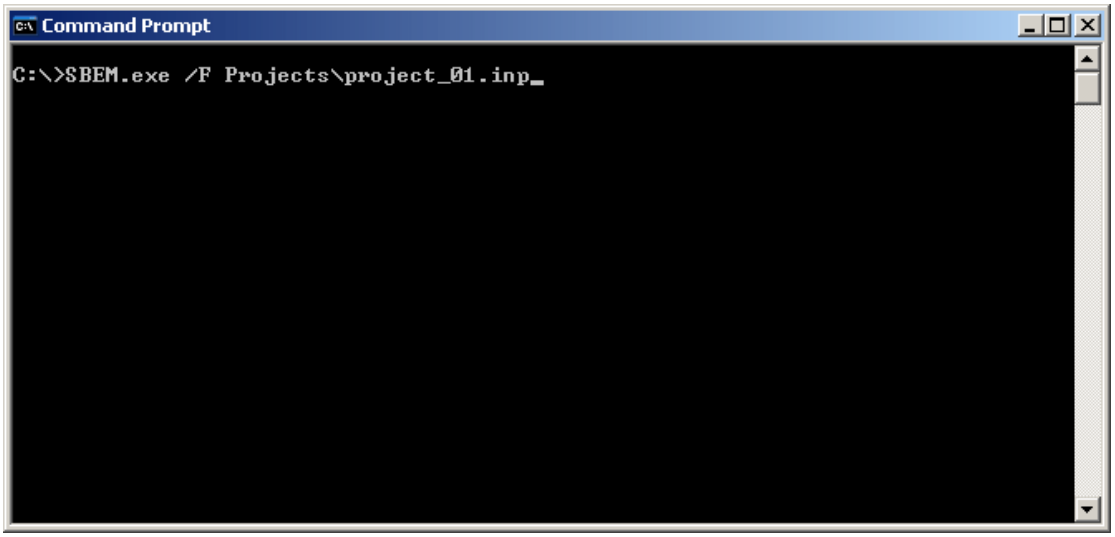


Figure 1: Example execution at the Command Prompt

To run the program:

1. Open a Command Prompt and go to the folder in which **SBEM.zip** was unzipped.

2. Ensure that the **SBEM.exe** file exists by using the DIR command and checking the list of files you get as a result. **SBEM.exe** should be on the list.

3. a) Execute the program using its name followed by **/F** and then the name of the input file you want to run the calculation on, e.g.:

   C:\>SBEM.exe /F Projects\project_01.inp eac=activation_code pps= username:proxy_password

   This instruction runs the **SBEM.exe** program with an input file called **project_01.inp** located in the folder **Projects**. The remaining arguments are optional and are used to indicate the Energy Assessor's EPCgen.net activation code (**eac=**) and, if applicable, the username and proxy server password (**pps=**), respectively.

   Note: It is no longer required to enter the word "test" (or any other text) after the name of the input file.

   b) An alternative way to run **sbem.exe** is by using the **SBEM.ini** file. This file must contain at least two lines (and optionally 3). The first line is the full path of the **SBEM.exe** application, the second line is the full path of the input "*.inp" file plus the name of the input file you want to run the calculation on, and the optional third line is the full path of the folder to store the SBEM.log file. Execute the program using its name followed by **/I** and then the name of the **SBEM.ini** file, e.g.:

   C:\>SBEM.exe /I SBEM.ini eac=activation_code pps= username:proxy_password

   Where **SBEM.ini** contains the following three lines:

C:\NCM\iSBEM_v4.1.a
C:\NCM\iSBEM_v4.1.a\Projects\project_01.inp
C:\NCM\iSBEM_v4.1.a

Calculation Engine Structure

Figure 2 shows a flowchart of the calculation engine. Basically, the program has one sub-program for the translation of the User Input and instructions from the Standard and User Libraries into computer recognisable form (BOC processor, BOC stands for Building Object Constructor), and the main calculation program which produces printable reports of the results of its calculations.

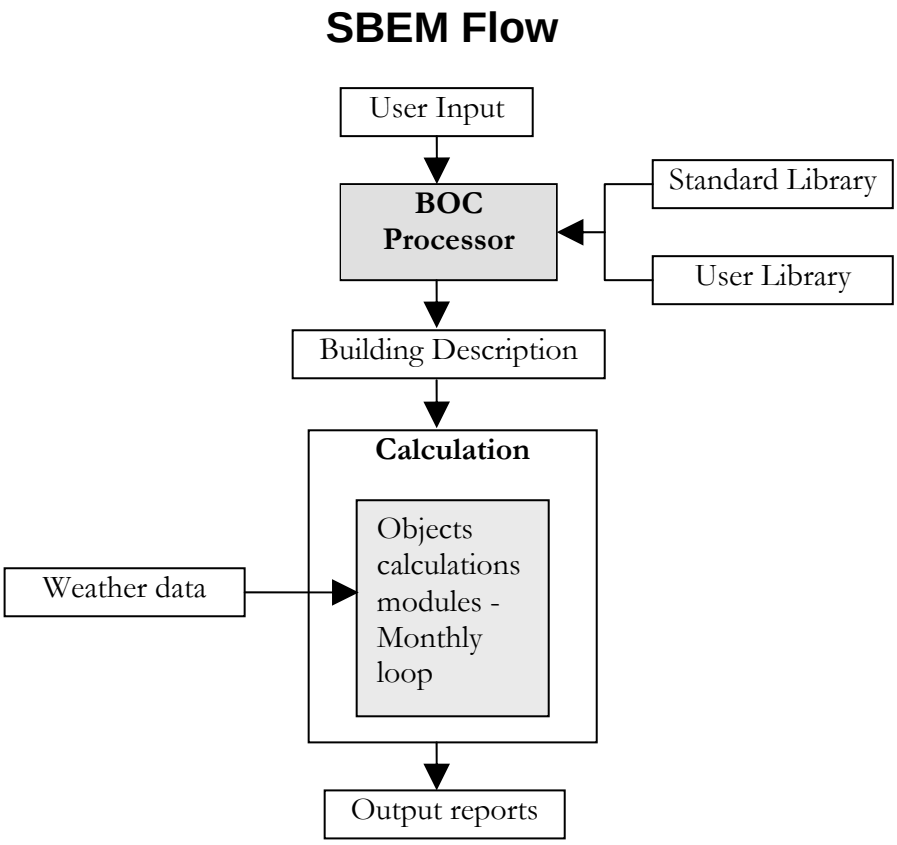


Figure 2: Calculation Engine Structure

The elements of the program shown in Figure 2 are as follows:

User Input
The User Input comprises of the description of the building's geometry, its building service systems, and the properties of the materials used in its construction.

Standard Library
The program comes with a library of input components.

User Library
The User Library contains descriptions of building elements such as the properties of materials used in the building, etc, that the user has used in a previous Asset Rating calculation but which are also relevant in the current calculation.

BOC Processor
The Building Object Constructor (BOC) processor reads the flexibly formatted input data that you supply and translates it into computer recognizable form. It also performs an initial check of the input data to detect input errors.

Calculation
The calculation modules are the kernel of the program. These modules combine the building's description data from the BOC Processor with the Weather Data to calculate the energy performance of the building split into different end-uses on a monthly basis. These modules are: Heating, cooling, lighting, ventilation, hot water, and auxiliary energy.

Weather Data
Weather Data is stored in a library file. CIBSE Weather data for 14 UK locations – Belfast, Birmingham, Cardiff, Edinburgh, Glasgow, Leeds, London, Manchester, Newcastle, Norwich, Nottingham, Plymouth, (Scotland,) Southampton, and Swindon - are included. It consists of monthly values including outside dry-bulb temperature and solar radiation based on the TRY (Test Reference Years) for the fourteen locations. It is not expected that the user would modify this data.

Uses for the program

This program was originally based on the Dutch methodology NEN 2916:1998 in which it is possible to find the methods used to determine the Energy Performance Coefficient (EPC) of a building or part of a building. The EPC standard characterises the energy performance of a building and its installations. However, no direct relationship exits between the determined energy performance and the actual energy consumption. The program has been developed in order to comply with the CEN Standard (Energy performance of buildings — Calculation of energy use for space heating and cooling [ISO 13790:2008]). This standard is one of a series of calculation methods for the design and evaluation of thermal and energy performance of buildings.

The program is intended for certification purposes. SBEM includes a Compliance check module for the Building Regulations of 2010 (Part L for England and Wales, Section 6 for Scotland, and (in the near future) Part F for Northern Ireland). It also produces a draft version of the Asset rating for EPBD compliance.

Validation of SBEM

SBEM validation followed the accredited scheme set up for the National Calculation Methodology 2005.

Input Hierarchy

The following list shows the recommended sequence in which the basic program commands (objects) should be entered.

GENERAL
COMPLIANCE
CONSTRUCTION
GLASS
CCHP-GENERATOR
DHW-GENERATOR
SES
PVS
WIND-GENERATOR
TSC
HVAC-SYSTEM
  ZONE
    WALL
      WALL-TM-BRIDGE
      DOOR or WINDOW
        DOOR-TM-BRIDGE or WINDOW-TM-BRIDGE
REC-USER

Section

2

Building Components

This section include the c++ code of all SBEM objects along with commands and keywords used to define the building object of study. Included here are the following objects (in alphabetical order):

1. CCHP-GENERATOR
2. COMPLIANCE
3. CONSTRUCTION
4. DHW-GENERATOR
5. DOOR
6. DOOR-TM-BRIDGE
7. GENERAL
8. GLASS
9. HVAC-SYSTEM
10. PVS
11. REC-USER
12. SES
13. *SMOKE VENT* (NO LONGER IN USE)
14. TSC
15. WALL
16. WALL-TM-BRIDGE
17. WIND-GENERATOR
18. WINDOW
19. WINDOW-TM-BRIDGE
20. ZONE

CCHP-GENERATOR (*Renamed from CHP-GENERATOR*)

The CCHP-GENERATOR command defines the performance parameters of a Combined Cooling, Heating, and Power (CCHP) generator for provision of hot water, space cooling, space heating, and electrical energy.

Example input:

```
"CCHP-GENERATOR 1" = CCHP-GENERATOR
   TRIGENERATION       = NO
```

```
HEAT-SEER           = 0.55
BUILD-CHP-SHEAT     = 0.5
BUILD-CHP-DHEAT     = 0.5
HEAT-POWER-RATIO    = 1.0
FUEL-TYPE           = Natural Gas
..
```

TRIGENERATION
Specifies whether this is a trigeneration system (i.e., it provides cooling, heating, and power) or not (i.e., it provides heating and power). The value can be YES or NO. The default is NO.

HEAT-SEER (*RENAMED FROM HEAT-SSEER*)
This is the seasonal thermal efficiency of the CCHP generator, defined as the total annual useful heat supplied by the generator divided by the total annual fuel energy input to the generator

COOL-SEER
This is the chiller seasonal energy efficiency ratio of the CCHP generator, defined as the cooling demand divided by the cooling energy for the generator. This keyword is included in the input if the CCHP-GENERATOR object's TRIGENERATION keyword is set to YES, i.e., the generator also provides cooling.

CHPQA_QI
This is an indicator of the energy efficiency and environmental performance of a CCHP scheme relative to the generation of the same amounts of heat and power by separate alternative means.

BUILD-CHP-SHEAT
This is a ratio representing the proportion, in decimal form, of the HVAC space heating energy demand which is provided by the CCHP generator for the HVAC-SYSTEM connected to this CCHP-GENERATOR. The default value is 0.

BUILD-CHP-DHEAT
This is a ratio representing the proportion, in decimal form, of the hot water heating energy demand which is provided by the CCHP generator for the DHW-GENERATOR connected to this CCHP-GENERATOR. The default value is 0.

BUILD-CHP-SCOOL
This is a ratio representing the proportion, in decimal form, of the HVAC space cooling energy demand which is provided by the CCHP generator for the HVAC-SYSTEM connected to this CCHP-GENERATOR. The default value is 0. This keyword is included in the input if the CCHP-GENERATOR object's TRIGENERATION keyword is set to YES, i.e., the generator also provides cooling.

HEAT-POWER-RATIO
Specifies the heat to power ratio of the CCHP generator, defined as kWh of heat produced per kWh of electricity generated, in decimal form. The default value is 1.5.

FUEL-TYPE

Specifies the fuel type for the CCHP generator, as one of the values listed below. The default is 'Natural Gas'.

Natural Gas
LPG
Biogas
Oil
Coal
Anthracite
Smokeless Fuel (inc Coke)
Dual Fuel Appliances (Mineral + Wood)
Biomass

c++ code

```
//: chp.cpp

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void CHP GENERATOR::TEST() {
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test ZONE" << endl;
//  cout << "Fin de Test ZONE" << endl;
  BEEMObject::TEST();
}


void CHP GENERATOR::Init(BOC& P) {
  if(func report) cout << endl << "  " << "CHP GENERATOR::Init";
//CHP may serve more than 1 HVAC system, in this function the CHP
heating and DHW energy are set to zero
//heating and DHW energy are later added from DHW-generator and ZONE
functions
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << 0.0 << ", ";
   res << 0.0 << " }";
   P.ModKey(Id,"Q-DEM-HEAT-M",res.str());
   P.ModKey(Id,"Q-DEM-HEATCOOL-M",res.str());
   P.ModKey(Id,"Q-DEM-DHW-M",res.str());
   P.ModKey(Id,"KWH-HEAT-M",res.str());
   P.ModKey(Id,"KWH-HEATCOOL-M",res.str());
   P.ModKey(Id,"KWH-DHW-M",res.str());
  }

  {
   ostringstream res; res << Id;
   P.ModKey(gId,"CHP-POS",res.str());
  }
}

void CHP GENERATOR::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func report) cout << endl << "  " << "CHP GENERATOR::Calc";
  if(calc report) cout << endl << P.vINP[Id].Name;
 //calculate kgCO2 produced by CHP fuel use:
 fKG CO2 PROD(D, T, P);
 //calculate grid displaced electricity produced by CHP:
 fDISP(D, T, P);
 //calculate kg CO2 displaced associated with electricity produced by
CHP:
 fKG_CO2_DISP(D, T, P);
```

```
}


void  CHP GENERATOR::fKG CO2 PROD(PROJECTDefaults&  D,  PROJECTTables&
T, BOC& P) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"CHP GENERATOR::fKG CO2 PROD";
//calculates kgCO2 produced by CHP fuel use:

 deque<double> mKWH HEAT M=P.Vlnum(Id,"KWH-HEAT-M");
 deque<double> mKWH_DHW_M=P.Vlnum(Id,"KWH-DHW-M");
 deque<double> mKWH HEATCOOL M=P.Vlnum(Id,"KWH-HEATCOOL-M");
 double mKG CO2 M[12]={ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0 };
 double mKG CO2=0.0;

 if((P.Vcodid(cId,"BR-TYPE")==5)   ||   (P.Vcodid(cId,"EPC-TYPE")==6)){
//Republic of Ireland only
     for(int i=0;i<12;i++) {

mKG CO2 M[i]=(mKWH HEAT M[i]+mKWH HEATCOOL M[i]+mKWH DHW M[i])*T.Valu
e("tKGCO2 IR",1,P.Vcodid(Id,"FUEL-TYPE"));
       mKG CO2+=mKG CO2 M[i];
     }
 }
 else if((P.Vcodid(cId,"BR-TYPE")==6)){  //Jersey
     for(int i=0;i<12;i++) {

mKG CO2 M[i]=(mKWH HEAT M[i]+mKWH HEATCOOL M[i]+mKWH DHW M[i])*T.Valu
e("tKGCO2 JER",1,P.Vcodid(Id,"FUEL-TYPE"));
       mKG CO2+=mKG CO2 M[i];
     }
 }
 else{  // all UK
     for(int i=0;i<12;i++) {

mKG CO2 M[i]=(mKWH HEAT M[i]+mKWH HEATCOOL M[i]+mKWH DHW M[i])*T.Valu
e("tKGCO2",1,P.Vcodid(Id,"FUEL-TYPE"));
       mKG_CO2+=mKG_CO2_M[i];
     }
 }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKG CO2 M[i] << ", ";
  res << mKG CO2 M[11] << " }";
  P.ModKey(Id,"KG-CO2-M",res.str());
 }
 {
  ostringstream res; res << mKG_CO2;
  P.ModKey(Id,"KG-CO2",res.str());
 }
}



void CHP GENERATOR::fDISP(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func_report) cout << endl << "  " << "CHP_GENERATOR::fDISP";
//calculates grid displaced electricity produced by CHP

  double mKWH DISP M[12]={ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };
  double mKWH DISP=0.0;
  deque<double> mKWH_DHW_M=P.Vlnum(Id,"KWH-DHW-M");
  deque<double> mKWH HEAT M=P.Vlnum(Id,"KWH-HEAT-M");
  deque<double> mKWH HEATCOOL M=P.Vlnum(Id,"KWH-HEATCOOL-M");
  //elec power out/heat produced:
```

```
  double mHEAT POWER RATIO=P.Vnum(Id,"HEAT-POWER-RATIO");

 for(int i=0;i<12;i++) {

mKWH_DISP_M[i]=(mKWH_DHW_M[i]+mKWH_HEAT_M[i]+mKWH_HEATCOOL_M[i])*P.Vn
um(Id,"HEAT-SEER")/mHEAT POWER RATIO;
mKWH DISP+=mKWH DISP M[i];
 }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH DISP M[i] << ", ";
  res << mKWH_DISP_M[11] << " }";
  P.ModKey(Id,"KWH-DISP-M",res.str());
 }
 {
  ostringstream res; res << mKWH DISP;
  P.ModKey(Id,"KWH-DISP",res.str());
 }

}

void  CHP GENERATOR::fKG CO2 DISP(PROJECTDefaults&  D,  PROJECTTables&
T, BOC& P) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"CHP GENERATOR::fKG CO2 DISP";

 double mKG CO2 DISP M[12]={ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };
 deque<double> mKWH DISP M=P.Vlnum(Id,"KWH-DISP-M");
 double mKG CO2 DISP=0.0;

 if((P.Vcodid(cId,"BR-TYPE")==5)   ||   (P.Vcodid(cId,"EPC-TYPE")==6)){
//Republic of Ireland only
     for(int i=0;i<12;i++) {
       mKG CO2 DISP M[i]=mKWH DISP M[i]*T.Value("tKGCO2 IR",1,11);
       mKG CO2 DISP+=mKG CO2 DISP M[i];
     }
 }
 else if((P.Vcodid(cId,"BR-TYPE")==6)){  //Jersey
     for(int i=0;i<12;i++) {
       mKG CO2 DISP M[i]=mKWH DISP M[i]*T.Value("tKGCO2 JER",1,11);
       mKG CO2 DISP+=mKG CO2 DISP M[i];
     }
 }
 else{ // all UK
     for(int i=0;i<12;i++) {
       mKG CO2 DISP M[i]=mKWH DISP M[i]*T.Value("tKGCO2",1,11);
       mKG CO2 DISP+=mKG CO2 DISP M[i];
     }
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKG CO2 DISP M[i] << ", ";
  res << mKG_CO2_DISP_M[11] << " }";
  P.ModKey(Id,"KG-CO2-DISP-M",res.str());
 }
 {
  ostringstream res; res << mKG CO2 DISP;
  P.ModKey(Id,"KG-CO2-DISP",res.str());
 }
}

void CHP GENERATOR::Recommendation(PROJECTDefaults& D, PROJECTTables&
T, BOC& P) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"CHP_GENERATOR::Recommendation";
```

```
}
```

COMPLIANCE

The second object in the input file must be the COMPLIANCE object. This specifies compliance data.

Example input:

```
"SBEM" = COMPLIANCE
   EPC-TYPE            = EPC England
   BR-TYPE             = England and Wales Building Regulations
Part L 2010
   ENG-HERITAGE        = NO
   BR-STAGE            = As built
   ..
```

Modified

BR-TYPE

Specifies the type of Building Regulations Compliance check for the analysis, as one of the values listed below. The default is 'None'.

None
England and Wales Building Regulations Part L 2010
Scottish Building Regulations 2010
Northern Ireland Building Regulations Part F 2010 (TEMPORARILY NOT IN USE)
Republic of Ireland: Building Regulations Part L (TEMPORARILY NOT IN USE)
Building Bye-laws Jersey Part 11

Modified

EPC-TYPE

Specifies the type of Energy Performance Certificate for the analysis, as one of the values listed below. The default is 'None'.

None
EPC England
EPC Wales
EPC Scotland
EPC Northern Ireland (TEMPORARILY NOT IN USE)
Republic of Ireland: Building Energy Rating (TEMPORARILY NOT IN USE)

BR-STAGE

Specifies the stage of the building at which the analysis is carried out, as one of the values listed below. The default is 'As built'. This keyword does not appear in the input file if the COMPLIANCE object's EPC-TYPE keyword has the value 'Republic of Ireland: Building Energy Rating'.

As designed

As built

BER-STAGE
Specifies the stage of the building at which the Republic of Ireland's BER analysis is carried out to generate a BER certificate, as one of the values listed below. The default is 'New Building – Provisional'. This keyword appears in the input if the COMPLIANCE object's EPC-TYPE keyword has the value 'Republic of Ireland: Building Energy Rating'.

Existing Building - Final
New Building - Final
New Building - Provisional

ENG-HERITAGE
This keyword is included in the input of the COMPLIANCE object if the energy calculation is for EPC generation purposes in England, Wales, or Northern Ireland. It specifies whether the building has been identified as being: one of special architectural or historical interest, in a conservation area, in a designated area of special character or appearance, or of traditional construction. The value can be YES or NO. The default is NO.

SHELL-AND-CORE
This keyword is included in the input of the COMPLIANCE object if the energy calculation is for compliance checking purposes in England, Wales, Northern Ireland, or Jersey. It specifies whether the building is a shell and core development. The value can be YES or NO. The default is NO.

MODULAR-PORTABLE
Specifies whether the building is modular and portable. The value can be YES or NO. The default is NO.

MODULAR-YEARS
This keyword is included in the input if the COMPLIANCE object's MODULAR-PORTABLE keyword is set to YES. It specifies the planned time of use of the modular and portable building, as one of the values listed below. The default is 'More than 2 years'.

Less than 2 years
More than 2 years

MODULAR-CORRECTION
This keyword is included in the input if the COMPLIANCE object's MODULAR-YEARS keyword is set to 'More than 2 years'. It specifies whether more than 70% of the modules making up the external envelope of the modular and portable building have been manufactured before the 2010 Building Regulations come into force. The value can be YES or NO. The default is NO.

MODULAR-ADL2-T3
This keyword is included in the input if the COMPLIANCE object's MODULAR-CORRECTION keyword is set to YES. It specifies the date of manufacture of 70% of the modules making up the external envelope of the modular and portable building, as one of the values listed below. The default is 'Post October 2010'.

Post October 2010
April 2006 - September 2010
April 2002 - March 2006
Pre April 2002

EPC-LANGUAGE
Specifies the language to use for producing the Energy Performance Certificate, as one of the values listed below. The default is 'ENGLISH'.

ENGLISH
WELSH

BR-S6-BTYPE
Specifies the type of building according to the Scottish Section 6 document, as one of the values listed below, when the COMPLIANCE object's BR-TYPE keyword is set to 'Scottish Building Regulations 2010'. The default is 'Other buildings'.

Other buildings
Shell buildings
Extensions to the insulation envelope

***BR-S6-ACCRED-CONST* (NO LONGER IN USE)**
Specifies whether the building is designed and built following the guidance in Accredited Construction Details (Scotland) when the keyword BR-TYPE is set to 'Scottish Building Regulations'. The value can be YES or NO. The default is NO.

BR-S6-NV-CREDIT
Specifies whether the building is fully naturally ventilated and separate calculations have confirmed that the temperature during occupied periods is always below 28ºC. The value can be YES or NO. The default is NO. This keyword is included in the input if the COMPLIANCE object's keyword BR-TYPE is set to 'Scottish Building Regulations'.

S-EPC-PER
Specifies the potential Asset (EPC) Rating of the building if the recommended energy performance improvements were implemented. This keyword is included in the input if the COMPLIANCE object's EPC-TYPE keyword is set to 'EPC Scotland', and a value for this keyword by the user.

***OVERHEAT-RISK* (NO LONGER IN USE)**
This keyword specifies whether one or more zones in the building have a risk of overheating. The value can be YES or NO. The default is NO.

AIR-CON-INSTALLED
Specifies if the building has an air-conditioning system, as one of the values listed below. The default is 'Unknown'. This keyword appears in the input if the COMPLIANCE object's EPC-TYPE keyword has the value 'EPC England', 'EPC Wales', or 'EPC Northern Ireland'.

Yes
No

Unknown

AIR-CON-KW
Specifies the total effective rated output, in kW, of the air-conditioning system in the building, as one of the values listed below. The default is 'Greater than 250kW'. This keyword appears in the input if the COMPLIANCE object's AIR-CON-INSTALLED keyword has the value 'Yes'.

Below 12kW
Between 12kW and 250kW
Greater than 250kW
EXACT

AIR-CON-KW-EXACT
Specifies the total effective rated output, in kW, of the air-conditioning system in the building when the exact value is known, i.e., it is included in the input if the COMPLIANCE object's AIR-CON-KW keyword has the value 'EXACT'. The default value is 0.

AIR-CON-INSPECTION
Specifies the total effective rated output, in kW, of the air-conditioning system in the building, as one of the values listed below. The default is 'Don't know'.

Yes - inspection completed
Yes - inspection commissioned
No inspection completed or commissioned
Not relevant
Don't know

EA-EPCGEN-KEY **(NO LONGER IN USE)**
This keyword specifies the security key provided by the Accreditation Schemes to accredited Energy Assessors to enable the generation of UK EPCs and Recommendations Reports which do not include the "Only For Illustration" watermark, and the generation of the XML message. If this keyword and a valid security key are not included in the input file, the output files produced will include the watermark, and the XML message will not be generated. This is applicable only if the COMPLIANCE object keyword EPC-TYPE has the value 'EPC England', 'EPC Wales', or 'EPC Northern Ireland'.

EA-BERGEN-KEY **(NO LONGER IN USE)**
This keyword specifies the security key provided by the Accreditation Schemes to accredited Energy Assessors to enable the generation of Republic of Ireland BER Certificates and Advisory Reports which do not include the "Only For Illustration" watermark, and the generation of the XML message. If this keyword and a valid security key are not included in the input file, the output files produced will include the watermark, and the XML message will not be generated. This is applicable only if the COMPLIANCE object keyword EPC-TYPE has the value 'Republic of Ireland: Building Energy Rating'.

c++ code

```
//: compliance.cpp

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;
```

```
void COMPLIANCE::Init(PROJECTTables& T, BOC &P) {
  if(func report) cout << endl << "  " << "COMPLIANCE::Init";

  // 3=Scottish Building Regulations 2010 , 2=Shell and fit out
buildings
  if( (P.Vcodid(Id,"BR-TYPE")==3) && (P.Vcodid(Id,"BR-S6-BTYPE")==2)
) {
   {
    ostringstream res; res << "YES";        // set the shell and core
flag to yes
    P.ModKey(Id,"SHELL-AND-CORE",res.str());
   }
   {   ostringstream res; res << "As designed";
    P.ModKey(Id,"BR-STAGE",res.str());
   }
  } // end if

}

void COMPLIANCE::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC
&P, BOC &N) {
  if(func report) cout << endl << "  " << "COMPLIANCE::Notional";
  BObject Temp;
  Temp.Ntype=P.vINP[Id].Ntype;
  Temp.Type=P.vINP[Id].Type;
  Temp.Name=P.vINP[Id].Name;
  KeywordValue KV;
 {
  ostringstream res; res << P.Vcod(Id,"BR-TYPE");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("BR-TYPE",KV));
 }
 {
  ostringstream res; res << P.Vcod(Id,"EPC-TYPE");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("EPC-TYPE",KV));
 }
 {
  ostringstream res; res << P.Vcod(Id,"SHELL-AND-CORE");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make_pair("SHELL-AND-CORE",KV));
 }
 {
  ostringstream res; res << P.Vcod(Id,"BR-STAGE");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("BR-STAGE",KV));
 }

  N.vINP.push back(Temp);
}

void COMPLIANCE::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC& P,
char* file_INP) {
  if(func report) cout << endl << "   " << "COMPLIANCE::Calc   " <<
file INP;
  if(calc_report) cout << endl << P.vINP[Id].Name;
  fU VALUES INF(D, T, P);
  fSOLAR GAIN CHECK(D, T, P);

  // Checks if any thermal bridge junction in any zone has been
defined with accredited construction details
  fACCRED_CONST_CHECK(D, T, P);

}


void COMPLIANCE::fIF RB CHECK(PROJECTDefaults& D, PROJECTTables& T,
BOC& P) {
```

```
  if(func report) cout << endl << "  " << "COMPLIANCE::fIF RB CHECK";

}

void COMPLIANCE::fCalcInf(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func report) cout << endl << "  " << "COMPLIANCE::fCalcInf";
 double mQ50_INF=0.0;
 double mZONE AREA=0.0;
 double mZONE AREA TOTAL=0.0;
 double INF_AREA=0.0;

 for (int i=0;i<P.vINP.size();i++)
   if (P.vINP[i].Type=="ZONE") {
     mZONE AREA=P.Vnum(i,"AREA-SUR-EXT")*P.Vnum(i,"MULTIPLIER");
     mQ50 INF=P.Vnum(i,"Q50-INF");
     INF_AREA+=mQ50_INF*mZONE_AREA;
     mZONE AREA TOTAL+=mZONE AREA;
   }
// area weighted infiltration:
 if (mZONE AREA TOTAL>0)
    INF AREA/=mZONE AREA TOTAL;

 {
  ostringstream res; res << INF AREA;
  P.ModKey(Id,"Q50-INF-AVE",res.str());
 }

}

void COMPLIANCE::fU VALUES INF(PROJECTDefaults& D, PROJECTTables& T,
BOC &P) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"COMPLIANCE::fU VALUES INF";

//U-values for each construction type:
//          - calculating area weighted ave and max values,total and
for each orientation:

 double mROOF U AVE[10]={0,0,0,0,0,0,0,0,0,0};
 double mWALL U AVE[10]={0,0,0,0,0,0,0,0,0,0};
 double mFLOOR_U_AVE[10]={0,0,0,0,0,0,0,0,0,0};
 double mWINDOW U AVE[10]={0,0,0,0,0,0,0,0,0,0};
 double mP DOOR U AVE[10]={0,0,0,0,0,0,0,0,0,0};
 double mV_DOOR_U_AVE[10]={0,0,0,0,0,0,0,0,0,0};
 double mH DOOR U AVE[10]={0,0,0,0,0,0,0,0,0,0};
 double mS VENT U AVE[10]={0,0,0,0,0,0,0,0,0,0};
 double mWINROOF_U_AVE[10]={0,0,0,0,0,0,0,0,0,0};
 double mROOFLIGHT U AVE[10]={0,0,0,0,0,0,0,0,0,0};
 double mBUILD U AVE[10]={0,0,0,0,0,0,0,0,0,0};

 double mROOF U MAX[10]={0,0,0,0,0,0,0,0,0,0};
 double mWALL U MAX[10]={0,0,0,0,0,0,0,0,0,0};
 double mFLOOR_U_MAX[10]={0,0,0,0,0,0,0,0,0,0};
 double mWINDOW U MAX[10]={0,0,0,0,0,0,0,0,0,0};
 double mP DOOR U MAX[10]={0,0,0,0,0,0,0,0,0,0};
 double mV_DOOR_U_MAX[10]={0,0,0,0,0,0,0,0,0,0};
 double mH DOOR U MAX[10]={0,0,0,0,0,0,0,0,0,0};
 double mS VENT U MAX[10]={0,0,0,0,0,0,0,0,0,0};
 double mWINROOF_U_MAX[10]={0,0,0,0,0,0,0,0,0,0};
 double mROOFLIGHT U MAX[10]={0,0,0,0,0,0,0,0,0,0};

 double mROOF_U_MIN[10]={10,10,10,10,10,10,10,10,10,10};
 double mWALL U MIN[10]={10,10,10,10,10,10,10,10,10,10};
 double mFLOOR U MIN[10]={10,10,10,10,10,10,10,10,10,10};
 double mP_DOOR_U_MIN[10]={10,10,10,10,10,10,10,10,10,10};
 double mV DOOR U MIN[10]={10,10,10,10,10,10,10,10,10,10};
 double mH DOOR U MIN[10]={10,10,10,10,10,10,10,10,10,10};
 double mWINROOF_U_MIN[10]={10,10,10,10,10,10,10,10,10,10};
```

```
 double mROOF WIN U AVE[10]={0,0,0,0,0,0,0,0,0,0};
 double mWALL WIN U AVE[10]={0,0,0,0,0,0,0,0,0,0};
 double mROOF WIN AREA[10]={0,0,0,0,0,0,0,0,0,0};
 double mWALL_WIN_AREA[10]={0,0,0,0,0,0,0,0,0,0};
 double mWALL WIN U MAX[10]={0,0,0,0,0,0,0,0,0,0};
 double mROOF WIN U MAX[10]={0,0,0,0,0,0,0,0,0,0};

 double mFLOOR AREA[10]={0,0,0,0,0,0,0,0,0,0};
 double mWINDOW AREA[10]={0,0,0,0,0,0,0,0,0,0};
 double mP_DOOR_AREA[10]={0,0,0,0,0,0,0,0,0,0};
 double mV DOOR AREA[10]={0,0,0,0,0,0,0,0,0,0};
 double mH DOOR AREA[10]={0,0,0,0,0,0,0,0,0,0};
 double mS_VENT_AREA[10]={0,0,0,0,0,0,0,0,0,0};
 double mROOFLIGHT AREA[10]={0,0,0,0,0,0,0,0,0,0};
 double mWINROOF AREA[10]={0,0,0,0,0,0,0,0,0,0};
 double mROOF_AREA[10]={0,0,0,0,0,0,0,0,0,0};
 double mWALL AREA[10]={0,0,0,0,0,0,0,0,0,0};
 double mTOTAL AREA[10]={0,0,0,0,0,0,0,0,0,0};

 double mAREA=0.0;
 double mWALL WIN U VALUE=0.0;
 double mROOF WIN U VALUE=0.0;
 double mU VALUE=0.0;
 int mORIENTATION;
 double INF AREA=0.0;
 int hvac id=0;
 int zone id=0;
 int wall id=0;
 int mWALL MAX=P.Vint(Id,"WALL-MAX");
 int mFLOOR MAX=P.Vint(Id,"FLOOR-MAX");
 int mROOF MAX=P.Vint(Id,"ROOF-MAX");
 int mP DOOR MAX=P.Vint(Id,"P-DOOR-MAX");
 int mV DOOR MAX=P.Vint(Id,"V-DOOR-MAX");
 int mH DOOR MAX=P.Vint(Id,"H-DOOR-MAX");
 int mS VENT MAX=P.Vint(Id,"S-VENT-MAX");
 int mWINROOF MAX=P.Vint(Id,"WINROOF-MAX");

 int mWALL_MIN=P.Vint(Id,"WALL-MIN");
 int mFLOOR MIN=P.Vint(Id,"FLOOR-MIN");
 int mROOF MIN=P.Vint(Id,"ROOF-MIN");
 int mP_DOOR_MIN=P.Vint(Id,"P-DOOR-MIN");
 int mV DOOR MIN=P.Vint(Id,"V-DOOR-MIN");
 int mH DOOR MIN=P.Vint(Id,"H-DOOR-MIN");
 int mWINROOF_MIN=P.Vint(Id,"WINROOF-MIN");

 int mWALL MAX VAL[10]={0,0,0,0,0,0,0,0,0,0};
 int mWALL_MAX_ID[10]={0,0,0,0,0,0,0,0,0,0};
 int j;


 for (int i=0;i<P.vINP.size();i++) {
// Next 2 lines added in order to exclude envelopes in zones with no
hvac from u-value check - AN 01/02/07
   if (P.vINP[i].Type=="HVAC-SYSTEM") {
     hvac id=i;
   }
//   if (P.vINP[i].Type=="WALL") {
   else           if           ((P.vINP[i].Type=="WALL")           &&
(P.Vcodid(hvac id,"TYPE")!=1))  {   //ignore  walls  in  zones  with  no
HVAC - AN 01/02/07
     wall id=i;
     mAREA=P.Vnum(i,"NET-AREA")*P.Vnum(i,"MULTIPLIER");
     mTOTAL AREA[0]+=mAREA;
     mU VALUE=P.Vnum(i,"U-VALUE");
     mORIENTATION=P.Vcodid(i,"ORIENTATION");
     if ( (P.Vcodid(i,"TYPE")!=4) && (P.Vcodid(i,"TYPE")!=6) ) {
       switch (P.Vcodid(i,"TYPE-ENV")) {
         //wall:
```

```
          case 1: {
            mWALL AREA[0]+=mAREA;
            mWALL WIN U VALUE=P.Vnum(i,"U-VALUE-WALL/WIN");
            mWALL AREA[mORIENTATION]+=mAREA;
            if (mU_VALUE>mWALL_U_MAX[mORIENTATION]) {
              for(j=0;j<10;j++)
                if(mU VALUE<=mWALL U MAX[j]) break;
              if (j==10)
                 mWALL MAX=i;
              mWALL U MAX[mORIENTATION]=mU VALUE;
            }

            if (mU VALUE<mWALL U MIN[mORIENTATION]) {
              for(j=0;j<10;j++)
                if(mU VALUE>=mWALL U MIN[j]) break;
              if (j==10)
                 mWALL_MIN=i;
              mWALL U MIN[mORIENTATION]=mU VALUE;
            }

            mWALL U AVE[mORIENTATION]+=(mU VALUE*mAREA);
            mBUILD U AVE[mORIENTATION]+=(mU VALUE*mAREA);
            if        (mWALL WIN U VALUE>mWALL WIN U MAX[mORIENTATION])
mWALL WIN U MAX[mORIENTATION]=mWALL WIN U VALUE;
          }
          break;
          //roof:
          case 2: {
            mROOF AREA[0]+=mAREA;
            mROOF WIN U VALUE=P.Vnum(i,"U-VALUE-WALL/WIN");
            mROOF AREA[mORIENTATION]+=mAREA;
            if (mU VALUE>mROOF U MAX[mORIENTATION]) {
              mROOF U MAX[mORIENTATION]=mU VALUE;
              mROOF MAX=i;
            }

            if (mU VALUE<mROOF U MIN[mORIENTATION]) {
              mROOF U MIN[mORIENTATION]=mU VALUE;
              mROOF_MIN=i;
            }

            mROOF_U_AVE[mORIENTATION]+=(mU_VALUE*mAREA);
            mBUILD U AVE[mORIENTATION]+=(mU VALUE*mAREA);
            if        (mROOF WIN U VALUE>mROOF WIN U MAX[mORIENTATION])
mROOF_WIN_U_MAX[mORIENTATION]=mROOF_WIN_U_VALUE;
          }
          break;
          //floor/ceiling:
          case 3: {
            // Check floors adjacent to anything other than CAS and
same space - AN - 15/01/2010
            if ( (P.Vcodid(i,"TYPE")!=4) && (P.Vcodid(i,"TYPE")!=6) )
{             mFLOOR AREA[0]+=mAREA;
              mFLOOR_AREA[mORIENTATION]+=mAREA;
              if (mU VALUE>mFLOOR U MAX[mORIENTATION]) {
                mFLOOR U MAX[mORIENTATION]=mU VALUE;
                mFLOOR_MAX=i;
              }

              if (mU_VALUE<mFLOOR_U_MIN[mORIENTATION]) {
                mFLOOR U MIN[mORIENTATION]=mU VALUE;
                mFLOOR MIN=i;
              }

              mFLOOR U AVE[mORIENTATION]+=(mU VALUE*mAREA);
              mBUILD_U_AVE[mORIENTATION]+=(mU_VALUE*mAREA);
            }
          }
          break;
```

```
          default: {
          }
        }
      }
    }
    else          if          ((P.vINP[i].Type=="WINDOW")          &&
(P.Vcodid(hvac id,"TYPE")!=1)) {   //ignore windows in zones with no
HVAC - AN 01/02/07
     if (P.Vcodid(i,"DISP-WIND")==2 && P.Vcodid(wall id,"TYPE")==1) {
//ignore display windows and non exterior walls
       mAREA=P.Vnum(i,"AREA")*P.Vnum(i,"MULTIPLIER");
       mU VALUE=P.Vnum(i,"U-VALUE-VERT");
       switch (P.Vcodid(wall id,"TYPE-ENV")) {
         case 1: {
           mWINDOW AREA[0]+=mAREA;
           mWINDOW AREA[mORIENTATION]+=mAREA;
           mWINROOF_AREA[mORIENTATION]+=mAREA;
           mTOTAL AREA[0]+=mAREA;
           mWINROOF AREA[0]+=mAREA;
           if (mU_VALUE>mWINROOF_U_MAX[mORIENTATION]) {
             for(j=0;j<10;j++)
               if(mU VALUE<=mWINROOF U MAX[j]) break;
             if (j==10)
                mWINROOF MAX=i;
             mWINROOF U MAX[mORIENTATION]=mU VALUE;
           }

           if (mU VALUE<mWINROOF U MIN[mORIENTATION]) {
             for(j=0;j<10;j++)
               if(mU VALUE>=mWINROOF U MIN[j]) break;
             if (j==10)
                mWINROOF MIN=i;
             mWINROOF U MIN[mORIENTATION]=mU VALUE;
           }

           if                    (mU VALUE>mWINDOW U MAX[mORIENTATION])
mWINDOW U MAX[mORIENTATION]=mU VALUE;
           mWINROOF U AVE[mORIENTATION]+=(mU VALUE*mAREA);
           mWINDOW_U_AVE[mORIENTATION]+=(mU_VALUE*mAREA);
           mBUILD U AVE[mORIENTATION]+=(mU VALUE*mAREA);
         }
         break;
         case 2: {
           mROOFLIGHT AREA[0]+=mAREA;
           mROOFLIGHT_AREA[mORIENTATION]+=mAREA;
           mWINROOF AREA[mORIENTATION]+=mAREA;
           mTOTAL AREA[0]+=mAREA;
           mWINROOF_AREA[0]+=mAREA;

           if (mU VALUE>mWINROOF U MAX[mORIENTATION]) {
             for(j=0;j<10;j++)
               if(mU VALUE<=mWINROOF U MAX[j]) break;
             if (j==10)
                mWINROOF_MAX=i;
             mWINROOF U MAX[mORIENTATION]=mU VALUE;
           }

           if (mU VALUE<mWINROOF U MIN[mORIENTATION]) {
             for(j=0;j<10;j++)
               if(mU_VALUE>=mWINROOF_U_MIN[j]) break;
             if (j==10)
                mWINROOF MIN=i;
             mWINROOF_U_MIN[mORIENTATION]=mU_VALUE;
           }

           if                (mU VALUE>mROOFLIGHT U MAX[mORIENTATION])
mROOFLIGHT U MAX[mORIENTATION]=mU VALUE;
           mWINROOF U AVE[mORIENTATION]+=(mU VALUE*mAREA);
           mROOFLIGHT_U_AVE[mORIENTATION]+=(mU_VALUE*mAREA);
```

```
            mBUILD U AVE[mORIENTATION]+=(mU VALUE*mAREA);
          }
          break;
          default: {
          }
        }
      }
    }
// adding checks for doors ID9a:
   else           if           ((P.vINP[i].Type=="DOOR")           &&
(P.Vcodid(hvac id,"TYPE")!=1)) {   //ignore doors in zones with no
HVAC - AN 01/02/07
     mAREA=P.Vnum(i,"AREA")*P.Vnum(i,"MULTIPLIER");
     mU_VALUE=P.Vnum(i,"U-VALUE");
     switch (P.Vcodid(i,"TYPE")) { //personel-door, vehicle-door or
high usage-door
       case 1: {
         mP DOOR AREA[0]+=mAREA;
         mP DOOR AREA[mORIENTATION]+=mAREA;
         mTOTAL_AREA[0]+=mAREA;

           if (mU VALUE>mP DOOR U MAX[mORIENTATION]) {
             for(j=0;j<10;j++)
               if(mU VALUE<=mP DOOR U MAX[j]) break;
             if (j==10)
                mP DOOR MAX=i;
             mP DOOR U MAX[mORIENTATION]=mU VALUE;
           }

           if (mU VALUE<mP DOOR U MIN[mORIENTATION]) {
             for(j=0;j<10;j++)
               if(mU VALUE>=mP DOOR U MIN[j]) break;
             if (j==10)
                mP DOOR MIN=i;
             mP DOOR U MIN[mORIENTATION]=mU VALUE;
           }

         if                     (mU VALUE>mP DOOR U MAX[mORIENTATION])
mP_DOOR_U_MAX[mORIENTATION]=mU_VALUE;

         if                     (mU VALUE<mP DOOR U MIN[mORIENTATION])
mP_DOOR_U_MIN[mORIENTATION]=mU_VALUE;

         mP DOOR U AVE[mORIENTATION]+=(mU VALUE*mAREA);
         mBUILD_U_AVE[mORIENTATION]+=(mU_VALUE*mAREA);
       }
       break;
       case 2: {
         mV DOOR AREA[0]+=mAREA;
         mV DOOR AREA[mORIENTATION]+=mAREA;
         mTOTAL_AREA[0]+=mAREA;

           if (mU VALUE>mV DOOR U MAX[mORIENTATION]) {
             for(j=0;j<10;j++)
               if(mU VALUE<=mV DOOR U MAX[j]) break;
             if (j==10)
                mV_DOOR_MAX=i;
             mV DOOR U MAX[mORIENTATION]=mU VALUE;
           }

           if (mU VALUE<mV DOOR U MIN[mORIENTATION]) {
             for(j=0;j<10;j++)
               if(mU_VALUE>=mV_DOOR_U_MIN[j]) break;
             if (j==10)
                mV DOOR MIN=i;
             mV_DOOR_U_MIN[mORIENTATION]=mU_VALUE;
           }

         if                     (mU VALUE>mV DOOR U MAX[mORIENTATION])
```

```
mV DOOR U MAX[mORIENTATION]=mU VALUE;

          if                        (mU VALUE<mV DOOR U MIN[mORIENTATION])
mV DOOR U MIN[mORIENTATION]=mU VALUE;

          mV DOOR U AVE[mORIENTATION]+=(mU VALUE*mAREA);
          mBUILD U AVE[mORIENTATION]+=(mU VALUE*mAREA);
        }
        break;
        case 3: {
          mH_DOOR_AREA[0]+=mAREA;
          mH DOOR AREA[mORIENTATION]+=mAREA;
          mTOTAL AREA[0]+=mAREA;

            if (mU VALUE>mH DOOR U MAX[mORIENTATION]) {
              for(j=0;j<10;j++)
                if(mU_VALUE<=mH_DOOR_U_MAX[j]) break;
              if (j==10)
                 mH DOOR MAX=i;
              mH_DOOR_U_MAX[mORIENTATION]=mU_VALUE;
            }

            if (mU VALUE<mH DOOR U MIN[mORIENTATION]) {
              for(j=0;j<10;j++)
                if(mU VALUE>=mH DOOR U MIN[j]) break;
              if (j==10)
                 mH DOOR MIN=i;
              mH DOOR U MIN[mORIENTATION]=mU VALUE;
            }

          if                        (mU VALUE>mH DOOR U MAX[mORIENTATION])
mH DOOR U MAX[mORIENTATION]=mU VALUE;

          if                        (mU VALUE<mH DOOR U MIN[mORIENTATION])
mH DOOR U MIN[mORIENTATION]=mU VALUE;

          mH DOOR U AVE[mORIENTATION]+=(mU VALUE*mAREA);
          mBUILD U AVE[mORIENTATION]+=(mU VALUE*mAREA);
        }
        break;
        default: {
        }
      }
    }
}

// area weighted infiltration:
  INF_AREA=P.Vnum(Id,"Q50-INF-AVE");


// Testing if a compliance check on air permeability is applicable on
this building - AN 02/02/07
 bool INF CHECK=0;
if ((P.Vcodid(Id,"BR-TYPE"))!=3 && (P.Vcodid(Id,"EPC-TYPE"))!=4) {
   if (redondea(INF AREA,2)==D.Value("Q50 INF MAX2"))
     if                                    ((P.Vnum(gId,"BUILDING-AREA-
CALC"))<(D.Value("Q50_INF_AREA")))
        INF CHECK=1;
 }


//area weighted average U-values:
 for (int i=1;i<10;i++) {
   if(mWALL U MAX[0]<mWALL U MAX[i]) mWALL U MAX[0]=mWALL U MAX[i];
   if(mROOF U MAX[0]<mROOF U MAX[i]) mROOF U MAX[0]=mROOF U MAX[i];
   if(mFLOOR_U_MAX[0]<mFLOOR_U_MAX[i])
mFLOOR U MAX[0]=mFLOOR U MAX[i];
   if(mWINDOW U MAX[0]<mWINDOW U MAX[i])
mWINDOW_U_MAX[0]=mWINDOW_U_MAX[i];
```

```
    if(mP DOOR U MAX[0]<mP DOOR U MAX[i])
mP DOOR U MAX[0]=mP DOOR U MAX[i];
    if(mV DOOR U MAX[0]<mV DOOR U MAX[i])
mV DOOR U MAX[0]=mV DOOR U MAX[i];
    if(mH_DOOR_U_MAX[0]<mH_DOOR_U_MAX[i])
mH DOOR U MAX[0]=mH DOOR U MAX[i];
    if(mS VENT U MAX[0]<mH DOOR U MAX[i])
mS_VENT_U_MAX[0]=mS_VENT_U_MAX[i];
    if(mWINROOF U MAX[0]<mWINROOF U MAX[i])
mWINROOF U MAX[0]=mWINROOF U MAX[i];
    if(mROOFLIGHT_U_MAX[0]<mROOFLIGHT_U_MAX[i])
mROOFLIGHT U MAX[0]=mROOFLIGHT U MAX[i];
    if(mWALL WIN U MAX[0]<mWALL WIN U MAX[i])
mWALL_WIN_U_MAX[0]=mWALL_WIN_U_MAX[i];
    if(mROOF WIN U MAX[0]<mROOF WIN U MAX[i])
mROOF WIN U MAX[0]=mROOF WIN U MAX[i];

    if(mWALL U MIN[0]>mWALL U MIN[i]) mWALL U MIN[0]=mWALL U MIN[i];
    if(mROOF U MIN[0]>mROOF U MIN[i]) mROOF U MIN[0]=mROOF U MIN[i];
    if(mFLOOR_U_MIN[0]>mFLOOR_U_MIN[i])
mFLOOR U MIN[0]=mFLOOR U MIN[i];
    if(mP DOOR U MIN[0]>mP DOOR U MIN[i])
mP DOOR U MIN[0]=mP DOOR U MIN[i];
    if(mV DOOR U MIN[0]>mV DOOR U MIN[i])
mV DOOR U MIN[0]=mV DOOR U MIN[i];
    if(mH DOOR U MIN[0]>mH DOOR U MIN[i])
mH DOOR U MIN[0]=mH DOOR U MIN[i];
    if(mWINROOF U MIN[0]>mWINROOF U MIN[i])
mWINROOF U MIN[0]=mWINROOF U MIN[i];

    mWALL U AVE[0]+=mWALL U AVE[i];
    mROOF U AVE[0]+=mROOF U AVE[i];
    mFLOOR U AVE[0]+=mFLOOR U AVE[i];
    mWINDOW U AVE[0]+=mWINDOW U AVE[i];
    mP DOOR U AVE[0]+=mP DOOR U AVE[i];
    mV DOOR U AVE[0]+=mV DOOR U AVE[i];
    mH DOOR U AVE[0]+=mH DOOR U AVE[i];
    mS VENT U AVE[0]+=mS VENT U AVE[i];
    mROOFLIGHT_U_AVE[0]+=mROOFLIGHT_U_AVE[i];
    mWINROOF U AVE[0]+=mWINROOF U AVE[i];
    mBUILD U AVE[0]+=mBUILD U AVE[i];

    mWALL WIN U AVE[i]+=mWALL U AVE[i]+mWINDOW U AVE[i];
    mROOF WIN U AVE[i]+=mROOF U AVE[i]+mROOFLIGHT U AVE[i];


mTOTAL AREA[i]+=(mWALL AREA[i]+mROOF AREA[i]+mFLOOR AREA[i]+mWINDOW A
REA[i]+mP_DOOR_AREA[i]+mV_DOOR_AREA[i]+mH_DOOR_AREA[i]+mS_VENT_AREA[i
]+mROOFLIGHT AREA[i]);
 }

 for (int i=0;i<10;i++) {
    if(mWALL AREA[i] != 0.0) mWALL U AVE[i]/=mWALL AREA[i];
    if(mROOF_AREA[i] != 0.0) mROOF_U_AVE[i]/=mROOF_AREA[i];
    if(mFLOOR AREA[i] != 0.0) mFLOOR U AVE[i]/=mFLOOR AREA[i];
    if(mWINDOW AREA[i] != 0.0) mWINDOW U AVE[i]/=mWINDOW AREA[i];
    if(mP_DOOR_AREA[i] != 0.0) mP_DOOR_U_AVE[i]/=mP_DOOR_AREA[i];
    if(mV DOOR AREA[i] != 0.0) mV DOOR U AVE[i]/=mV DOOR AREA[i];
    if(mH DOOR AREA[i] != 0.0) mH DOOR U AVE[i]/=mH DOOR AREA[i];
    if(mS_VENT_AREA[i] != 0.0) mS_VENT_U_AVE[i]/=mS_VENT_AREA[i];
    if(mROOFLIGHT AREA[i]                    !=                    0.0)
mROOFLIGHT U AVE[i]/=mROOFLIGHT AREA[i];
    if(mWINROOF_AREA[i] != 0.0) mWINROOF_U_AVE[i]/=mWINROOF_AREA[i];
    if(mTOTAL AREA[i] != 0.0) mBUILD U AVE[i]/=mTOTAL AREA[i];

    mROOF_WIN_AREA[i]=mROOFLIGHT_AREA[i]+mROOF_AREA[i];
    mWALL WIN AREA[i]=mWINDOW AREA[i]+mWALL AREA[i];

    if(mROOF WIN AREA[i]                    !=                    0.0)
```

```
mROOF WIN U AVE[i]/=mROOF WIN AREA[i];
   if(mWALL WIN AREA[i]                    !=                    0.0)
mWALL WIN U AVE[i]/=mWALL WIN AREA[i];
 }

 {
  ostringstream res; res << INF AREA;
  P.ModKey(Id,"Q50-INF-AVE",res.str());
 }

// Writing a keyword indicating whether an air permeability check is
applicable on this building - AN 02/02/07
 if (INF CHECK) {
   {
    ostringstream res; res << "NO";
    P.ModKey(Id,"Q50-INF-CHECK",res.str());
   }
 }

 {
  ostringstream res; res << mWALL MAX;
  P.ModKey(Id,"WALL-MAX",res.str());
 }
 {
  ostringstream res; res << mFLOOR MAX;
  P.ModKey(Id,"FLOOR-MAX",res.str());
 }
 {
  ostringstream res; res << mROOF MAX;
  P.ModKey(Id,"ROOF-MAX",res.str());
 }
 {
  ostringstream res; res << mWINROOF MAX;
  P.ModKey(Id,"WINROOF-MAX",res.str());
 }
 {
  ostringstream res; res << mP DOOR MAX;
  P.ModKey(Id,"P-DOOR-MAX",res.str());
 }
 {
  ostringstream res; res << mH DOOR MAX;
  P.ModKey(Id,"H-DOOR-MAX",res.str());
 }
 {
  ostringstream res; res << mV_DOOR_MAX;
  P.ModKey(Id,"V-DOOR-MAX",res.str());
 }
 {
  ostringstream res; res << mS VENT MAX;
  P.ModKey(Id,"S-VENT-MAX",res.str());
 }

 {
  ostringstream res; res << mWALL_MIN;
  P.ModKey(Id,"WALL-MIN",res.str());
 }
 {
  ostringstream res; res << mFLOOR MIN;
  P.ModKey(Id,"FLOOR-MIN",res.str());
 }
 {
  ostringstream res; res << mROOF MIN;
  P.ModKey(Id,"ROOF-MIN",res.str());
 }
 {
  ostringstream res; res << mWINROOF_MIN;
  P.ModKey(Id,"WINROOF-MIN",res.str());
 }
 {
```

```
 ostringstream res; res << mP DOOR MIN;
 P.ModKey(Id,"P-DOOR-MIN",res.str());
}
{
 ostringstream res; res << mH_DOOR_MIN;
 P.ModKey(Id,"H-DOOR-MIN",res.str());
}
{
 ostringstream res; res << mV DOOR MIN;
 P.ModKey(Id,"V-DOOR-MIN",res.str());
}

{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mROOF U AVE[i] << ", ";
 res << mROOF_U_AVE[9] << " }";
 P.ModKey(Id,"ROOF-U-AVE",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mWALL U AVE[i] << ", ";
 res << mWALL U AVE[9] << " }";
 P.ModKey(Id,"WALL-U-AVE",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mFLOOR U AVE[i] << ", ";
 res << mFLOOR U AVE[9] << " }";
 P.ModKey(Id,"FLOOR-U-AVE",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mWINDOW U AVE[i] << ", ";
 res << mWINDOW U AVE[9] << " }";
 P.ModKey(Id,"WINDOW-U-AVE",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
    res << mP DOOR U AVE[i] << ", ";
 res << mP_DOOR_U_AVE[9] << " }";
 P.ModKey(Id,"P-DOOR-U-AVE",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mV_DOOR_U_AVE[i] << ", ";
 res << mV DOOR U AVE[9] << " }";
 P.ModKey(Id,"V-DOOR-U-AVE",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mH DOOR U AVE[i] << ", ";
 res << mH DOOR U AVE[9] << " }";
 P.ModKey(Id,"H-DOOR-U-AVE",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mS VENT U AVE[i] << ", ";
 res << mS_VENT_U_AVE[9] << " }";
 P.ModKey(Id,"S-VENT-U-AVE",res.str());
}
{
```

```
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mWINROOF U AVE[i] << ", ";
 res << mWINROOF U AVE[9] << " }";
 P.ModKey(Id,"WINROOF-U-AVE",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mROOFLIGHT U AVE[i] << ", ";
 res << mROOFLIGHT_U_AVE[9] << " }";
 P.ModKey(Id,"ROOFLIGHT-U-AVE",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mBUILD_U_AVE[i] << ", ";
 res << mBUILD U AVE[9] << " }";
 P.ModKey(Id,"BUILD-U-AVE",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mROOF WIN U AVE[i] << ", ";
 res << mROOF WIN U AVE[9] << " }";
 P.ModKey(Id,"ROOF-WIN-U-AVE",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mWALL WIN U AVE[i] << ", ";
 res << mWALL WIN U AVE[9] << " }";
 P.ModKey(Id,"WALL-WIN-U-AVE",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mROOF U MAX[i] << ", ";
 res << mROOF_U_MAX[9] << " }";
 P.ModKey(Id,"ROOF-U-MAX",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
    res << mWALL_U_MAX[i] << ", ";
 res << mWALL U MAX[9] << " }";
 P.ModKey(Id,"WALL-U-MAX",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mFLOOR U MAX[i] << ", ";
 res << mFLOOR U MAX[9] << " }";
 P.ModKey(Id,"FLOOR-U-MAX",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mWINDOW U MAX[i] << ", ";
 res << mWINDOW_U_MAX[9] << " }";
 P.ModKey(Id,"WINDOW-U-MAX",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mP_DOOR_U_MAX[i] << ", ";
 res << mP DOOR U MAX[9] << " }";
 P.ModKey(Id,"P-DOOR-U-MAX",res.str());
}
```

```
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mV_DOOR_U_MAX[i] << ", ";
 res << mV_DOOR_U_MAX[9] << " }";
 P.ModKey(Id,"V-DOOR-U-MAX",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mH_DOOR_U_MAX[i] << ", ";
 res << mH_DOOR_U_MAX[9] << " }";
 P.ModKey(Id,"H-DOOR-U-MAX",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
    res << mS_VENT_U_MAX[i] << ", ";
 res << mS_VENT_U_MAX[9] << " }";
 P.ModKey(Id,"S-VENT-U-MAX",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mWINROOF_U_MAX[i] << ", ";
 res << mWINROOF_U_MAX[9] << " }";
 P.ModKey(Id,"WINROOF-U-MAX",res.str());
}
{
ostringstream res; res << "{ ";
for (int i=0;i<9;i++)
    res << mROOFLIGHT_U_MAX[i] << ", ";
res << mROOFLIGHT_U_MAX[9] << " }";
P.ModKey(Id,"ROOFLIGHT-U-MAX",res.str());
}

{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mROOF_U_MIN[i] << ", ";
 res << mROOF_U_MIN[9] << " }";
 P.ModKey(Id,"ROOF-U-MIN",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
    res << mWALL_U_MIN[i] << ", ";
 res << mWALL_U_MIN[9] << " }";
 P.ModKey(Id,"WALL-U-MIN",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mFLOOR_U_MIN[i] << ", ";
 res << mFLOOR_U_MIN[9] << " }";
 P.ModKey(Id,"FLOOR-U-MIN",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mP_DOOR_U_MIN[i] << ", ";
 res << mP_DOOR_U_MIN[9] << " }";
 P.ModKey(Id,"P-DOOR-U-MIN",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mV_DOOR_U_MIN[i] << ", ";
 res << mV_DOOR_U_MIN[9] << " }";
```

```
 P.ModKey(Id,"V-DOOR-U-MIN",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mH DOOR U MIN[i] << ", ";
 res << mH DOOR U MIN[9] << " }";
 P.ModKey(Id,"H-DOOR-U-MIN",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mWINROOF U MIN[i] << ", ";
 res << mWINROOF_U_MIN[9] << " }";
 P.ModKey(Id,"WINROOF-U-MIN",res.str());
}

{
ostringstream res; res << "{ ";
for (int i=0;i<9;i++)
    res << mROOF AREA[i] << ", ";
res << mROOF AREA[9] << " }";
P.ModKey(Id,"ROOF-AREA",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mWALL AREA[i] << ", ";
 res << mWALL AREA[9] << " }";
 P.ModKey(Id,"WALL-AREA",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mFLOOR AREA[i] << ", ";
 res << mFLOOR AREA[9] << " }";
 P.ModKey(Id,"FLOOR-AREA",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mWINDOW_AREA[i] << ", ";
 res << mWINDOW AREA[9] << " }";
 P.ModKey(Id,"WINDOW-AREA",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
   res << mP DOOR AREA[i] << ", ";
 res << mP DOOR AREA[9] << " }";
 P.ModKey(Id,"P-DOOR-AREA",res.str());
}
{
ostringstream res; res << "{ ";
for (int i=0;i<9;i++)
    res << mV DOOR AREA[i] << ", ";
res << mV_DOOR_AREA[9] << " }";
P.ModKey(Id,"V-DOOR-AREA",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
     res << mH_DOOR_AREA[i] << ", ";
 res << mH DOOR AREA[9] << " }";
 P.ModKey(Id,"H-DOOR-AREA",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<9;i++)
```

```
      res << mS VENT AREA[i] << ", ";
  res << mS VENT AREA[9] << " }";
  P.ModKey(Id,"S-VENT-AREA",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<9;i++)
      res << mROOF_WIN_AREA[i] << ", ";
  res << mROOF WIN AREA[9] << " }";
  P.ModKey(Id,"ROOF-WIN-AREA",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<9;i++)
      res << mWALL WIN AREA[i] << ", ";
  res << mWALL WIN AREA[9] << " }";
  P.ModKey(Id,"WALL-WIN-AREA",res.str());
  }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<9;i++)
      res << mWALL WIN U MAX[i] << ", ";
  res << mWALL WIN U MAX[9] << " }";
  P.ModKey(Id,"WALL-WIN-U-MAX",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<9;i++)
      res << mROOF WIN U MAX[i] << ", ";
  res << mROOF WIN U MAX[9] << " }";
  P.ModKey(Id,"ROOF-WIN-U-MAX",res.str());
 }
}

void COMPLIANCE::fSOLAR GAIN CHECK(PROJECTDefaults& D, PROJECTTables&
T, BOC& P) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"COMPLIANCE::fSOLAR GAIN CHECK";

//at the first encounter of a ZONE with a solar gain check, change
the flag for the building

  for (int i=0;i<P.vINP.size();i++) {
    if (P.vINP[i].Type=="ZONE")
      {
        ostringstream res;
        if(P.Vcodid(i,"SOLAR-GAIN-CHECK")==1) {     // 1 = YES
          res << "YES";
          P.ModKey(Id,"SOLAR-GAIN-CHECK",res.str());    // default is
YES
          break;
        }
      }
  } // end for

}

void        COMPLIANCE::fACCRED CONST CHECK(PROJECTDefaults&        D,
PROJECTTables& T, BOC& P) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"COMPLIANCE::fACCRED CONST CHECK";
  // Checks if any thermal bridge junction in any zone has been
defined with accredited construction details and
  // updates the flag for reporting key features in BRUKL

  // Non-metal cladding junctions
  deque<double> mJNCT ACCR DETAILS;
  // Metal cladding junctions
  deque<double> mJNCT_ACCR_DETAILS_MC;
```

```
  bool ACD flag=0;

  for (int i=0;i<P.vINP.size();i++) {
    if (P.vINP[i].Type=="ZONE") {
      // Exit the for loop as soon as any zone turns the overall flag
to yes
      if(ACD_flag==1)
        break;

      mJNCT_ACCR_DETAILS=P.Vlnum(i,"JNCT-ACCR-DETAIL");
      mJNCT ACCR DETAILS MC=P.Vlnum(i,"JNCT-ACCR-DETAIL-MC");

      // Check non-metal cladding junctions
      for (int j=0;j<mJNCT ACCR DETAILS.size();j++)
        // Check if any thermal bridge junction in any zone has been
defined with accredited construction details
        if (mJNCT ACCR DETAILS[j]>0.0) {
          ACD flag=1;
          break;
        }

      // Check metal cladding junctions
      for (int j=0;j<mJNCT ACCR DETAILS MC.size();j++)
        // Check if any thermal bridge junction in any zone has been
defined with accredited construction details
        if (mJNCT ACCR DETAILS MC[j]>0.0) {
          ACD flag=1;
          break;
        }
    } // end if
  } // end for


  if(ACD flag==1)
  {
    ostringstream res;
    res << "YES";
    P.ModKey(Id,"ACCRED-CONST-DETAILS",res.str());   // default is NO
  }

} // end fACCRED CONST CHECK
```

CONSTRUCTION

Specifies the construction characteristics and properties of a construction element (wall, roof, floor/ceiling, or opaque door). A U-name must be specified so that the CONSTRUCTION may be referenced in a subsequent WALL or DOOR objects.

Example input:

```
"Concrete Wall"     = CONSTRUCTION
   U-VALUE          = 0.32
   CM               = { 51, 51 }
   METAL-CLADDING   = NO
   ..
```

U-VALUE

For exterior surfaces, this equals the overall conductance of the surface including the inside and outside air films, in W/m^2K. For underground surfaces, it includes the inside air film. This is a required keyword.

U-VALUE-CORR
This keyword specifies whether the U-value of the ground floor construction was modified according to guidance specified in section 3.5.2 of "CIBSE Guide - Vol. A – 2007" (ISO 13370:2007 – "Thermal Performance of Buildings – Heat Transfer via the Ground – Calculation Methods" or BR 443:2006 – "Conventions for U-value Calculations") to account for the heat loss through floors in contact with the ground. The value can be YES (then SBEM will use the U-value as input) or NO (then SBEM will make the modification according to the abovementioned guidance, which is always the case when the floor construction is selected from the Library or using Inference procedures). The default is NO.

CM
The internal heat capacity associated with the construction, calculated according to the CEN Standards, in kJ/m^{2}K, in the form shown below (enclosed by braces and separated by commas). This is a required keyword.

{ cm_1, cm_2 }
* Note that SBEM uses the second value, cm_2, only during calculation.

METAL-CLADDING
This keyword specifies whether the construction involves metal-cladding. The value can be YES or NO. The default is NO.

c++ code

```
//: construction.cpp

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void CONSTRUCTION::TEST() {
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test CONSTRUCTION" << endl;
//  cout << "Fin de Test CONSTRUCTION" << endl;
  BEEMObject::TEST();
}

void CONSTRUCTION::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC& P)
{
  if(func report) cout << endl << "  " << "CONSTRUCTION::Calc";
  if(calc_report) cout << endl << P.vINP[Id].Name;
}

void CONSTRUCTION::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC
&P, BOC &N, int not case) {
  if(func report) cout << endl << "  " << "CONSTRUCTION::Notional  "
<< not_case;

//Notional Pitched Roof:
  {
    BObject Temp;
    Temp.Ntype=2;
    Temp.Type="CONSTRUCTION";
    Temp.Name="Notional Pitched Roof";
    KeywordValue KV;
   {
    ostringstream res; res << "U VALUE";
    KV.Reset(); KV.Init(4,res.str());
    Temp.Prop.insert(make_pair("TYPE",KV));
```

```
    }
    {
     ostringstream              res;              res              <<
T.Value("Not U Values",1,not case)*T.Value("Not U Value Factor",1,not
_case);
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make pair("U-VALUE",KV));
    }
    {
     ostringstream      res;      res      <<      "{      "      <<
T.Value("Not CM Values",1,not case)       <<       ",       "       <<
T.Value("Not CM Values",1,not case) << " }";
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make_pair("CM",KV));
    }
     N.vINP.push back(Temp);
   }

//Notional Pitched Roof with metal cladding:
   {
     BObject Temp;
     Temp.Ntype=2;
     Temp.Type="CONSTRUCTION";
     Temp.Name="Notional Pitched Roof MC";
     KeywordValue KV;
    {
     ostringstream res; res << "U VALUE";
     KV.Reset(); KV.Init(4,res.str());
     Temp.Prop.insert(make pair("TYPE",KV));
    }
    {
     ostringstream              res;              res              <<
T.Value("Not U Values",1,not case)*T.Value("Not U Value Factor",1,not
 case);
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make pair("U-VALUE",KV));
    }
    {
     ostringstream      res;      res      <<      "{      "      <<
T.Value("Not CM Values MC",1,not case)       <<       ",       "       <<
T.Value("Not CM Values MC",1,not case) << " }";
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("CM",KV));
    }
    {
     ostringstream res; res << "YES";
     KV.Reset(); KV.Init(4,res.str());
     Temp.Prop.insert(make_pair("METAL-CLADDING",KV));
    }

     N.vINP.push_back(Temp);
   }


//Notional Flat Roof:
   {
     BObject Temp;
     Temp.Ntype=2;
     Temp.Type="CONSTRUCTION";
     Temp.Name="Notional Flat Roof";
     KeywordValue KV;
    {
     ostringstream res; res << "U_VALUE";
     KV.Reset(); KV.Init(4,res.str());
     Temp.Prop.insert(make pair("TYPE",KV));
    }
    {
     ostringstream              res;              res              <<
T.Value("Not_U_Values",2,not_case)*T.Value("Not_U_Value_Factor",1,not
```

```
 case);
    KV.Reset(); KV.Init(1,res.str());
    Temp.Prop.insert(make pair("U-VALUE",KV));
   }
   {
    ostringstream      res;      res      <<      "{      "      <<
T.Value("Not CM Values",2,not case)      <<      ",      "      <<
T.Value("Not_CM_Values",2,not_case) << " }";
    KV.Reset(); KV.Init(5,res.str());
    Temp.Prop.insert(make pair("CM",KV));
   }
    N.vINP.push back(Temp);
  }

//Notional Flat Roof with metal cladding:
  {
    BObject Temp;
    Temp.Ntype=2;
    Temp.Type="CONSTRUCTION";
    Temp.Name="Notional Flat Roof MC";
    KeywordValue KV;
   {
    ostringstream res; res << "U VALUE";
    KV.Reset(); KV.Init(4,res.str());
    Temp.Prop.insert(make pair("TYPE",KV));
   }
   {
    ostringstream              res;              res              <<
T.Value("Not U Values",2,not case)*T.Value("Not U Value Factor",1,not
 case);
    KV.Reset(); KV.Init(1,res.str());
    Temp.Prop.insert(make pair("U-VALUE",KV));
   }
   {
    ostringstream      res;      res      <<      "{      "      <<
T.Value("Not CM Values MC",2,not case)      <<      ",      "      <<
T.Value("Not CM Values MC",2,not case) << " }";
    KV.Reset(); KV.Init(5,res.str());
    Temp.Prop.insert(make_pair("CM",KV));
   }
   {
    ostringstream res; res << "YES";
    KV.Reset(); KV.Init(4,res.str());
    Temp.Prop.insert(make pair("METAL-CLADDING",KV));
   }

    N.vINP.push back(Temp);
  }


//Notional Exposed Wall:
  {
    BObject Temp;
    Temp.Ntype=2;
    Temp.Type="CONSTRUCTION";
    Temp.Name="Notional Exposed Wall";
    KeywordValue KV;
   {
    ostringstream res; res << "U VALUE";
    KV.Reset(); KV.Init(4,res.str());
    Temp.Prop.insert(make pair("TYPE",KV));
   }
   {
    ostringstream              res;              res              <<
T.Value("Not U Values",3,not case)*T.Value("Not U Value Factor",1,not
_case);
    KV.Reset(); KV.Init(1,res.str());
    Temp.Prop.insert(make pair("U-VALUE",KV));
   }
```

```
    {
     ostringstream      res;      res      <<      "{      "      <<
T.Value("Not CM Values",3,not case)       <<       ",       "       <<
T.Value("Not CM Values",3,not case) << " }";
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("CM",KV));
    }
     N.vINP.push_back(Temp);
   }

//Notional Exposed Wall with metal cladding:
   {
     BObject Temp;
     Temp.Ntype=2;
     Temp.Type="CONSTRUCTION";
     Temp.Name="Notional Exposed Wall MC";
     KeywordValue KV;
    {
     ostringstream res; res << "U VALUE";
     KV.Reset(); KV.Init(4,res.str());
     Temp.Prop.insert(make pair("TYPE",KV));
    }
    {
     ostringstream              res;              res              <<
T.Value("Not U Values",3,not case)*T.Value("Not U Value Factor",1,not
 case);
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make pair("U-VALUE",KV));
    }
    {
     ostringstream      res;      res      <<      "{      "      <<
T.Value("Not CM Values MC",3,not case)      <<      ",      "      <<
T.Value("Not CM Values MC",3,not case) << " }";
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("CM",KV));
    }
    {
     ostringstream res; res << "YES";
     KV.Reset(); KV.Init(4,res.str());
     Temp.Prop.insert(make pair("METAL-CLADDING",KV));
    }
     N.vINP.push_back(Temp);
   }

//Notional Internal Wall:
   {
     BObject Temp;
     Temp.Ntype=2;
     Temp.Type="CONSTRUCTION";
     Temp.Name="Notional Internal Wall";
     KeywordValue KV;
    {
     ostringstream res; res << "U VALUE";
     KV.Reset(); KV.Init(4,res.str());
     Temp.Prop.insert(make pair("TYPE",KV));
    }
    {
     ostringstream              res;              res              <<
T.Value("Not U Values",9,not case)*T.Value("Not U Value Factor",1,not
_case);
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make pair("U-VALUE",KV));
    }
    {
     ostringstream      res;      res      <<      "{      "      <<
T.Value("Not CM Values",9,not case)       <<       ",       "       <<
T.Value("Not CM Values",9,not case) << " }";
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("CM",KV));
```

```
    }
     N.vINP.push_back(Temp);
   }

//Notional Personnel door:
   {
     BObject Temp;
     Temp.Ntype=2;
     Temp.Type="CONSTRUCTION";
     Temp.Name="Notional Personnel Door";
     KeywordValue KV;
    {
     ostringstream res; res << "U VALUE";
     KV.Reset(); KV.Init(4,res.str());
     Temp.Prop.insert(make_pair("TYPE",KV));
    }
    {
     ostringstream               res;               res               <<
T.Value("Not U Values",5,not case)*T.Value("Not U Value Factor",1,not
_case);
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make_pair("U-VALUE",KV));
    }
    {
     ostringstream      res;       res       <<       "{       "       <<
T.Value("Not CM Values",5,not case)       <<       ",       "       <<
T.Value("Not CM Values",5,not case) << " }";
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make_pair("CM",KV));
    }
     N.vINP.push_back(Temp);
   }

//Notional vehicle door:
   {
     BObject Temp;
     Temp.Ntype=2;
     Temp.Type="CONSTRUCTION";
     Temp.Name="Notional Vehicle Door";
     KeywordValue KV;
    {
     ostringstream res; res << "U_VALUE";
     KV.Reset(); KV.Init(4,res.str());
     Temp.Prop.insert(make_pair("TYPE",KV));
    }
    {
     ostringstream               res;               res               <<
T.Value("Not_U_Values",7,not_case)*T.Value("Not_U_Value_Factor",1,not
 case);
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make_pair("U-VALUE",KV));
    }
    {
     ostringstream      res;       res       <<       "{       "       <<
T.Value("Not CM Values",7,not case)       <<       ",       "       <<
T.Value("Not CM Values",7,not case) << " }";
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make_pair("CM",KV));
    }
     N.vINP.push_back(Temp);
   }

//Notional High Usage Door:

  if(not case==4 || not case>=9)    // Scotland OR 2010 England and
Wales
     {
       BObject Temp;
       Temp.Ntype=2;
```

```
      Temp.Type="CONSTRUCTION";
      Temp.Name="Notional High Usage Entrance Door";
      KeywordValue KV;
     {
      ostringstream res; res << "U_VALUE";
      KV.Reset(); KV.Init(4,res.str());
      Temp.Prop.insert(make pair("TYPE",KV));
     }
     {
      ostringstream              res;              res              <<
T.Value("Not_U_Values",6,not_case)*T.Value("Not_U_Value_Factor",1,not
 case);
      KV.Reset(); KV.Init(1,res.str());
      Temp.Prop.insert(make_pair("U-VALUE",KV));
     }
     {
      ostringstream      res;      res      <<      "{      "      <<
T.Value("Not CM Values",6,not case)       <<       ",       "       <<
T.Value("Not CM Values",6,not case) << " }";
      KV.Reset(); KV.Init(5,res.str());
      Temp.Prop.insert(make pair("CM",KV));
     }
     N.vINP.push back(Temp);
    } // end if

//Notional Exposed Floor:
  {
    BObject Temp;
    Temp.Ntype=2;
    Temp.Type="CONSTRUCTION";
    Temp.Name="Notional Exposed Floor";
    KeywordValue KV;
   {
    ostringstream res; res << "U VALUE";
    KV.Reset(); KV.Init(4,res.str());
    Temp.Prop.insert(make pair("TYPE",KV));
   }
   {
    ostringstream              res;              res              <<
T.Value("Not U Values",4,not case)*T.Value("Not U Value Factor",1,not
 case); //Check 1.1 factor with Notinoal floor calculation
    KV.Reset(); KV.Init(1,res.str());
    Temp.Prop.insert(make pair("U-VALUE",KV));
   }
   {
    ostringstream      res;      res      <<      "{      "      <<
T.Value("Not CM Values",4,not case)       <<       ",       "       <<
T.Value("Not_CM_Values",4,not_case) << " }";
    KV.Reset(); KV.Init(5,res.str());
    Temp.Prop.insert(make pair("CM",KV));
   }
    N.vINP.push back(Temp);
  }

//Notional Internal Floor:
  {
    BObject Temp;
    Temp.Ntype=2;
    Temp.Type="CONSTRUCTION";
    Temp.Name="Notional Internal Floor";
    KeywordValue KV;
   {
    ostringstream res; res << "U_VALUE";
    KV.Reset(); KV.Init(4,res.str());
    Temp.Prop.insert(make pair("TYPE",KV));
   }
   {
    ostringstream              res;              res              <<
T.Value("Not_U_Values",10,not_case)*T.Value("Not_U_Value_Factor",1,no
```

```
t case); //Check 1.1 factor with Notinoal floor calculation
    KV.Reset(); KV.Init(1,res.str());
    Temp.Prop.insert(make pair("U-VALUE",KV));
   }
   {
    ostringstream      res;      res      <<      "{      "      <<
T.Value("Not CM Values",10,not case)      <<      ",      "      <<
T.Value("Not_CM_Values",10,not_case) << " }";
    KV.Reset(); KV.Init(5,res.str());
    Temp.Prop.insert(make pair("CM",KV));
   }
    N.vINP.push back(Temp);
  }

}


void CONSTRUCTION::Recommendation(PROJECTDefaults& D, PROJECTTables&
T, BOC& P) {
  if(func report)    cout    <<    endl    <<    "          "    <<
"CONSTRUCTION::Recommendation";

}



// End Object: CONSTRUCTION
```

DHW-GENERATOR

The DHW-GENERATOR command defines the performance parameters of the heat system for the provision of hot water. A U-name is required.

Example input:

```
"DHW boiler" = DHW-GENERATOR
   ISBEM-ID           = 11
   HEAT-GEN-TYPE      = Dedicated hot water boiler
   FUEL-TYPE          = Natural Gas
   HEAT-GEN-POST98    = NO
   STORE-SYSTEM       = NO
  ..
```

HEAT-GEN-TYPE

Specifies the type of heat generator used to produce hot water, as one of the values listed below. The default is 'Dedicated hot water boiler'.

Using Central Heating boiler
Dedicated hot water boiler
Stand-alone water heater
Instantaneous hot water only
Instantaneous combi
Direct (or off-peak) electricity (NOT IN USE)
Heat pump

BH-HEAT-GEN-TYPE

This keyword in included in the input if the hot water system (which is independent, i.e., the DHW-GENRATOR object's HEAT-GEN-TYPE keyword is set to any option other than 'Using Central Heating boiler') is bi-valent (or multi-valent), i.e., it is served by more than one heat generator. It specifies a list of the secondary heat generators that provide hot water, to support the primary heat generator already defined in the DHW-GENRATOR object's HEAT-GEN-TYPE keyword, in the form shown below (enclosed by braces and separated by commas). If the hot water system is served by only one heat generator, then this keyword is not included in the input.

{ "sec-heat-gen-type$_1$", "sec-heat-gen-type$_2$", .., .., .., .., "sec-heat-gen-type$_n$" } where sec-heat-gen-type$_i$ is one of the values listed below.

Dedicated hot water boiler
Stand-alone water heater
Instantaneous hot water only
Instantaneous combi
Heat pump

HEAT-SOURCE
Specifies the heat source for the HVAC system, which also provides the hot water, as one of the values listed below. The default is 'LTHW boiler' (low temperature hot water boiler). This keyword is included in the input if the DHW-GENRATOR object's HEAT-GEN-TYPE keyword is set to 'Using Central Heating boiler'.

LTHW boiler
MTHW boiler
HTHW boiler
Air heater
Unflued gas warm air heater
Unitary radiant heater
Radiant heater
Unflued radiant heater
Direct or storage electric heater
Heat pump (gas/oil): air source
Heat pump (gas/oil): ground or water source
Room heater
District heating
Direct gas firing
Heat pump (electric): air source
Heat pump (electric): ground or water source

BH-HEAT-SOURCE
This keyword in included in the input if the hot water is provided by an HVAC system (i.e., the DHW-GENERATOR object's HEAT-GEN-TYPE keyword is set to 'Using Central Heating boiler') which is bi-valent (or multi-valent), i.e., it is served by more than one heat source. It specifies a list of the secondary heat sources that provide heating, to support the primary heat source already defined in the DHW-GENERATOR object's HEAT-SOURCE keyword, in the form shown below (enclosed by braces and separated by commas). If the HVAC system providing the hot water is served by only one heat source, then this keyword is not included in the input.

{ "sec-heat-source$_1$", "sec-heat-source$_2$", .., .., .., .., "sec-heat-source$_n$" } where sec-heat-source$_i$ is one of the values listed below.

LTHW boiler
MTHW boiler
HTHW boiler
Air heater
Unflued gas warm air heater
Unitary radiant heater
Radiant heater
Unflued radiant heater
Direct or storage electric heater
Heat pump (gas/oil): air source
Heat pump (gas/oil): ground or water source
Room heater
District heating
Direct gas firing
Heat pump (electric): air source
Heat pump (electric): ground or water source

DHW-GEN-SEFF
This is the effective seasonal heat generating efficiency for the hot water generator. This keyword is included in the input if the DHW-GENERATOR object's HEAT-GEN-TYPE keyword is set to any option other than 'Using Central Heating boiler'.

BH-DHW-GEN-SEFF
This keyword in included in the input if the hot water system (which is independent, i.e., the DHW-GENERATOR object's HEAT-GEN-TYPE keyword is set to any option other than 'Using Central Heating boiler') is bi-valent (or multi-valent), i.e., it is served by more than one heat generator. It specifies a list of the effective seasonal heat generating efficiencies of the secondary heat generators that provide hot water (to support the primary heat generator already defined in the DHW-GENERATOR object's HEAT-GEN-TYPE keyword and whose effective seasonal heat generating efficiency is defined in the DHW-GENERATOR object's DHW-GEN-SEFF keyword), in the form shown below (enclosed by braces and separated by commas). The list of efficiencies in this keyword must correspond, in number and in order, to the heat generators listed in the DHW-GENERATOR object's BH-HEAT-GEN-TYPE keyword. If the hot water system is served by only one heat generator, then this keyword is not included in the input.

{ sec-heat-gen-seff$_1$, sec-heat-gen-seff$_2$, .., .., .., .., sec-heat-gen-seff$_n$ } where sec-heat-gen-seff$_i$ is a value for the effective seasonal heat generating efficiency for the heat generator.

ISBEM-ID *(RENAMED FROM DHW-ISBEM-ID)*
This keyword specifies the interface's ID number for the DHW-GENERATOR object, and it is used when calculated values relating to that object need to be reported within the interface.

***HEAT-GEN-POST98* (NO LONGER IN USE)**

For heat generator types 2 to 5 above, this keyword specifies whether the heat generator is post 98. The value can be YES or NO. The default is NO.

STORE-SYSTEM
Specifies whether the hot water system includes storage. The value can be YES or NO. The default is NO.

STORE-VOL
Where the hot water system has storage (i.e., the DHW-GENERATOR object's STORE-SYSTEM keyword is set to YES), this is the storage volume, in litres. If this keyword is not included in the input, its value is calculated by SBEM.

STORE-LOSS
Where the hot water system has storage (i.e., the DHW-GENERATOR object's STORE-SYSTEM keyword is set to YES), this is the storage loss, in MJ/month. The same value is used for all months. If this keyword is not included in the input, its value is calculated by SBEM, based on the information input in the DHW-GENERATOR object's STORE-INSULAT-TYPE and STORE-INSULAT-THICK keywords.

STORE-INSULAT-TYPE
Where the hot water system has storage (i.e., the DHW-GENERATOR object's STORE-SYSTEM keyword is set to YES), and the DHW-GENERATOR object's STORE-LOSS keyword has not been included in the input, this is the insulation type on the hot water storage vessel, as one of the values listed below. The default is 'Uninsulated'.

Uninsulated
Loose jacket
Factory insulated

STORE-INSULAT-THICK
Where the hot water system has storage (i.e., the DHW-GENERATOR object's STORE-SYSTEM keyword is set to YES), and the DHW-GENERATOR object's STORE-LOSS keyword has not been included in the input, this is the insulation thickness on the hot water storage vessel, in mm. The default is 0.

SEC-CIRCULATION
Specifies whether the hot water system has a secondary circulation. The value can be YES or NO. The default is NO.

SEC-LOOP-LENGTH
Where the hot water system has secondary circulation (i.e., the DHW-GENERATOR object's SEC-CIRCULATION keyword is set to YES), this keyword specifies the secondary circulation's loop length, in metres. If this keyword is not included in the input, its value is calculated by SBEM.

SEC-CIRC-LOSS/M
Where the hot water system has a secondary circulation (i.e., the DHW-GENERATOR object's SEC-CIRCULATION keyword is set to YES), this keyword specifies the secondary circulation's losses per unit length of secondary circulation pipework, in W/m. If this keyword is not included in the input, its value is calculated by SBEM.

SEC-CIRC-TS
Where the hot water system has a secondary circulation (i.e., the DHW-GENERATOR object's SEC-CIRCULATION keyword is set to YES), this keyword specifies whether there is a time switch on the secondary circulation. The value can be YES or NO. The default is NO.

SEC-PUMP-POWER
Where the hot water system has a secondary circulation (i.e., the DHW-GENERATOR object's SEC-CIRCULATION keyword is set to YES), this keyword specifies the secondary circulation's pump power, in kW. If this keyword is not included in the input, its value is calculated by SBEM.

HVAC-SYSTEM-EFF
This is the effective seasonal heat generating efficiency of the generator of the HVAC-SYSTEM which also provides the hot water. It is included in the input if the DHW-GENERATOR object's HEAT-GEN-TYPE keyword is set to 'Using Central Heating boiler'.

BH-HVAC-SYSTEM-EFF
This keyword in included in the input if the hot water is provided by an HVAC system (i.e., the DHW-GENERATOR object's HEAT-GEN-TYPE keyword is set to 'Using Central Heating boiler') which is bi-valent (or multi-valent), i.e., it is served by more than one heat source. It specifies a list of the effective seasonal heat generating efficiencies of the secondary heat sources that provide heating (to support the primary heat source already defined in the DHW-GENERATOR object's HEAT-SOURCE keyword and whose effective seasonal heat generating efficiency is defined in the DHW-GENERATOR object's HVAC-SYSTEM-EFF keyword), in the form shown below (enclosed by braces and separated by commas). The list of effective seasonal heat generator efficiencies in this keyword must correspond, in number and in order, to the heat sources listed in the DHW-GENERATOR object's BH-HEAT-SOURCE keyword. If the HVAC system providing the hot water is served by only one heat source (and therefore has one effective seasonal heat generator efficiency), then this keyword is not included in the input.

{ sec-hvac-seff$_1$, sec-hvac-seff$_2$, .., .., .., .., sec-hvac-seff$_n$ } where sec-hvac-seff$_i$ is a value for the effective seasonal heat generating efficiency for the heat generator of the HVAC system providing the hot water.

FUEL-TYPE
Specifies the fuel type for hot water generator, as one of the values listed below. The default is 'Natural Gas'. If the DHW-GENERATOR object's HEAT-GEN-TYPE keyword is set to 'Using Central Heating boiler', then the value for this keyword will be the same as that for the FUEL-TYPE keyword of the HVAC-SYSTEM object referred to.

Natural Gas
LPG
Biogas
Oil
Coal

Anthracite
Smokeless Fuel (inc Coke)
Dual Fuel Appliances (Mineral + Wood)
Biomass
Grid Supplied Electricity
Waste Heat
District Heating

BH-FUEL-TYPE
This keyword in included in the input if the hot water is provided by a system (whether an independent hot water system or through the HVAC system) which is bi-valent (or multi-valent), i.e., it is served by more than one heat generator/heat source. It specifies a list of the fuel types supplying the secondary heat generators/heat sources that provide hot water (to support the primary heat generator/heat source already defined in the DHW-GENERATOR object's HEAT-GEN-TYPE/HEAT-SOURCE keyword and which is powered by the fuel defined in the DHW-GENERATOR object's FUEL-TYPE keyword), in the form shown below (enclosed by braces and separated by commas). The list of fuel types in this keyword must correspond, in number and in order, to the heat generators/heat sources listed in the DHW-GENERATOR object's BH-HEAT-GEN-TYPE/BH-HEAT-SOURCE keyword. If the hot water system is served by only one heat generator/heat source (and one fuel type), then this keyword is not included in the input.

{ "sec-fuel-type$_1$", "sec-fuel-type$_2$", .., .., .., .., "sec-fuel-type$_n$" } where sec-fuel-type$_i$ is one of the values listed below.

Natural Gas
LPG
Biogas
Oil
Coal
Anthracite
Smokeless Fuel (inc Coke)
Dual Fuel Appliances (Mineral + Wood)
Biomass
Grid Supplied Electricity
Waste Heat
District Heating

BH-LOAD
This keyword in included in the input if the hot water is provided by a system (whether an independent hot water system or through the HVAC system) which is bi-valent (or multi-valent), i.e., it is served by more than one heat generator/heat source. It specifies a list of the partial heating loads of the secondary heat generators/heat sources that provide hot water (to support the primary heat generator/heat source already defined in the DHW-GENERATOR object's HEAT-GEN-TYPE/HEAT-SOURCE keyword), in the form shown below (enclosed by braces and separated by commas). The list of partial loads in this keyword must correspond, in number and in order, to the heat generators/heat sources listed in the DHW-GENERATOR object's BH-HEAT-GEN-TYPE/BH-HEAT-SOURCE keyword. If the hot water system is served by only one

heat generator/heat source (and therefore providing the whole load), then this keyword is not included in the input.

{ sec-heat-load$_1$, sec-heat-load$_2$, .., .., .., .., sec-heat-load$_n$ } where sec-heat-load$_i$ is a value for the partial load, in decimal form.

***MULTIPLIER* (NO LONGER IN USE)**
This keyword specifies how many DHW-GENERATORS with this description exist in the project. E.g., if the value of multiplier is 3, there are 3 DHW-GENERATORS each with identical keyword values in the project. This enables repeating DHW-GENERATORS to be entered quickly and easily. The default value is 1.

CHP
If the DHW-GENERATOR object's HEAT-GEN-TYPE keyword is set to 'Using Central Heating boiler', and the HVAC-SYSTEM referred to uses a CCHP generator as a heat source, there is the option for a proportion of the hot water load to also be provided by the same CCHP-GENERATOR object. This keyword specifies whether the CCHP-GENERATOR provides a proportion of the hot water. The value can be YES or NO. The default is NO.

c++ code

```
//: dhw generator.cpp

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void DHW GENERATOR::TEST() {
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test ZONE" << endl;
//  cout << "Fin de Test ZONE" << endl;
  BEEMObject::TEST();
}

void DHW_GENERATOR::Reset() {
  if(func report) cout << endl << "  " << "DHW GENERATOR::Reset";
  dS.clear();
//  vS.clear();
}

void DHW_GENERATOR::DtoV() {
  if(func report) cout << endl << "  " << "DHW GENERATOR::DtoV";
//  vS.reserve(dS.size());
//  vS.assign(dS.begin(),dS.end());
//  dS.clear();
}


void  DHW GENERATOR::Init fuel(PROJECTDefaults&  D,  PROJECTTables&  T,
BOC& P, PROJECTDefaults& HV, PROJECTTables& HVG) {
  if(func report) cout << endl << "  " << "DHW GENERATOR::Init fuel";
//This function determines whether the DHW generator uses gas as the
fuel type.
//This is so that SBEM can use oil as the notional fuel if there is
NO gas anywhere in the project.
//if DHW fuel is gas, set general keyword "NOT-FUEL" to "GAS" (ID=2)
//Note: There is a similar function in 'hvac.cpp': HVAC::Init fuel()

  //for DHW generator:
  if (P.Vcod(Id,"FUEL-TYPE")=="Natural Gas")
```

```
    //if HVAC fuel is gas, set general keyword "NOT-FUEL" to "GAS"
(ID=2):
    {
     ostringstream res; res << "GAS";
     P.ModKey(gId,"NOT-FUEL",res.str());
    }

} // end Init_Fuel


void DHW GENERATOR::Init(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func report) cout << endl << "  " << "DHW GENERATOR::Init";

 // Check if the actual building has a bivalent heating system, and
if yes, add the "main" system parameters to
 // the deque of supplementary systems. Otherwise, the main system is
the only system.

  // Calculate the equivalent system HW efficiency in the actual
building based on the combination of bivalent
  // systems
  if (P.Vcod(gId,"ACT-NOT")=="ACT") {   // if this is the actual
building


   deque<string> BH HS=P.Vltext(Id,"BH-HEAT-SOURCE");
   deque<string> BH FT=P.Vltext(Id,"BH-FUEL-TYPE");
   deque<double> BH EFF=P.Vlnum(Id,"BH-DHW-GEN-SEFF");
   deque<string> BH GT=P.Vltext(Id,"BH-HEAT-GEN-TYPE");
   deque<double> BH HVAC EFF=P.Vlnum(Id,"BH-HVAC-SYSTEM-EFF");
   deque<double> BH LOAD=P.Vlnum(Id,"BH-LOAD");

    // JO added, 12Nov2010, No District heating in Bivalent systems
    deque<int> toerase;
    for(int i=0;i<BH HS.size();i++)
      if(BH HS[i]=="District heating") toerase.push front(i);
    for(int i=0;i<toerase.size();i++) {
      BH_HS.erase(BH_HS.begin()+toerase[i]);
      BH FT.erase(BH FT.begin()+toerase[i]);
      BH EFF.erase(BH EFF.begin()+toerase[i]);
      BH_GT.erase(BH_GT.begin()+toerase[i]);
      BH HVAC EFF.erase(BH HVAC EFF.begin()+toerase[i]);
      BH LOAD.erase(BH LOAD.begin()+toerase[i]);
    }

    if (P.Vcodid(Id,"HEAT-SOURCE")==15) {
      BH_HS.clear();
      BH FT.clear();
      BH EFF.clear();
      BH_GT.clear();
      BH HVAC EFF.clear();
      BH LOAD.clear();
      BH_HS.push_back("-5555");
      BH FT.push back("-5555");
      BH EFF.push back(-5555);
      BH_GT.push_back("-5555");
      BH HVAC EFF.push back(-5555);
      BH LOAD.push back(-5555);
    }




       // Heat sources
       if (BH_HS[0]=="-5555")
         BH HS[0]=P.Vcod(Id,"HEAT-SOURCE");
       else
         BH_HS.push_back(P.Vcod(Id,"HEAT-SOURCE"));
```

```
       // Fuel types
       if (BH FT[0]=="-5555")
         BH FT[0]=P.Vcod(Id,"FUEL-TYPE");
       else
         BH FT.push back(P.Vcod(Id,"FUEL-TYPE"));

       // Generator types
       // If the hot water is provided by the HVAC system, create a
deque of heat generator types "Using Central Heating
       // boiler" a number of times equal to the number of fuel types
of the HVAC system in case it is bivalent.
       if (P.Vcodid(Id,"HEAT-GEN-TYPE")==1) {    // 1=Using Central
Heating boiler
         BH GT[0]=P.Vcod(Id,"HEAT-GEN-TYPE");
         // if the HVAC system is bivalent, i.e., there is more than
one fuel type
         if (BH FT.size()>1) {
           for (int i=1;i<BH FT.size();i++)
             BH_GT.push_back(P.Vcod(Id,"HEAT-GEN-TYPE"));
         }
       } // end else
       else { //i.e., the hot water system is independent of the HVAC
system
         if (BH GT[0]=="-5555")
           BH GT[0]=P.Vcod(Id,"HEAT-GEN-TYPE");
         else
           BH GT.push back(P.Vcod(Id,"HEAT-GEN-TYPE"));
       } // end else

       // Generator efficiency if provided by HVAC
       if (BH HVAC EFF[0]==-5555)
         BH HVAC EFF[0]=P.Vnum(Id,"HVAC-SYSTEM-EFF");
       else
         BH HVAC EFF.push back(P.Vnum(Id,"HVAC-SYSTEM-EFF"));

       // Generator efficiency if independent
       if (BH EFF[0]==-5555)
         BH_EFF[0]=P.Vnum(Id,"DHW-GEN-SEFF");
       else
         BH EFF.push back(P.Vnum(Id,"DHW-GEN-SEFF"));

       // Heat load
       double main load=1.0;

       if (BH LOAD[0]==-5555)
         BH LOAD[0]=main load;       // i.e., main system is the only
system and it provides all the load
       else {
         for(int i=0;i<BH LOAD.size();i++)
           main_load-=BH_LOAD[i];
         if (main load>0.0)
           BH LOAD.push back(main load);
       } // end else


       { // Save the updated deque of heat sources
        ostringstream res; res << "{ ";
        for (int i=0;i<BH HS.size()-1;i++)
          res << "\"" << BH_HS[i] << "\", ";
        res << "\"" << BH HS[BH HS.size()-1] << "\" }";
        P.ModKey(Id,"BH-HEAT-SOURCE",res.str());
       }

       { // Save the updated deque of fuel types
        ostringstream res; res << "{ ";
        for (int i=0;i<BH FT.size()-1;i++)
          res << "\"" << BH FT[i] << "\", ";
        res << "\"" << BH_FT[BH_FT.size()-1] << "\" }";
```

```
        P.ModKey(Id,"BH-FUEL-TYPE",res.str());
       }

       { // Save the updated deque of generator efficiencies if
independent
        ostringstream res; res << "{ ";
        for (int i=0;i<BH EFF.size()-1;i++)
          res << BH_EFF[i] << ", ";
        res << BH EFF[BH EFF.size()-1] << " }";
        P.ModKey(Id,"BH-DHW-GEN-SEFF",res.str());
       }

       { // Save the updated deque of generator type
        ostringstream res; res << "{ ";
        for (int i=0;i<BH GT.size()-1;i++)
          res << "\"" << BH GT[i] << "\", ";
        res << "\"" << BH_GT[BH_GT.size()-1] << "\" }";
        P.ModKey(Id,"BH-HEAT-GEN-TYPE",res.str());
       }

       { // Save the updated deque of generator efficiencies if
provided by HVAC
        ostringstream res; res << "{ ";
        for (int i=0;i<BH HVAC EFF.size()-1;i++)
          res << BH HVAC EFF[i] << ", ";
        res << BH HVAC EFF[BH HVAC EFF.size()-1] << " }";
        P.ModKey(Id,"BH-HVAC-SYSTEM-EFF",res.str());
       }

       { // Save the updated deque of generator efficiencies
        ostringstream res; res << "{ ";
        for (int i=0;i<BH LOAD.size()-1;i++)
          res << BH LOAD[i] << ", ";
        res << BH LOAD[BH LOAD.size()-1] << " }";
        P.ModKey(Id,"BH-LOAD",res.str());
       }


    if (BH GT.size()>1) { // If there is more than one generator type
defined (i.e., it is a bivalent system)

      double mHVAC_EFF=0.0;   // HVAC generator efficiency
      double mEFF=0.0;        // HW generator efficiency
      int FT Id;
      int GT_Id;

      for(int i=0;i<BH GT.size();i++) {

        FT Id=FuelTypeId(BH FT[i]);   // get CODID of fuel type
        GT Id=GenTypeId(BH GT[i]);    // get CODID of generator type

        // Check if the HW is provided by HVAC system or is
independent and modify relevant equivalent efficiency
        if (GT_Id==1)  // 1=Using Central Heating boiler

mHVAC EFF+=BH LOAD[i]*T.Value("tKGCO2",1,FT Id)/BH HVAC EFF[i];
        else // i.e., the HW is independent
          mEFF+=BH LOAD[i]*T.Value("tKGCO2",1,FT Id)/BH EFF[i];

      } // end for

      // Save the HVAC generator efficiency equivalent to the
combination of bivalent systems
      {
       ostringstream             res;             res             <<
T.Value("tKGCO2",1,P.Vcodid(Id,"FUEL-TYPE"))/mHVAC_EFF;
       P.ModKey(Id,"HVAC-SYSTEM-EFF",res.str());
      }
```

```
      // Save the HW generator efficiency equivalent to the
combination of bivalent systems
      {
       ostringstream             res;             res             <<
T.Value("tKGCO2",1,P.Vcodid(Id,"FUEL-TYPE"))/mEFF;
       P.ModKey(Id,"DHW-GEN-SEFF",res.str());
      }

    } // end if
  } // end if


} // end Init


int DHW GENERATOR::FuelNot2010(int HS, int FT, int GT) {

  int mNot Fuel;
  // Check if the hot water is provided by an electric heat pump
  // 7=Heat pump , 10=Grid Supplied Electricity
  if ( (GT==7) && (FT==10) )
    mNot Fuel=FT;    // same as heating fuel in the actual building
(grid electricity)

  else { // i.e., not an independent electric heat pump
    // Check if the hot water is provided by an HVAC system through
an electric heat pump
    // 1=Using Central Heating boiler , 18=Heat pump (electric): air
source ,
    // 19=Heat pump (electric): ground or water source , 10=Grid
Supplied Electricity
    if ( (GT==1) && ((HS==18) || (HS==19)) && (FT==10) )
         mNot Fuel=FT;     // same as heating fuel in the actual
building (grid electricity)

    else { // i.e., not an electric heat pump
      // Check if the heating fuel in the actual building is one
whose emission factor is >= that of oil,
      // i.e., >= 0.306 kg/kWh
      // 4=Oil, 5=Coal, 6=Anthracite, 7=Smokeless Fuel (inc Coke),
10=Grid Supplied Electricity
      if ( (FT==4) || (FT==5) || (FT==6) || (FT==7) || (FT==10) )
        mNot Fuel=4;

      else // i.e., the heating fuel in the actual building is one
whose emission factor < that of oil,
         // i.e., < 0.306 kg/kWh
        mNot_Fuel=FT;     // same as heating fuel in the actual
building
    } // end else
  } // end else

  return(mNot Fuel);

} // end FuelNot2010


double       DHW GENERATOR::fHEAT GEN EFF BH(PROJECTDefaults&       D,
PROJECTTables& T, BOC& P, string HS, string FT, string GT, double
BH_LOAD, int gId) {

  int HS Id=HeatSourceId(HS);
  int FT_Id=FuelTypeId(FT);
  int GT Id=GenTypeId(GT);

  int FTnoti=FuelNot2010(HS_Id, FT_Id, GT_Id);

  double mHEAT GEN EFF=0.0;
```

```
  // if hot water generator type is a heat pump
  if (GT Id==7) {
    // Check if the heating fuel in the actual building is grid
electricity, i.e., electric heat pump
    if (FT_Id==10)

mHEAT GEN EFF=T.Value("Not Water Heat SEff 4",1,P.Vint(gId,"NOT-
CASE"));
    else // i.e., non-electric heat pump

mHEAT_GEN_EFF=T.Value("Not_Water_Heat_SEff_3",1,P.Vint(gId,"NOT-
CASE"));
  } // end if
  else { // i.e., not an independent heat pump
    // Check if the hot water is provided by an HVAC system through
an electric heat pump
    // 1=Using Central Heating boiler , 18=Heat pump (electric): air
source , 19=Heat pump (electric): ground or water source
    // 10=Grid Supplied Electricity
    if ( (GT_Id==1) && ((HS_Id==18) || (HS_Id==19)) && (FT_Id==10) )

mHEAT GEN EFF=T.Value("Not Water Heat SEff 4",1,P.Vint(gId,"NOT-
CASE"));

    else {
      // Check if the hot water is provided by an HVAC system through
a non-electric heat pump
      // 1=Using Central Heating boiler , 12=Heat pump (gas/oil): air
source , 13=Heat pump (gas/oil): ground or water source
      if ( (GT Id==1) && ((HS Id==12) || (HS Id==13)) )

mHEAT GEN EFF=T.Value("Not Water Heat SEff 3",1,P.Vint(gId,"NOT-
CASE"));

      else {
        // Check if the heating fuel in the actual building is
district heating
        // 1=Using Central Heating boiler , 13=District Heating ,
15=District heating
        if ( (GT Id==1) && (FT Id==13) && (HS Id==15) ) {
          if ((P.Vnum(gId,"DH-CO2F"))<D.Value("dNOT DISTRICT HEAT1"))
            mHEAT_GEN_EFF=T.Value("Not_Water_Heat_SEff_1",1,9);
            else {
              if                                        ((P.Vnum(gId,"DH-
CO2F"))>D.Value("dNOT_DISTRICT_HEAT2"))
                mHEAT GEN EFF=T.Value("Not Water Heat SEff 2",1,9);
              else // i.e., 0.15 <= emission factor <= 0.2
                mHEAT_GEN_EFF=(((P.Vnum(gId,"DH-CO2F"))-
D.Value("dNOT DISTRICT HEAT1"))*(T.Value("Not Water Heat SEff 2",1,9)
-
T.Value("Not_Water_Heat_SEff_1",1,9))/(D.Value("dNOT_DISTRICT_HEAT2")
-
D.Value("dNOT DISTRICT HEAT1")))+T.Value("Not Water Heat SEff 1",1,9)
;
            } // end else
        } // end if

        else { // i.e., any case other than those above
          // Check if the heating fuel in the actual building is one
whose emission factor <= 0.156 kg/kWh
          // 3=Biogas, 9=Biomass, 12=Waste heat
          if ( (FT Id==3) || (FT Id==9) || (FT Id==12) )

mHEAT GEN EFF=T.Value("Not Water Heat SEff 1",1,P.Vint(gId,"NOT-
CASE"));
          else // i.e., the heating fuel in the actual building is
one whose emission factor > 0.156 kg/kWh

mHEAT_GEN_EFF=T.Value("Not_Water_Heat_SEff_2",1,P.Vint(gId,"NOT-
```

```
CASE"));
        } // end else
      } // end else
    } // end else
  } // end else

  return(BH LOAD*T.Value("tKGCO2",1,FTnoti)/mHEAT GEN EFF);

} // end fHEAT GEN EFF BH



void  DHW GENERATOR::Notional(PROJECTDefaults&  D,  PROJECTTables&  T,
BOC &P, BOC &N, int i) {
  if(func report) cout << endl << "   " << "DHW GENERATOR::Notional  "
<< i;
//create DHW generator for the notional building

  BObject Temp;
  Temp.Ntype=12;
  Temp.Type="DHW-GENERATOR";
  Temp.Name=P.vINP[i].Name;
  KeywordValue KV;

  string mNot Fuel;
 {
//  ostringstream res; res << "NOT";
  ostringstream res; res << P.Vcod(i,"HEAT-GEN-TYPE");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("HEAT-GEN-TYPE",KV));
 }

  if((P.Vcodid(cId,"BR-TYPE")==5)  ||  (P.Vcodid(cId,"EPC-TYPE")==6)){
//Republic of Ireland only
    // if fuel in the actual building is a low or zero carbon heating
technology (Biogas, Biomass, Waste Heat, or Heat pump),
    // then use natural gas instead
    if              ((P.Vcodid(i,"FUEL-TYPE")==3)||(P.Vcodid(i,"FUEL-
TYPE")==9)||(P.Vcodid(i,"FUEL-TYPE")==12)||
        (P.Vcodid(i,"HEAT-GEN-TYPE")==7))
       {
        ostringstream res; res << "Natural Gas";
        KV.Reset(); KV.Init(4,res.str());
        Temp.Prop.insert(make pair("FUEL-TYPE",KV));
       }
    // if the hot water system is served by an HVAC system whose heat
source is a heat pump, then use natural gas instead
    else     if     (     (P.Vcodid(i,"HEAT-GEN-TYPE")==1)     &&     (
(P.Vcodid(i,"HEAT-SOURCE")==12)||(P.Vcodid(i,"HEAT-
SOURCE")==13)||(P.Vcodid(i,"HEAT-SOURCE")==18)||(P.Vcodid(i,"HEAT-
SOURCE")==19) ) )
           {
            ostringstream res; res << "Natural Gas";
            KV.Reset(); KV.Init(4,res.str());
            Temp.Prop.insert(make pair("FUEL-TYPE",KV));
           }
         else  // use the same fuel as in the actual building
           {
            ostringstream res; res << P.Vcod(i,"FUEL-TYPE");
            KV.Reset(); KV.Init(4,res.str());
            Temp.Prop.insert(make pair("FUEL-TYPE",KV));
           }
  }
  else if((P.Vcodid(cId,"BR-TYPE")==6)) { // Jersey
        //if elec is used in the actual building, set notional fuel
to elec, otherwise use LPG
        if (P.Vcodid(i,"FUEL-TYPE")==10)
           {
            ostringstream res; res << P.Vcod(i,"FUEL-TYPE");
```

```
            KV.Reset(); KV.Init(4,res.str());
            Temp.Prop.insert(make pair("FUEL-TYPE",KV));
           }
         else
           {
            ostringstream res; res << "LPG";
            KV.Reset(); KV.Init(4,res.str());
            Temp.Prop.insert(make_pair("FUEL-TYPE",KV));
           }
        }
  else { // all UK
    // Check if this is the 2010 notional
    if (P.Vint(gId,"NOT-CASE")>=9)  { // 9=EW NOT 2010

      // Check if the hot water is provided by an electric heat pump
      // 7=Heat pump , 10=Grid Supplied Electricity
      if  (  (P.Vcodid(i,"HEAT-GEN-TYPE")==7)  &&  (P.Vcodid(i,"FUEL-
TYPE")==10) )
        mNot Fuel=P.Vcod(i,"FUEL-TYPE");   // same as heating fuel in
the actual building (grid electricity)

      else { // i.e., not an independent electric heat pump
        // Check if the hot water is provided by an HVAC system
through an electric heat pump
        // 1=Using Central Heating boiler , 18=Heat pump (electric):
air source ,
        // 19=Heat pump (electric): ground or water source , 10=Grid
Supplied Electricity
        if ( (P.Vcodid(i,"HEAT-GEN-TYPE")==1) && ((P.Vcodid(i,"HEAT-
SOURCE")==18) || (P.Vcodid(i,"HEAT-SOURCE")==19))
             && (P.Vcodid(i,"FUEL-TYPE")==10) )
            mNot Fuel=P.Vcod(i,"FUEL-TYPE");     // same as heating
fuel in the actual building (grid electricity)

        else { // i.e., not an electric heat pump
          // Check if the heating fuel in the actual building is one
whose emission factor is >= that of oil,
          // i.e., >= 0.306 kg/kWh
          //  4=Oil,  5=Coal,  6=Anthracite,  7=Smokeless  Fuel  (inc
Coke), 10=Grid Supplied Electricity
          if  (  (P.Vcodid(i,"FUEL-TYPE")==4)  ||  (P.Vcodid(i,"FUEL-
TYPE")==5) || (P.Vcodid(i,"FUEL-TYPE")==6)
               ||  (P.Vcodid(i,"FUEL-TYPE")==7)  ||  (P.Vcodid(i,"FUEL-
TYPE")==10) )
            mNot_Fuel="Oil";

          else // i.e., the heating fuel in the actual building is
one whose emission factor < that of oil,
            // i.e., < 0.306 kg/kWh
            mNot Fuel=P.Vcod(i,"FUEL-TYPE");     //  same  as  heating
fuel in the actual building
        } // end else
      } // end else

      // Save the heating fuel type
      {
        ostringstream res; res << mNot_Fuel;
        KV.Reset(); KV.Init(4,res.str());
        Temp.Prop.insert(make pair("FUEL-TYPE",KV));
      }
    } // end if

    else { // i.e., not 2010 notional
      //if gas is used anywhere in the actual building, set notional
fuel to gas, otherwise use oil
      //keyword   NOT-FUEL   is   set   to   gas   in   funtions
DHW GENERATOR::Init fuel() and HVAC::Init fuel() if there is gas in
the actual building
      if (P.Vcod(gId,"NOT-FUEL")=="GAS")
```

```
          {
           ostringstream res; res << "Natural Gas";
           KV.Reset(); KV.Init(4,res.str());
           Temp.Prop.insert(make pair("FUEL-TYPE",KV));
          }
       else
          {
           ostringstream res; res << "Oil";
           KV.Reset(); KV.Init(4,res.str());
           Temp.Prop.insert(make pair("FUEL-TYPE",KV));
          }
     } // end else
   }


 //notional system is not a storage system
 {
  ostringstream res; res << "NO";
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make_pair("STORE-SYSTEM",KV));
 }
 //HVAC guide no is same as in the actual building
 {
  ostringstream res; res << P.Vnum(i,"HVACGUIDE-DHW");
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("HVACGUIDE-DHW",KV));
 }
 //store actual building DHW heat generator efficiency in keyword
HEAT-GEN-EFF-ACT
 {
  ostringstream res; res << P.Vnum(i,"HEAT-GEN-EFF");
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("HEAT-GEN-EFF-ACT",KV));
 }
 {
  ostringstream res; res << P.Vnum(i,"HEAT-GEN-EFF");
  P.ModKey(i,"HEAT-GEN-EFF-ACT",res.str());
 }
 //Notional DHW heat generator efficiency is from default value
(TABLE) DHW NOT EFF

  //--- Define efficiency
  // Check if this is the 2010 notional
  if (P.Vint(gId,"NOT-CASE")>=9)  { // 9=EW NOT 2010

    double mHEAT GEN EFF=0.0;
    double dHEAT GEN EFF=0.0;    // partial efficiency

    int mFuel Id=FuelTypeId(mNot Fuel);   // CODID of the notional
fuel determined earlier
    deque<string> BH_HS=P.Vltext(i,"BH-HEAT-SOURCE");
    deque<string> BH FT=P.Vltext(i,"BH-FUEL-TYPE");
    deque<string> BH GT=P.Vltext(i,"BH-HEAT-GEN-TYPE");
    deque<double> BH_LOAD=P.Vlnum(i,"BH-LOAD");

    int Nfuel=BH FT.size();     // number of fuel types
    for(int i=0;i<Nfuel;i++)
      dHEAT GEN EFF+=fHEAT GEN EFF BH(D,  T,  P,  BH HS[i],  BH FT[i],
BH GT[i], BH LOAD[i], gId);

   //  The efficiency equivalent to the combination of bivalent
systems
    mHEAT_GEN_EFF=T.Value("tKGCO2",1,mFuel_Id)/dHEAT_GEN_EFF;

    // Save the equivalent efficiency
    {
     ostringstream res; res << mHEAT GEN EFF;
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make_pair("HEAT-GEN-EFF",KV));
```

```
    }

  } // end if

  else // i.e., not 2010 notional building
    {
     ostringstream              res;              res              <<
T.Value("DHW_NOT_EFF",1,P.Vint(gId,"NOT-CASE"));  // JO 24 Feb 09
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make pair("HEAT-GEN-EFF",KV));
    }

  N.vINP.push back(Temp);

}


void DHW GENERATOR::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC&
P, PROJECTTables& HVG) {
  if(func_report) cout << endl << "  " << "DHW_GENERATOR::Calc";
  if(calc report) cout << endl << P.vINP[Id].Name;
//calls functions for DHW calculations


//function to produce DHW HVAC guide system number for efficiency
check in BRUKL (for actual building only)
    if (P.Vcod(gId,"ACT-NOT")=="ACT") fHVAC GUIDE NO(D, T, P, HVG);
//total dhw demand, area and dist losses for DHW generator:
    fDEMAND DIST LOSSES(D, T, P);
//calculations for each Solar Energy System (SES) serving this DHW
generator:
  for (int i=0;i<vS.size();i++)
    vS[i].Calc(D, T, P);
//storage losses: also calculates the storage volume if not input
    fSTOR LOSS(D, T, P);
// net dhw demand after contributions from SES and CHP
    fTOTAL VALUES(D, T, P);
//secondary circulation losses:
    fSEC_CIRC(D, T, P);
//distribution efficiency:
    fDIST EFF(D, T, P);
//secondary circulation pump energy:
    fSEC CIRC PUMP(D, T, P);
//set heat generator efficiency (actual building only):
if (P.Vcod(gId,"ACT-NOT")=="ACT")
    fHEAT GEN EFF(D, T, P);
//DHW energy consumption:
    fDHW_ENERGY(D, T, P);
}

void DHW_GENERATOR::fHVAC_GUIDE_NO(PROJECTDefaults& D, PROJECTTables&
T, BOC& P, PROJECTTables& HVG) {
  if(func report)    cout    <<    endl    <<    "         "    <<
"DHW_GENERATOR::fHVAC_GUIDE_NO";
//writes keywords to represent the HVAC guide numbers for the system,
this is passed to BRUKL
//HVACGUIDE_DHW (DHW) is the HVAC guide DHW system number:

  double mHVACGUIDE DHW=-5555;

//find HVAC guide DHW system number
//set to -1111 if DHW uses HVAC heat generator
//will be -5555 if system is not in the HVAC guide, or there is no
min eff defined in HVAC guide

  if (P.Vcodid(Id,"HEAT-GEN-TYPE")!=1) { //DHW does not use HVAC heat
generator
    //need to check DHW-GENERATOR keyword HEAT-GEN-TYPE and DHW-
GENERATOR keyword FUEL-TYPE
```

```
    int mHEAT GEN TYPE=P.Vcodid(Id,"HEAT-GEN-TYPE");
    int mFUEL TYPE=P.Vcodid(Id,"FUEL-TYPE");

//use heat gen type and fuel type IDs to create unique table name:
//table name is prefixed "tHVACGUIDE_DHW":
    string HVACGUIDE DHW name="tHVACGUIDE DHW";
    char HVACGUIDE DHW name char[20];
    char *line = "_";

//add system type Id to table name:
    sprintf(HVACGUIDE_DHW_name_char,"%s%d","",mHEAT_GEN_TYPE);
    { string s(HVACGUIDE DHW name char); HVACGUIDE DHW name+=s; }
//add underscore then heat source Id to table name:
    sprintf(HVACGUIDE_DHW_name_char,"%s%d",line,mFUEL_TYPE);
    { string s(HVACGUIDE DHW name char); HVACGUIDE DHW name+=s; }

//you now have the complete table name to look up in the HVACdefault
tables:
//the HVAC guide number is in the 2nd column (the first column
represents whether additional checks
//e.g. chiller type, heat recovery, need to be made in order to
determine the system number,
//you don't need to make any extra checks for DHW so just use 2nd
column directly)

    mHVACGUIDE DHW=HVG.Value(HVACGUIDE DHW name,1,2);

    {
     ostringstream res; res << mHVACGUIDE DHW;
     P.ModKey(Id,"HVACGUIDE-DHW",res.str());
    }
  }
  else {
    mHVACGUIDE DHW=-1111; //DHW uses HVAC heat generator
  }

}


void      DHW GENERATOR::fDEMAND DIST LOSSES(PROJECTDefaults&      D,
PROJECTTables& T, BOC& P) {
  if(func_report)     cout     <<     endl     <<     "          "     <<
"DHW GENERATOR::fDEMAND DIST LOSSES";
 //calculates total demand, total distribution losses and total area
served by the DHW GENERATOR

  //demand for the dhw generator
  double mQ DEM[12]={ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0 };
  double mQ DEM CORR[12]={ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };  // adjusted for 2010 notional top-lit
  //demand for an individual zone
  deque<double> mQ DEM ZONE;
  deque<double>  mQ_DEM_ZONE_CORR;     // adjusted for 2010 notional
top-lit
  //distribution loss for the dhw generator
  double  mDIST LOSS[12]={ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };
  //distribution loss for an individual zone
  deque<double> mDIST_LOSS_ZONE;
  //total area served by dhw generator
  double mAREA = 0.0;
  //hours of operation per month for dhw generator
  double  mHOURS[12]={ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0 };
  //occupation start time for zone
  deque<double> mOCC T START ZONE;
  //occupation end time for zone
  deque<double> mOCC_T_END_ZONE;
```

```
  //occupation start time for dhw generator
  double mOCC T START[12]={ 25.0, 25.0, 25.0, 25.0, 25.0, 25.0, 25.0,
25.0, 25.0, 25.0, 25.0, 25.0 };      //occupation start time for dhw
generator
  //occupation end time for dhw generator
  double mOCC T END[12]={ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };

  map<string, KeywordValue>::iterator it;

  //added
  //used to store the code for the system type:
  int HVACId; //
  //sys-t-hours, sys-t-start and sys-t-end now uses to evaluate no.
of hours/ month DHW generator use
  deque<double> mSYS T HOURS ZONE;
  deque<double> mSYS_T_START_ZONE;
  deque<double> mSYS T END ZONE;
  //the following used to store the 'usage density' for each hour of
day
  //this is the SYS-T-HOURS divided by the hours of use of the system

  //used to store the 'usage density' for a zone for each hour of day
for each month:
  double mUSE DENSITY ZONE[288];
  //used to store the maximum 'usage density' for each hour of day
for each month
  double mMAX USE DENSITY[288];
  //set to zero:
  for (int i=0;i<288;i++) mMAX USE DENSITY[i]=0;
  //end added

  // 18 April 2010 JO: Shell and Core. Ratio for SES contribution
  double mAREA ACTUAL=0.0;
  double mAREA ASSUMED=0.0;

  //find zones which you serve:
  //go through all objects in project
  for (int i=0;i<P.vINP.size();i++) {

    //added - store type of HVAC system, this will be needed below to
determine whether system has cooling
    if (P.vINP[i].Type=="HVAC-SYSTEM") {
      HVACId=i;
    }
    //end added

    //if object is a zone:
    if (P.vINP[i].Type=="ZONE") {
      //if(P.Vcodid(i,"SHELL-ASSUMED")==1)
      if((P.Vcodid(cId,"SHELL-AND-CORE")==1   &&   P.Vcodid(cId,"EPC-
TYPE")==1   &&   P.Vcodid(cId,"BR-STAGE")==2   &&   P.Vcodid(i,"SHELL-
ASSUMED")==1))
        mAREA_ASSUMED+=P.Vnum(i,"AREA");
      else {
        mAREA ACTUAL+=P.Vnum(i,"AREA");

        //find which DHW-GENERATOR it is served by:
        it=P.vINP[i].Prop.find("DHW-GENERATOR");
        //if it is this DHW generator:
        if (it->second.REFId==Id) {
          //demand for the zone:
          mQ_DEM_ZONE=P.Vlnum_Mul(i,"Q-DHW",P.Vnum(i,"MULTIPLIER"));
          mQ DEM ZONE CORR=P.Vlnum Mul(i,"Q-DHW-
CORR",P.Vnum(i,"MULTIPLIER"));    // adjusted for 2010 notional top-
lit
          //distribution loss for the zone :
          mDIST LOSS ZONE=P.Vlnum Mul(i,"DHW-DIST-
LOSS",P.Vnum(i,"MULTIPLIER"));
```

```
          //SYS-T-HOURS, SYS-T-START: values can be 0 to 23
          if (T.Value("tSYS COOL",1,P.Vcodid(HVACId,"TYPE"))==1) //if
system has cooling: use sys-t-hours
          {
            mSYS T HOURS ZONE=P.Vlnum(i,"SYS-T-HOURS");
            mSYS T START ZONE=P.Vlnum(i,"SYS-T-START");
            mSYS_T_END_ZONE=P.Vlnum(i,"SYS-T-END");
          }
          else
          {
            mSYS T HOURS ZONE=P.Vlnum(i,"SYS-HEAT-T-HOURS");      //if
system has no cooling: use sys-heat-t-hours
            mSYS_T_START_ZONE=P.Vlnum(i,"SYS-HEAT-T-START");
            mSYS T END ZONE=P.Vlnum(i,"SYS-HEAT-T-END");
          }
          //enter values in use density vector for zone:
          //first set every hour to zero:
          for (int j=0;j<288;j++) mUSE DENSITY ZONE[j]=0;
          //for each month:
          for (int j=0;j<12;j++) {
            //for each hour of day:
            for (int k=0;k<24;k++) {
              if             ((k>=mSYS T START ZONE[j])             &&
(k<=mSYS T END ZONE[j])) {
                if (mSYS T END ZONE[j]!=mSYS T START ZONE[j])

mUSE DENSITY ZONE[j*24+k]=mSYS T HOURS ZONE[j]/(mSYS T END ZONE[j]-
mSYS T START ZONE[j]);
              }
            }
          }

          //store largest value of mUSE DENSITY ZONE for each hour of
each month in mMAX USE DENSITY
          //for each month:
          for (int j=0;j<12;j++) {
            //for each hour of day:
            for (int k=0;k<24;k++)
              if  (mMAX USE DENSITY[j*24+k]<mUSE DENSITY ZONE[j*24+k])
mMAX USE DENSITY[j*24+k]=mUSE DENSITY ZONE[j*24+k];
          }
          //end added



          //calculate total area served for the DHW GENERATOR:
          mAREA+=P.Vnum(i,"AREA")*P.Vnum(i,"MULTIPLIER");

          //for each month:
          for (int j=0;j<12;j++) {
            //add demand for zone to demand for the DHW GENERATOR:
            mQ DEM[j]+=mQ DEM ZONE[j]; //Demand in MJ
            mQ DEM CORR[j]+=mQ DEM ZONE CORR[j];  //Demand  in  MJ  ,
adjusted for 2010 notional top-lit
            //add distribution losses for zone to distribution losses
for DHW GENERATOR:
            mDIST LOSS[j]+=mDIST LOSS ZONE[j];
/*                 //find earliest start time and latest end time -
whenever any zone requires DHW, DHW generator is operating
            //if  zone  start  time  is  earlier  than  DHW  start  time,
change DHW start time to the earlier time:
            if                  (mOCC_T_START_ZONE[j]<mOCC_T_START[j])
mOCC T START[j]=mOCC T START ZONE[j];
            //if zone end time is later than DHW end time, change DHW
end time to the later time:
            if                        (mOCC T END ZONE[j]>mOCC T END[j])
mOCC T END[j]=mOCC T END ZONE[j];
*/
```

```
          }
        }
      }
    }
  }

  //added - calculate hours of operation per month for dhw generator,
mHOURS

  //for each month:
  if (P.Vcodid(Id,"SEC-CIRC-TS")==1) { //DHW system has time switch
on secondary circulation system
    for (int j=0;j<12;j++) {
      //for each hour of day:
      for (int k=0;k<24;k++) {
        mHOURS[j]+=mMAX USE DENSITY[j*24+k];
      }
      if                              (mHOURS[j]>24*T.Value("NJ",1,j+1))
mHOURS[j]=24*T.Value("NJ",1,j+1);
    }
  }
  else { //DHW system does not have time switch on secondary
circulation system
    for (int k=0;k<12;k++) {
      mHOURS[k]=24*T.Value("NJ",1,k+1);
    }
  }
  //end added

  //store gross (total) demand including the portions that might be
provided by SES or CHP
  {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ DEM[i] << ", ";
  res << mQ DEM[11] << " }";
  P.ModKey(Id,"Q-DEM-TOT",res.str());
  }
  {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ_DEM_CORR[i] << ", ";
  res << mQ DEM CORR[11] << " }";
  P.ModKey(Id,"Q-DEM-TOT-CORR",res.str());
  }

  //store distribution losses:
  {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mDIST_LOSS[i] << ", ";
  res << mDIST LOSS[11] << " }";
  P.ModKey(Id,"DIST-LOSS",res.str());
  }
  //store hours of operation per month
  {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mHOURS[i] << ", ";
  res << mHOURS[11] << " }";
  P.ModKey(Id,"HOURS",res.str());
  }
  //store area:
  {
  ostringstream res; res << mAREA;
  P.ModKey(Id,"AREA-SERVED",res.str());
  }

  // 18 April 2010 JO: Shell and core. Save ratio
```

```
  {
  double mSC ACTUAL RATIO=1.0;
  if(mAREA ACTUAL+mAREA ASSUMED>0)
mSC ACTUAL RATIO=mAREA ACTUAL/(mAREA ACTUAL+mAREA ASSUMED);
  ostringstream res; res << mSC_ACTUAL_RATIO;
  P.ModKey(Id,"SC-ACTUAL-RATIO",res.str());
  }

} // end fDEMAND DIST LOSSES


void DHW GENERATOR::fTOTAL VALUES(PROJECTDefaults& D, PROJECTTables&
T, BOC& P) {
  if(func report)    cout    <<    endl    <<    "         "    <<
"DHW GENERATOR::fTOTAL VALUES";
  //calculates net demand from HW generator, taking into account
contributions from SES and CHP

  //gross demand for the hot water before contributions from SES or
CHP are deducted
  deque<double> mQ DEM=P.Vlnum(Id,"Q-DEM-TOT");

  deque<double> mQ SES;
  double mQ SES DHW[12]={ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };

  for (int i=0;i<vS.size();i++) {
    mQ SES=P.Vlnum Mul(vS[i].Id,"Q-SES-HW-
M",P.Vnum(vS[i].Id,"MULTIPLIER"));
    for (int j=0;j<12;j++)
      mQ SES DHW[j]+=mQ SES[j];
  }
  for (int i=0;i<12;i++) {
    if (mQ SES DHW[i]>1.0*mQ DEM[i])
      mQ SES DHW[i]=1.0*mQ DEM[i];
  }

  //demand = demand for dhw-SES contribution:
  for (int i=0;i<12;i++)
    mQ DEM[i]-=mQ SES DHW[i];

 double mQ DEM CHP[12]= { 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };
 double mQ PRIM CHP[12]= { 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };
 if (P.Vcod(Id,"CHP")=="YES") {
    double  mBUILD CHP DHEAT=P.Vnum(P.Vint(gId,"CHP-POS"),"BUILD-CHP-
DHEAT");
    double mCHP HEAT SEER=P.Vnum(P.Vint(gId,"CHP-POS"),"HEAT-SEER");
    deque<double>  mKWH CHP M=P.Vlnum(P.Vint(gId,"CHP-POS"),"KWH-DHW-
M");
    double mKWH CHP=0;

    if(mCHP_HEAT_SEER!=0) {
      for (int m=0;m<12;m++) {
          mQ DEM CHP[m]=mBUILD CHP DHEAT*mQ DEM[m];
          mQ_PRIM_CHP[m]=mQ_DEM_CHP[m]/mCHP_HEAT_SEER;
          mKWH CHP M[m]+=mQ_PRIM_CHP[m]/3.6;
          mKWH CHP+=mKWH CHP M[m];
          mQ_DEM[m]*=(1-mBUILD_CHP_DHEAT);
      }
      {
       ostringstream res; res << "{ ";
       for (int i=0;i<11;i++)
         res << mKWH CHP M[i] << ", ";
       res << mKWH_CHP_M[11] << " }";
       P.ModKey(P.Vint(gId,"CHP-POS"),"KWH-DHW-M",res.str());
      }
```

```
      {
       ostringstream res; res << mKWH CHP;
       P.ModKey(P.Vint(gId,"CHP-POS"),"KWH-DHW",res.str());
      }
    }
 }

  //store net demand from HW generator after contributions from SES
or CHP have been deducted
  {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ DEM[i] << ", ";
  res << mQ_DEM[11] << " }";
  P.ModKey(Id,"Q-DEM",res.str());
  }
  {// Save HW demand portion provided by CHP
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ_DEM_CHP[i] << ", ";
  res << mQ DEM CHP[11] << " }";
  P.ModKey(Id,"Q-CHP-DHW",res.str());
  }
  { // Save HW demand portion provided by SES
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ SES DHW[i] << ", ";
  res << mQ SES DHW[11] << " }";
  P.ModKey(Id,"Q-SES-DHW",res.str());
  }

} // end fTOTAL VALUES



void DHW GENERATOR::fSTOR LOSS(PROJECTDefaults& D, PROJECTTables& T,
BOC& P) {
  if(func report)    cout    <<    endl    <<    "         "    <<
"DHW_GENERATOR::fSTOR_LOSS";
//calculate storage losses for this DHW generator

  //if it is a storage system
  if(P.Vcodid(Id,"STORE-SYSTEM")==1) { // 1=yes
    double                      mAREA SERVED=P.Vnum(Id,"AREA-SERVED");
//total area served by this dhw generator
    double                           mSTORE VOL=P.Vnum(Id,"STORE-VOL");
//storage volume for dhw generator
    double mSTORE_LOSS=P.Vnum(Id,"STORE-LOSS");                    //
monthly DHW storage losses in MJ / month
    //the  following  2  variables  are  used  to  assess  the  DHW
recommendations in function PROJECT::DHG_Recommendation(...)
    double mSTORE VOL DEFAULT=0;                      //default storage
volume for dhw generator
    double mSTORE LOSS DEFAULT=0;                     //default monthly
DHW storage losses in MJ / month
    deque<double> mMonth Demand;       //monthly dhw demand
    double mDay_Demand;                // daily DHW demand (MJ/day)
    double mDay MaxDemand;                // maximum daily DHW demand
(MJ/day)
    double mSTORE VOL5;                 // storage volume (in litres)
when the annual DHW demand is 5 MJ/m2

    double  mSTORE_SES=P.Vnum(Id,"STORE-VOL-SES");     // total solar
storage of all connected SES, in litres

    // If the storage volume has been input by the user but it is
smaller than the total solar storage volume
    // required for all the connected SES, then increase the HW
storage volume accordingly (assume solar volume ratio=2/3)
```

```
    if( (mSTORE VOL!=-6666) && (mSTORE SES>mSTORE VOL) ){

      mSTORE VOL=mSTORE SES/D.Value("dSES VOL RATIO");
      { // display a warning in the log file
       ostringstream logtext;
       logtext << endl << "WARNING: Storage for HW system " <<
P.vINP[Id].Name << " was increased to accommodate the required solar
storage for the connected SES." << endl;
       SendLog(logtext.str()); WriteLog();
      }
      {
      ostringstream res; res << mSTORE VOL;
      P.ModKey(Id,"STORE-VOL",res.str());
      }
    } // end if


//-----calculate default storage volume (in litres)-----

    //find max value of daily demand for the DHW GENERATOR:
    mMonth Demand=P.Vlnum(Id,"Q-DEM-TOT");
    //initialise to value for 1st month (i.e., January with 31 days):
    mDay MaxDemand=mMonth Demand[0]/31;
    //put highest daily demand in mMax Demand:
    for (int i=1;i<12;i++) {
        mDay Demand=mMonth Demand[i]/T.Value("NJ",1,i+1);
      if (mDay Demand>mDay MaxDemand)
         mDay MaxDemand=mDay Demand;    //  maximum  daily  DHW  demand
(MJ/day)
    }

    // The value in Default DHW 3 has been changed to 18 (litres) -
defined as 18 litres per MJ of daily demand) - It used
    //  to  be  36  and  was  modified  as  per  advice  from  RH  -  AN
(03/11/2009)
    mSTORE VOL DEFAULT=mDay MaxDemand*D.Value("DHW 3");             //
litres

    // If the default storage volume is smaller than the total solar
storage volume
    //  required  for  all  the  connected  SES,  then  increase  the  HW
storage volume accordingly (assume solar volume=2/3)
    if(mSTORE SES>mSTORE VOL DEFAULT)
      mSTORE VOL DEFAULT=mSTORE SES/D.Value("dSES VOL RATIO");

    //store default volume for recommendations only:
    {
      ostringstream res; res << mSTORE_VOL_DEFAULT;
      P.ModKey(Id,"STORE-VOL-DEFAULT",res.str());
    }

    //if  user  has  not  entered  storage  volume  (in  litres),  and  if
monthly storage losses (in MJ/month) were not given by
    //the user either, then store default storage to be used both for
actual building calculation and recommendations
    if ((mSTORE VOL==-6666)&&(mSTORE LOSS==-6666)) {
      mSTORE_VOL=mSTORE_VOL_DEFAULT;
      {
      ostringstream res; res << mSTORE VOL;
      P.ModKey(Id,"STORE-VOL",res.str());
      }
    }

//-----calculate default storage losses (in MJ/month)-----

    // Since the storage losses are defined (in RH's spreadsheet) as
0.1425 kWh/litre.day (DHW 4), then we multiply by 365 to
    // get the annual losses and divide by 12 to get the monthly
losses. We also muliply by 3.6 to convert the kWh to MJ.
```

```
    // DHW 10 is a new default containing DHW storage volume in
litres/m2 when the annual DHW demand is 5 MJ/m2,
    // calculated as: daily DHW demand 5/365 * 18 litres per MJ of
daily demand
    mSTORE VOL5=D.Value("DHW 10")*mAREA SERVED;  // as line above but
multiplied by area to get value in litres
    double mpwr1=1.0/3.0;
    double mpwr2=2.0/3.0;

    if (mSTORE VOL DEFAULT!=-1.0) {

mSTORE LOSS DEFAULT=D.Value("DHW 4")*(pow(mSTORE VOL5,mpwr1))*(365.0/
12.0)*(pow(mSTORE VOL DEFAULT,mpwr2))*3.6;     // MJ/month
    }
    else
      mSTORE LOSS DEFAULT=-1.0;
    //in any system where there is storage, store the default storage
loss for use in recommendations report
    //default storage loss value uses default storage volume
    {
    ostringstream res; res << mSTORE LOSS DEFAULT;
    P.ModKey(Id,"STORE-LOSS-DEFAULT",res.str());
    }

//-----calculate storage losses (in MJ/month) if not given by user:
    if (mSTORE LOSS==-6666) {

      if (mSTORE VOL==-1.0)
        mSTORE LOSS=-1.0;
      else {
        double  mINSUL THICK=P.Vnum(Id,"STORE-INSULAT-THICK");      //
insulation thickness, in mm, on storage vessel
        double  mKWH LOSS DAY=0.0;       //  daily  storage  losses  in
kWh/litre of storage

        // Ensure user has not input a negative value for insulation
thickness
        if (mINSUL THICK<0)
          mINSUL_THICK=0;

        switch(P.Vcodid(Id,"STORE-INSULAT-TYPE")) {
          case 1: { //Uninsulated
                    mKWH LOSS DAY=D.Value("DHW 4");          //  daily
storage losses in kWh/litre of storage
                  }
                  break;
          case 2: { //Loose jacket

mKWH LOSS DAY=D.Value("DHW 11")+(D.Value("DHW 12")/(mINSUL THICK+D.Va
lue("DHW 13")));     // daily storage losses in kWh/litre of storage
                  }
                  break;
          case 3: { //Factory insulated

mKWH LOSS DAY=D.Value("DHW 11")+(D.Value("DHW 14")/(mINSUL THICK+D.Va
lue("DHW 15")));     // daily storage losses in kWh/litre of storage
                  }
                  break;
        } // end switch

        // Monthly storage losses in MJ

mSTORE_LOSS=mKWH_LOSS_DAY*(pow(mSTORE_VOL5,mpwr1))*(365.0/12.0)*(pow(
mSTORE VOL,mpwr2))*3.6;     // MJ/month

        { // Save daily storage loss to display in BRUKL document
        ostringstream res; res << mKWH LOSS DAY;
        P.ModKey(Id,"STORE-LOSS-DAY",res.str());    //  daily  storage
losses in kWh/litre of storage
```

```
        }

      } // end else

      {
      ostringstream res; res << mSTORE LOSS;
      P.ModKey(Id,"STORE-LOSS",res.str());   // MJ/month
      }
    } // end if
    else { //i.e., user has entered a value for the monthly losses so
we check it is valid and produce a warning if it's not
      double mLOSS DAY;       // daily loss in kWh/litre of storage
corresponding to claimed monthly losses from the user

mLOSS DAY=mSTORE LOSS/((pow(mSTORE VOL5,mpwr1))*(365.0/12.0)*(pow(mST
ORE VOL DEFAULT,mpwr2))*3.6); // kWh/day per litre of storage
      if (mLOSS DAY<D.Value("dMIN DHW LOSS")) // assuming a maximum
possible insulation thickness of 160mm
      { ostringstream logtext;
        logtext << endl << "WARNING: Check the input monthly storage
losses for HW system: " << P.vINP[Id].Name << " as they would
correspond to an unusual level of insulation." << endl;
        SendLog(logtext.str()); WriteLog();
      }
    } // end else
  } // end if
  else { // not a storage system
    {
    ostringstream res; res << 0.0;
    P.ModKey(Id,"STORE-VOL",res.str());
    }
    {
    ostringstream res; res << 0.0;
    P.ModKey(Id,"STORE-LOSS",res.str());
    }
  } // end else

}


void DHW GENERATOR::fSEC CIRC(PROJECTDefaults& D, PROJECTTables& T,
BOC& P) {
  if(func report) cout << endl << "  " << "DHW GENERATOR::fSEC CIRC";
//calculate secondary circulation loss for this DHW generator

  //if there is secondary circulation:
  if(P.Vcodid(Id,"SEC-CIRCULATION")==1) {
    double mSEC_LOOP_LENGTH=P.Vnum(Id,"SEC-LOOP-LENGTH");

    //calculate secondary loop length if not given by user:
    if (mSEC_LOOP_LENGTH==-6666) {
      mSEC LOOP LENGTH=sqrt(P.Vnum(Id,"AREA-
SERVED"))*D.Value("DHW 5");

      {
       ostringstream res; res << mSEC LOOP LENGTH;
       P.ModKey(Id,"SEC-LOOP-LENGTH",res.str());
      }
    }

      //calculate secondary circulation losses per metre if not given
by user:
      double mSEC_CIRC_LOSS_MTR=P.Vnum(Id,"SEC-CIRC-LOSS/M");

      if (mSEC CIRC LOSS MTR==-6666) {
        mSEC_CIRC_LOSS_MTR=D.Value("DHW_6");
        {
         ostringstream res; res << mSEC CIRC LOSS MTR;
         P.ModKey(Id,"SEC-CIRC-LOSS/M",res.str());
```

```
        }
      }

    //calculate secondary circulation losses per month:
    double mSEC_CIRC_LOSS_MON[12];
    deque<double> mHOURS=P.Vlnum(Id,"HOURS");

    for (int i=0;i<12;i++)

mSEC CIRC LOSS MON[i]=mSEC CIRC LOSS MTR*mSEC LOOP LENGTH*mHOURS[i]*(
3.6/1000);

    {
     ostringstream res; res << "{ ";
     for(int i=0;i<11;i++)
       res << mSEC CIRC LOSS MON[i] << ", ";
     res << mSEC_CIRC_LOSS_MON[11] << " }";
     P.ModKey(Id,"SEC-CIRC-LOSS",res.str());
    }
  }
  else {
    {
     ostringstream res; res << 0.0;
     P.ModKey(Id,"SEC-LOOP-LENGTH",res.str());
    }
    {
     ostringstream res; res << 0.0;
     P.ModKey(Id,"SEC-CIRC-LOSS/M",res.str());
    }
    {
     ostringstream res; res << "{ ";
     for(int i=0;i<11;i++)
       res << 0.0 << ", ";
     res << 0.0 << " }";
     P.ModKey(Id,"SEC-CIRC-LOSS",res.str()); //MJ/day !!
    }
  }
}


void  DHW GENERATOR::fDIST EFF(PROJECTDefaults&  D,  PROJECTTables&  T,
BOC& P) {
  if(func report) cout << endl << "  " << "DHW GENERATOR::fDIST EFF";
//calculate distribution efficiency for DHW generator per month

  double mDIST EFF[12];
  deque<double> mQ DEM=P.Vlnum(Id,"Q-DEM-TOT");
  deque<double> mDIST_LOSS=P.Vlnum(Id,"DIST-LOSS");
  double mSTORE LOSS=P.Vnum(Id,"STORE-LOSS");
  deque<double> mSEC CIRC LOSS=P.Vlnum(Id,"SEC-CIRC-LOSS");

  //for the notional building, distribution loss, storage loss and
secondary circulation loss are all zero
  for (int i=0;i<12;i++) {
    if   (((mQ DEM[i]+mDIST LOSS[i]+mSTORE LOSS+mSEC CIRC LOSS[i])!=0)
&& (mSTORE LOSS!=-1.0))
      mDIST_EFF[i]=
mQ DEM[i]/(mQ DEM[i]+mDIST LOSS[i]+mSTORE LOSS+mSEC CIRC LOSS[i]);
    else
      mDIST_EFF[i]=-1.0;
  }

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mDIST_EFF[i] << ", ";
   res << mDIST EFF[11] << " }";
   P.ModKey(Id,"DIST-EFF",res.str());
  }
```

```
}


void DHW GENERATOR::fSEC CIRC PUMP(PROJECTDefaults& D, PROJECTTables&
T, BOC& P) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"DHW GENERATOR::fSEC CIRC PUMP";
//calculate secondary circulation pump energy for this DHW generator

  //if there is secondary circulation:
  if(P.Vcodid(Id,"SEC-CIRCULATION")==1) {
    //calculate circulation pump power if not given by user:
    double mSEC PUMP POWER=P.Vnum(Id,"SEC-PUMP-POWER");

    if (mSEC PUMP POWER==-6666) {
      if (D.Value("DHW 9")!=0.0)
        mSEC_PUMP_POWER=(D.Value("DHW_7")*P.Vnum(Id,"SEC-LOOP-
LENGTH")+D.Value("DHW 8"))/D.Value("DHW 9");
      else
        mSEC_PUMP_POWER=-1.0;
      {
       ostringstream res; res << mSEC PUMP POWER;
       P.ModKey(Id,"SEC-PUMP-POWER",res.str());
      }
    }

    //calculate circulation pump energy per month:
    double mSEC PUMP ENERGY[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
};
    deque<double> mHOURS=P.Vlnum(Id,"HOURS");

    if  (mSEC PUMP POWER!=-1.0)
      for                    (int                    i=0;i<12;i++)
mSEC PUMP ENERGY[i]=mSEC PUMP POWER*mHOURS[i];
    else
      for (int i=0;i<12;i++)   mSEC PUMP ENERGY[i]=-1.0;

    {
     ostringstream res; res << "{ ";
     for (int i=0;i<11;i++)
       res << mSEC PUMP ENERGY[i] << ", ";
     res << mSEC_PUMP_ENERGY[11] << " }";
     P.ModKey(Id,"SEC-PUMP-ENERGY",res.str());  //kWh
    }
  }
  else {
    {
     ostringstream res; res << 0.0;
     P.ModKey(Id,"SEC-PUMP-POWER",res.str());
    }
    {
     ostringstream res; res << "{ ";
     for (int i=0;i<11;i++)
       res << 0.0 << ", ";
     res << 0.0 << " }";
     P.ModKey(Id,"SEC-PUMP-ENERGY",res.str());
    }
  }
}


void  DHW GENERATOR::fHEAT GEN EFF(PROJECTDefaults&  D,  PROJECTTables&
T, BOC& P) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"DHW GENERATOR::fHEAT GEN EFF";
//set heat generator efficiency (actual building only):

  if (P.Vcodid(Id,"HEAT-GEN-TYPE")==1)
    mHEAT_GEN_EFF=P.Vnum(Id,"HVAC-SYSTEM-
```

```
EFF")*D.Value("DHW HVAC SYS RED");
  else
    mHEAT GEN EFF=P.Vnum(Id,"DHW-GEN-SEFF");
  {
   ostringstream res; res << mHEAT_GEN_EFF;
   P.ModKey(Id,"HEAT-GEN-EFF",res.str());
  }
}


void DHW GENERATOR::fDHW ENERGY(PROJECTDefaults& D, PROJECTTables& T,
BOC& P) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"DHW_GENERATOR::fDHW_ENERGY";
//calculate DHW energy consumption for this DHW generator

  //calculate overall thermal efficiency per month:
  double mTHERM EFF[12];
  deque<double> mDIST EFF=P.Vlnum(Id,"DIST-EFF");
  double mHEAT_GEN_EFF=P.Vnum(Id,"HEAT-GEN-EFF");

  //calculate DHW energy consumption per month:
  double mDHW ENERGY[12];
  double mQ PRIM=0;
//   deque<double> mQ DEM TOT=P.Vlnum(Id,"Q-DEM-TOT");    // Total HW
demand of building
  deque<double>  mQ DEM TOT CORR=P.Vlnum(Id,"Q-DEM-TOT-CORR");      //
Total HW demand of building, adjusted for 2010 notional top-lit
  deque<double> mQ SES DHW=P.Vlnum(Id,"Q-SES-DHW");    // portion of
HW demand provided by SES
  deque<double> mQ CHP DHW=P.Vlnum(Id,"Q-CHP-DHW");    // portion of
HW demand provided by CHP

  // Note that to calculate the DHW energy, the total demand is
divided by the distribution efficiency and then the
  // the result net of the contributions from SES and CHP is divided
by the generator efficiency.
  for (int i=0;i<12;i++)
   if  (  (mDIST_EFF[i]  !=  0.0)  &&  (mDIST_EFF[i]  !=  -1.0)  &&
(mHEAT GEN EFF != 0.0) && (mHEAT GEN EFF != -1.0) ) {
     mDHW ENERGY[i]=((mQ DEM TOT CORR[i]/mDIST EFF[i])-
(mQ_SES_DHW[i]+mQ_CHP_DHW[i]))/mHEAT_GEN_EFF;
     mQ PRIM+=mDHW ENERGY[i];
   }
   else {
     mDHW ENERGY[i]=0;
   }

// Applying the correction for electric power factor for DHW is now
done here on calculated energy
if ((P.Vcodid(Id,"FUEL-TYPE")==10)) { // If the fuel of the DHW is
grid-electricity and this is the actual building
   for(int m=0;m<12;m++)
     mDHW_ENERGY[m]*=(1-T.Value("tPOWER_FACTOR",1,P.Vcodid(gId,"ELEC-
POWER-FACTOR")));
   mQ PRIM*=(1-T.Value("tPOWER FACTOR",1,P.Vcodid(gId,"ELEC-POWER-
FACTOR")));
 }

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mDHW_ENERGY[i] << ", ";
   res << mDHW ENERGY[11] << " }";
   P.ModKey(Id,"Q-PRIM-M",res.str());
  }
  {
   ostringstream res; res << mQ PRIM;
   P.ModKey(Id,"Q-PRIM",res.str());
```

```
   }
   {
    ostringstream res; res << "{ ";
    for (int i=0;i<11;i++)
      res << mDHW_ENERGY[i]/3.6 << ", ";
    res << mDHW_ENERGY[11]/3.6 << " }";
    P.ModKey(Id,"KWH-DHW-M",res.str());
   }
   {
    ostringstream res; res << mQ_PRIM/3.6;
    P.ModKey(Id,"KWH-DHW",res.str());
   }

  //write KG CO2 for DHW
 if((P.Vcodid(cId,"BR-TYPE")==5)   ||   (P.Vcodid(cId,"EPC-TYPE")==6))
{
      ostringstream              res;              res              <<
mQ_PRIM/3.6*T.Value("tKGCO2_IR",1,P.Vcodid(Id,"FUEL-TYPE"));
      P.ModKey(Id,"KG-CO2-DHW",res.str());
     }
 else if((P.Vcodid(cId,"BR-TYPE")==6))  //Jersey
     {
      ostringstream              res;              res              <<
mQ_PRIM/3.6*T.Value("tKGCO2_JER",1,P.Vcodid(Id,"FUEL-TYPE"));
      P.ModKey(Id,"KG-CO2-DHW",res.str());
     }
 else // all UK
     {
      ostringstream              res;              res              <<
mQ_PRIM/3.6*T.Value("tKGCO2",1,P.Vcodid(Id,"FUEL-TYPE"));
      P.ModKey(Id,"KG-CO2-DHW",res.str());
     }
}


void DHW_GENERATOR::Recommendation(PROJECTDefaults& D, PROJECTTables&
T, BOC& P, DATA_REC& TNR) {
  if(func_report)     cout     <<     endl     <<     "          "     <<
"DHW_GENERATOR::Recommendation";
/*
  EPC recommendations function for DHW generator

  This function:
     - assesses energy and CO2 performance of this DHW system (GOOD,
FAIR or POOR)
     - assesses the impact of recommendations for DHW

   Recommendations for DHW:
      EPC-W1 - Install more efficient water heater
        -to assess impact, calculate modified energy and CO2 as ratio
between actual efficiency and 0.8
      EPC-W2 - Consider point of use system
        -to assess impact, calculate modified energy and CO2 as ratio
between actual efficiency and 0.75
      EPC-W3 - DHW storage insulation
        -to assess impact, calculate reduction in DHW energy and CO2
as 50% of storage losses
      EPC-W4 - DHW secondary circulation time control
        -to assess impact, calculate reduction in DHW energy and CO2
as 30% of total DHW energy

*/


  for (int i=0; i<vS.size();i++)
    vS[i].Recommendation(D,T,P);

  //DHW Recommendations
```

```
  //indicates whether DHW efficiency recommendation is triggered
  //1-> recommendation is triggered, 0->recommendation is not
triggered
  bool EPCW1=0;
  //indicates whether DHW recommendation on replacing system with
point of use system is triggered
  bool EPCW2=0;
  //indicates whether DHW storage recommendation is triggered
  bool EPCW3=0;
  //indicates whether DHW secondary circulation time switch
recommendation is triggered
  bool EPCW4=0;

  double mHEAT_GEN_EFF=P.Vnum(Id,"HEAT-GEN-EFF");

  if (P.Vcodid(Id,"HEAT-GEN-TYPE")>1) {            // i.e., if DHW is
NOT provided by HVAC system

  //depending on DHW-GENERATOR efficiency set keyword REC-CODE to
GOOD, FAIR or POOR
  //this does not need to be done if the DHW is provided by the HVAC
heat source:

    //REC-CODE can have codid equal to either 1(GOOD), 2(FAIR) or
3(POOR)
    string mREC CODE="POOR";
    //if heat generator efficiency > x, classify heat generator
efficiency as "good":
    if (mHEAT GEN EFF>D.Value("REC DEF1a"))
       mREC CODE="GOOD";
    else {               // i.e., efficiency will be either FAIR or
POOR
         //trigger recommendation EPCW1: more efficient water heater
         EPCW1=1;
         //If x >= heat generator efficiency > y, classify heat
generator efficiency as "fair":
         if (mHEAT GEN EFF>D.Value("REC DEF2"))
            mREC CODE="FAIR";
         else // i.e., heat generator efficiency =< y
              //trigger recommendation EPCW2: replace with point of
use system
              EPCW2=1;
              //heat generator efficiency is "poor" (this is set as
the initial value of mREC CODE above)
    }

    {
     ostringstream res; res << mREC_CODE;
     P.ModKey(Id,"REC-PERFORMANCE",res.str());
    }
  }

    //---- DHW storage ----


  //Check whether there is storage
  if(P.Vcodid(Id,"STORE-SYSTEM")==1) {
     //storage losses for dhw generator
     double mSTORE LOSS=P.Vnum(Id,"STORE-LOSS");
     //default storage losses for dhw generator
     double mSTORE LOSS DEFAULT=P.Vnum(Id,"STORE-LOSS-DEFAULT");
      //If heat loss > default loss * 0.9, trigger storage loss
recommendation EPC-W3
     if (mSTORE LOSS>D.Value("REC DEF12")*mSTORE LOSS DEFAULT) {
     //trigger recommendation EPCW3: improve insulation on DHW
storage
        EPCW3=1;
     }
  }
```

```
  //---- DHW secondary cirulation ----

  //If there is secondary DHW circulation and there is no time
control, trigger recommendation EPCW4
  //Check whether there is secondary cirulation
  if (P.Vcodid(Id,"SEC-CIRCULATION")==1) {
   //check if there is time control, if not trigger recommendation
EPCW4: add time control to sec circulation
    if (P.Vcodid(Id,"SEC-CIRC-TS")==2)
       EPCW4=1;
  }

  int FuelId=P.Vcodid(Id,"FUEL-TYPE");
  double mFUEL CARBON INTENSITY=0.0;
  if((P.Vcodid(cId,"BR-TYPE")==5)  ||  (P.Vcodid(cId,"EPC-TYPE")==6))
//Republic of Ireland only
      mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,FuelId);
  else // all UK
      mFUEL_CARBON_INTENSITY=T.Value("tKGCO2",1,FuelId);

  //price weighting value from table tFuel Price Weighting (depends
on relative prices of fuels)
  //used in calculation of energy impact
  double
Fuel Price Weighting=T.Value("tFuel Price Weighting",1,FuelId);

  //energy and CO2:
  double mQ PRIM=P.Vnum(Id,"Q-PRIM");
  //DHW energy price weighted
  double mQ PRIM PRICE WEIGHTED=mQ PRIM*Fuel Price Weighting;
  double mKG CO2=P.Vnum(Id,"KWH-DHW")*mFUEL CARBON INTENSITY;

  //fuel payback adjustment, used to calculate carbon saved/pound
spent
  double Fuel Payback=T.Value("tFuel Payback",1,FuelId);

  //DHW storage:
  double mSTORE_LOSS_DEFAULT=0;
  double mSTORE LOSS=0;

  double mAREA=P.Vnum(gId,"AREA");

  //used to store adjusted energy for DHW and total after
recommendations are applied:
  double mDHW ENERGY WEIGHTED=0.0;
  double mQ TOTAL ADJUSTED=0.0;

  //variable to store CO2 for DHW and total after recommendation is
applied
  double mDHW_CO2_WEIGHTED=0.0;
  double mCO2 TOTAL ADJUSTED=0.0;

  //total building energy for all end uses (to be used when assessing
impact of recommendation):
  //this is price weighted using factor from table
tFuel_Price_Weighting (depends on relative prices of fuels)
  //used in calculation of energy impact for recommendations
  double mQ TOTAL PRICE WEIGHTED=P.Vnum(gId,"Q-TOTAL-P-WEIGHTED");

  //stores impact of recommendation on CO2 (1=HIGH/2=MEDIUM/3=LOW)
  //set to HIGH initially:
  int IMPACT_CO2=1;
  //stores impact of recommendation on energy (1=HIGH/2=MEDIUM/3=LOW)
  //set to HIGH initially:
  int IMPACT_ENERGY=1;
  //payback period
  double PAYBACK=0;
  //carbon saving per pound
```

```
  int CARBON SAVING POUND=1;

  //total building CO2 for all end uses (to be used when assessing
impact of recommendation)
  //and for calculating the % CO2 for this end use:
  //this keyword is the bulding CO2 if there were no renewables:
  double mKG CO2 PROJECT=P.Vnum(gId,"KG-CO2-EXCL-RENEWABLES");


  //---- recommendation EPC-W1 ----

  //more efficient water heater

  //assess       impact       of       recommendation       EPC-W1:
(1=GOOD/2=FAIR/3=POOR/4=N/A)
  if (EPCW1) {
     if ((D.Value("REC_DEF1"))!=0) {
        //weight energy => energy*(actual eff/D.Value("REC DEF1"))

mDHW_ENERGY_WEIGHTED=mQ_PRIM*(mHEAT_GEN_EFF/D.Value("REC_DEF1"));
        //weight CO2 => CO2/*(actual eff/D.Value("REC DEF1"))

mDHW CO2 WEIGHTED=mKG CO2*(mHEAT GEN EFF/D.Value("REC DEF1"));
     }

     //variable for adjusted Total Building energy
     mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mQ PRIM PRICE WEIGHTED+(mDHW ENERGY WEIGHTED*Fuel Price Weighting);
     //variable for adjusted Total Building CO2
     mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-mKG CO2+mDHW CO2 WEIGHTED;

     //assess energy and CO2 impact
     IMPACT ENERGY=fIMPACT ENERGY(D,          mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
     IMPACT CO2=fIMPACT CO2(D, mKG CO2 PROJECT, mCO2 TOTAL ADJUSTED);

     //calculate  payback  (PAYBACK)  and  Carbon  saving  per  pound
(CARBON SAVING POUND)
     //Multiply default payback by 10 and divide by NOTIONAL building
DHW consumption (kWh/m2.year)
     if((mAREA!=0)&&(TNR.notKWH DHW!=0))
       PAYBACK=D.Value("dEPC_W1")*10.0/(TNR.notKWH_DHW/mAREA);
     else
         PAYBACK=0.0;

     double CARBON SAVING NO=PAYBACK*Fuel Payback;

     //determine CARBON_SAVING_POUND value (GOOD, FAIR, POOR)
     CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

     //write the EPCW1 recommendation keyword
     string EPC XX="EPC-W1";
     Write Rec Keyword(D,  T,  P,  EPC XX,  IMPACT ENERGY,  IMPACT CO2,
PAYBACK, CARBON_SAVING_POUND);
  }

  //---- END recommendation EPC-W1 ----


  //---- recommendation EPC-W2 ----

  //DHW point of user system

  //assess       impact       of       recommendation       EPC-W2:
(1=GOOD/2=FAIR/3=POOR/4=N/A)
  if (EPCW2) {
     if ((D.Value("REC DEF2a"))!=0) {
        //weight energy => energy*(actual eff/D.Value("REC DEF2a"))
```

```
mDHW ENERGY WEIGHTED=mQ PRIM*(mHEAT GEN EFF/D.Value("REC DEF2a"));
        //weight CO2 => CO2/*(actual eff/D.Value("REC DEF2a"))

mDHW CO2 WEIGHTED=mKG CO2*(mHEAT GEN EFF/D.Value("REC DEF2a"));
     }

     //variable for adjusted Total Building energy
     mQ_TOTAL_ADJUSTED=mQ_TOTAL_PRICE_WEIGHTED-
mQ PRIM PRICE WEIGHTED+(mDHW ENERGY WEIGHTED*Fuel Price Weighting);
     //variable for adjusted Total Building CO2
     mCO2_TOTAL_ADJUSTED=mKG_CO2_PROJECT-mKG_CO2+mDHW_CO2_WEIGHTED;

     //assess energy and CO2 impact
     IMPACT ENERGY=fIMPACT ENERGY(D,          mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
     IMPACT CO2=fIMPACT CO2(D, mKG CO2 PROJECT, mCO2 TOTAL ADJUSTED);

     //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
     //Multiply default payback by 10 and divide by NOTIONAL building
DHW consumption (kWh/m2.year)
     if((mAREA!=0)&&(TNR.notKWH DHW!=0))
        PAYBACK=D.Value("dEPC W2")*10.0/(TNR.notKWH DHW/mAREA);
     else
        PAYBACK=0.0;

     double CARBON SAVING NO=PAYBACK*Fuel Payback;

     //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)
     CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

     //write the EPCW2 recommendation keyword
     string EPC XX="EPC-W2";
     Write Rec Keyword(D,  T,  P,  EPC XX,  IMPACT ENERGY,  IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
  }

  //---- END recommendation EPC-W2 ----


  //---- recommendation EPC-W3 ----

  //improve insulation on DHW storage

  //storage losses - look at impact of recommendation EPCW3
  if (EPCW3) {
  //assess impact by reducing DHW energy by 50% of storage losses:
(1=HIGH/2=MEDIUM/3=LOW)
  //Note that storage losses for dhw generator are in MJ/month so is
multiplied by 12 to get annual figure

    //weight           energy           =>           energy-(storage
losses*12.0*D.Value("REC DEF13"))
    mDHW_ENERGY_WEIGHTED=mQ_PRIM-
(mSTORE LOSS*12.0*D.Value("REC DEF13"));
    //weight             CO2             =>             CO2-((storage
losses*12.0*D.Value("REC_DEF13"))*(mFUEL_CARBON_INTENSITY/3.6))
    //Note that the following equation includes a division by 3.6 to
convert MJ to kWh before calculating the carbon
    mDHW_CO2_WEIGHTED=mKG_CO2-
((mSTORE LOSS*12.0*D.Value("REC DEF13"))*(mFUEL CARBON INTENSITY/3.6)
);

    //variable for adjusted Total Building energy
    mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mQ_PRIM_PRICE_WEIGHTED+(mDHW_ENERGY_WEIGHTED*Fuel_Price_Weighting);
    //variable for adjusted Total Building CO2
    mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-mKG CO2+mDHW CO2 WEIGHTED;
```

```
    //assess energy and CO2 impact
    IMPACT ENERGY=fIMPACT ENERGY(D,          mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
    IMPACT CO2=fIMPACT CO2(D, mKG CO2 PROJECT, mCO2 TOTAL ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
    //Multiply default payback by 10 and divide by NOTIONAL building
DHW consumption (kWh/m2.year)
    if((mAREA!=0)&&(TNR.notKWH DHW!=0))
        PAYBACK=D.Value("dEPC_W3")*10.0/(TNR.notKWH_DHW/mAREA);
    else
        PAYBACK=0.0;

    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON_SAVING_POUND value (GOOD, FAIR, POOR)
    CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

    //write the EPCW3 keyword (DHW system storage losses
recommendations):
    string EPC XX="EPC-W3";
    Write Rec Keyword(D, T, P, EPC XX, IMPACT ENERGY, IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
  }

  //---- END recommendation EPC-W3 ----


  //---- recommendation EPC-W4 ----

  //secondary circulation losses
  if (EPCW4) {

     //assess impact by reducing DHW energy by 30%:

    //weight energy => energy*(1.0-D.Value("REC DEF14"))
    mDHW ENERGY WEIGHTED=mQ PRIM*(1.0-D.Value("REC DEF14"));
    //weight CO2 => CO2*(1.0-D.Value("REC_DEF14"))
    mDHW CO2 WEIGHTED=mKG CO2*(1.0-D.Value("REC DEF14"));

    //variable for adjusted Total Building energy
    mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mQ PRIM PRICE WEIGHTED+(mDHW ENERGY WEIGHTED*Fuel Price Weighting);
    //variable for adjusted Total Building CO2
    mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-mKG CO2+mDHW CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=fIMPACT ENERGY(D,          mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
    IMPACT_CO2=fIMPACT_CO2(D, mKG_CO2_PROJECT, mCO2_TOTAL_ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON_SAVING_POUND)
    //Multiply default payback by 10 and divide by NOTIONAL building
DHW consumption (kWh/m2.year)
    if((mAREA!=0)&&(TNR.notKWH_DHW!=0))
        PAYBACK=D.Value("dEPC W4")*10.0/(TNR.notKWH DHW/mAREA);
    else
        PAYBACK=0.0;

    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)
    CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

    //write the EPCW4 recommendation keyword
    string EPC XX="EPC-W4";
    Write Rec Keyword(D, T, P, EPC XX, IMPACT ENERGY, IMPACT CO2,
```

```
PAYBACK, CARBON SAVING POUND);
  }
  //---- END recommendation EPC-W4 ----

}


void       DHW_GENERATOR::Write_Rec_Keyword(PROJECTDefaults&       D,
PROJECTTables&  T,  BOC&  P,  string  EPC XX,  int  IMPACT ENERGY,  int
IMPACT CO2, double PAYBACK, int CARBON SAVING POUND) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"DHW GENERATOR::Write Rec Keyword    "    <<  EPC XX  <<  "    "  <<
IMPACT ENERGY << "   " << IMPACT CO2 << "   " << PAYBACK << "   " <<
CARBON_SAVING_POUND;
/*

  Function to write keywords for EPC recommendations
  String EPC XX is name of recommendation keyword e.g. EPC-H1

*/


  //first see what the value of the keyword is. User may have set
some or all
  //of the values or set that this recommmendation should not be used
etc...
  deque<double> mEPC XX=P.Vlnum(Id,EPC XX);

  if (mEPC XX.size()==1) {
    //user has not modified this recommendation
    for (int i=0;i<4;i++) mEPC XX.push back(0.0);
  }
  if (mEPC XX[0]!=1) {
    //user modified this recommendation
    //write the parameters which were not set by the user
    //ENERGY impact (1=HIGH/2=MEDIUM/3=LOW)
    if ((mEPC XX[1]==-1)||(mEPC XX[1]==0)) mEPC XX[1]=IMPACT ENERGY;
    //CO2 impact (1=HIGH/2=MEDIUM/3=LOW) always low
    if ((mEPC_XX[2]==-1)||(mEPC_XX[2]==0)) mEPC_XX[2]=IMPACT_CO2;
    //payback period (number)
    if ((mEPC XX[3]==-1)||(mEPC XX[3]==0)) mEPC XX[3]=PAYBACK;
    //Carbon saving per pound (1=GOOD/2=FAIR/3=POOR)
    if                                ((mEPC XX[4]==-1)||(mEPC XX[4]==0))
mEPC XX[4]=CARBON SAVING POUND;
    //Id is included (user has modified values)
    if (mEPC XX.size()==6) {
      int REC USER Id=(int)mEPC XX[5];
      {
       ostringstream res; res << "{ ";
       for (int i=0;i<5;i++)
         res << mEPC_XX[i] << ", ";
       res << REC USER Id << " }";
       P.ModKey(Id,EPC XX,res.str());
      }
    }
    else {
      //calc  in  -  sbem  decided  that  the  recommendation  should  be
included and set the values
      mEPC XX[0]=4;
      {
       ostringstream res; res << "{ ";
       for (int i=0;i<4;i++)
         res << mEPC_XX[i] << ", ";
       res << mEPC XX[4] << " }";
       P.ModKey(Id,EPC XX,res.str());
      }
    }
  }
```

```
} //end of DHW GENERATOR::Write Rec Keyword(...)
```

DOOR

Specifies the size and construction of a door. Each DOOR command applies to the WALL instruction preceding it and describes a door in that envelope element.

Example input:

```
"z0/01/door" = DOOR
   TYPE             = Personnel Doors
   AREA             = 45
   CONSTRUCTION     = "External door for example building"
```

TYPE
Specifies the type of the door, as one of the values listed below. The default is 'Personnel Doors'.

Personnel Doors
Vehicle Access Doors
High Usage Entrance Doors

AREA
Specifies the area of the door, in m^2. This is a required keyword.

CONSTRUCTION
Specifies the U-name of a previously-defined CONSTRUCTION instruction to define the construction of the door. This is a required keyword.

MULTIPLIER
This keyword specifies how many doors with this exact description exist in the wall. E.g., if the value of the multiplier is 3, there are 3 doors each with identical keyword values in the wall considered. This enables repeating doors to be entered quickly and easily. The default value is 1.

c++ code

```
//: door.cpp
// added ID9

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void DOOR::TEST() {
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test DOOR" << endl;
//  cout << "Fin de Test DOOR" << endl;
  BEEMObject::TEST();
}

void DOOR::Reset() {
  if(func report) cout << endl << "  " << "DOOR::Reset";
  dTM.clear();
//  vTM.clear();
```

```
}

void DOOR::DtoV() {
  if(func report) cout << endl << "  " << "DOOR::DtoV";
//  vTM.reserve(dTM.size());
//  vTM.assign(dTM.begin(),dTM.end());
//  dTM.clear();
}

void DOOR::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P, BOC
&N, double mAREA) {
  if(func report) cout << endl << "   " << "DOOR::Notional   "   <<
mAREA;
  char C=34;
  BObject Temp;
  Temp.Ntype=17;
  Temp.Type="DOOR";
  Temp.Name=P.vINP[Id].Name + " Not DOOR";
  KeywordValue KV;

 {
  ostringstream res; res << "Personnel Doors";
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("TYPE",KV));
 }
 {
  ostringstream res; res << C << "Notional Personnel Door" << C;
  KV.Reset(); KV.Init(3,res.str());
  Temp.Prop.insert(make pair("CONSTRUCTION",KV));
 }
 {
  ostringstream res; res << mAREA;
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("AREA",KV));
 }
 {
  ostringstream  res;  res  <<  1.0;                //this  is  already
considered when calculating area
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("MULTIPLIER",KV));
 }
  N.vINP.push_back(Temp);
}

void  DOOR::NotionalVdoor(PROJECTDefaults&  D,  PROJECTTables&  T,  BOC
&P, BOC &N, double mAREA) {
  if(func report) cout << endl << "  " << "DOOR::NotionalVdoor  " <<
mAREA;
  char C=34;
  BObject Temp;
  Temp.Ntype=17;
  Temp.Type="DOOR";
  Temp.Name=P.vINP[Id].Name + " NotV Door";
  KeywordValue KV;

 {
  ostringstream res; res << "Vehicle Access Doors";
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("TYPE",KV));
 }
 {
 ostringstream res; res << C << "Notional Vehicle Door" << C;
 KV.Reset(); KV.Init(3,res.str());
 Temp.Prop.insert(make pair("CONSTRUCTION",KV));
 }
 {
  ostringstream res; res << mAREA;
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make_pair("AREA",KV));
```

```
  }
  {
   ostringstream res; res << 1.0;          //this is already considered
when calculating area
   KV.Reset(); KV.Init(1,res.str());
   Temp.Prop.insert(make pair("MULTIPLIER",KV));
  }

  N.vINP.push back(Temp);
}

void  DOOR::NotionalHdoor(PROJECTDefaults&  D,  PROJECTTables&  T,  BOC
&P, BOC &N, double mAREA) {
  if(func report) cout << endl << "   " << "DOOR::NotionalHdoor   " <<
mAREA;
  char C=34;
  BObject Temp;
  Temp.Ntype=17;
  Temp.Type="DOOR";
  Temp.Name=P.vINP[Id].Name + "_NotH_Door";
  KeywordValue KV;

 {
  ostringstream res; res << "High Usage Entrance Doors";
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("TYPE",KV));
 }
 {
 ostringstream res; res << C << "Notional High Usage Entrance Door"
<< C;
 KV.Reset(); KV.Init(3,res.str());
 Temp.Prop.insert(make pair("CONSTRUCTION",KV));
 }
 {
  ostringstream res; res << mAREA;
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("AREA",KV));
 }
 {
  ostringstream res; res << 1.0;          //this is already considered
when calculating area
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("MULTIPLIER",KV));
 }

  N.vINP.push back(Temp);
}

void DOOR::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "   " << "DOOR::Calc";
  if(calc_report) cout << endl << P.vINP[Id].Name;
  for (int i=0; i<vTM.size();i++)
    vTM[i].Calc(D, T, P);
//its own calc below...
  fU VALUE(D, T, P);
  fCM(D, P);
  fH_TR(D, P);
  fQ_SUN NT(D, T, P);
}

void DOOR::fU VALUE(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "   " << "DOOR::fU VALUE";
  double mU_VALUE;

  // Check if this is the 2010 notional as it affects the U-value
which in this case, depends on the DRIVER2A keyword of the
  // parent zone
  if   (   (P.Vcod(gId,"ACT-NOT")=="NOT")      &&   (P.Vint(ppId,"NOT-
CASE")>=9) )  { // 9=EW_NOT_2010
```

```
    if (P.Vcodid(Id,"TYPE")==1)  // 1=Personnel Doors
      mU VALUE=T.Value("Not U Values",5,P.Vint(ppId,"NOT-CASE"));
    else if (P.Vcodid(Id,"TYPE")==2)  // 2=Vehicle Access Doors
           mU VALUE=T.Value("Not U Values",7,P.Vint(ppId,"NOT-
CASE"));
         else // i.e., 3=High Usage Entrance Doors
           mU VALUE=T.Value("Not U Values",6,P.Vint(ppId,"NOT-
CASE"));
  } // end if

  else // not 2010 notional
    mU VALUE=P.Vnum(P.Vrefid(Id,"CONSTRUCTION"),"U-VALUE");


 {
  ostringstream res; res << mU VALUE;
  P.ModKey(Id,"U-VALUE",res.str());
 }
}

void DOOR::fCM(PROJECTDefaults& D, BOC& P) {
  if(func report) cout << endl << "  " << "DOOR::fCM";

  deque<double> mCM=P.Vlnum(P.Vrefid(Id,"CONSTRUCTION"),"CM");
  for(int i=0;i<2;i++) mCM[i]*=P.Vnum(Id,"AREA");

{
  ostringstream res; res << "{ ";
  res << mCM[0] << ", ";
  res << mCM[1] << " }";
  P.ModKey(Id,"CM",res.str());
 }
}

void DOOR::fH TR(PROJECTDefaults& D, BOC& P) {
  if(func report) cout << endl << "  " << "DOOR::fH TR";
  double mA, mH TR;
  double mAREA=P.Vnum(Id,"AREA");
  double mU_VALUE=P.Vnum(Id,"U-VALUE");

  switch(P.Vcodid(pId,"TYPE")) {
  case 1: {                     //exterior
           mA=D.Value("D011");
          }
           break;
  case 2: {                      //strongly ventilated space
           mA=D.Value("D011");
          }
           break;
  case 3: {                      //unheated space
           mA=D.Value("D011");
          }
           break;
  case 4: {                      //conditioned space
           mA=0;
          }
           break;
  case 5: {                      //underground
//           mA=1/(1+mU VALUE);
// The line above has been changed as below so that the heat transfer
coefficient calculation for ground floors is the same as
// that for the other envelopes (except those adjacent to CAS where
coefficient is 0).
// NEN2916 says that the old equation is based on transmission heat
losses of floors of dwellings in which the effect of thermal
// bridges at edges is incorporated. For larger floor areas this can
lead to overestimation of heat losses. Hence, the change. - AN
15/08/2007
           mA=D.Value("D011");
```

```
          }
           break;
  case 6: {                    //Same space
           mA=0;
          }
           break;
  }

  mH TR=mA*mAREA*mU VALUE;
  for (int i=0;i<vTM.size();i++)
    mH_TR+=P.Vnum(vTM[i].Id,"H-TR")*P.Vnum(vTM[i].Id,"MULTIPLIER");
 {
  ostringstream res; res << mH TR;
  P.ModKey(Id,"H-TR",res.str());
 }

}

void DOOR::fQ SUN NT(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "  " << "DOOR::fQ_SUN_NT";
  double mAREA=P.Vnum(Id,"AREA");
  double Q SUN[12];
  string mWEATHER;

  if(gId<0) mWEATHER=" LON"; else mWEATHER=" "+P.Vcod(gId,"WEATHER");
  for (int i=0;i<12;i++)

Q SUN[i]=T.Value("T010"+mWEATHER,i+1,P.Vcodid(pId,"ORIENTATION"));

  double mQ SUN NT[12];
  for (int i=0;i<12;i++)
    mQ SUN NT[i]=D.Value("D039")*Q SUN[i]*P.Vnum(Id,"U-VALUE")*mAREA;
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ SUN NT[i] << ", ";
  res << mQ SUN NT[11] << " }";
  P.ModKey(Id,"Q-SUN-NT",res.str());
 }
}

void DOOR::Recommendation(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func report) cout << endl << "  " << "DOOR::Recommendation";
  // First Child Recommendation?
  //all door thermal bridges
  for (int i=0; i<vTM.size();i++)
    vTM[i].Recommendation(D,T,P);

  // Then Door Recommendation?

}

// End Object: DOOR
```

DOOR-TM-BRIDGE

Specifies the linear thermal transmittance and length of a thermal bridge associated with a door.

Example input:

```
"TM1_D" = DOOR-TM-BRIDGE
  PSI            = 2
  L              = 0.4
```

. .

PSI
Specifies the linear thermal transmittance of the thermal bridge, in W/(mK). The default value is 2.

L
Specifies the length of the individual linear thermal bridge, in m. The default value is 10.

MULTIPLIER
This keyword specifies how many linear thermal bridges with this exact description exist in the current door. E.g., if the value of the multiplier is 3, there are 3 linear thermal bridges each with identical keyword values in the door considered. This enables repeating linear thermal bridges to be entered quickly and easily. The default value is 1.

c++ code

```
//: tm bridge.cpp

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void TM BRIDGE::TEST() {
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test TM B" << endl;
//  cout << "Fin de Test TM B" << endl;
  BEEMObject::TEST();
}

// v4.3.0 Addition of
// TM BRIDGE::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P,
BOC&N, int i)
void  TM BRIDGE::Notional(PROJECTDefaults&  D,  PROJECTTables&  T,  BOC
&P, BOC &N, int i) {
  if(func report) cout << endl << "   " << "TM BRIDGE::Notional " <<
i;

  N.vINP.push back(P.vINP[i]);
}
// end v4.3.0 Addition of
// TM BRIDGE::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P,
BOC&N, int i)

void TM BRIDGE::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "  " << "TM_BRIDGE::Calc";
  if(calc_report) cout << endl << P.vINP[Id].Name;
    fH TR(P);
}

void TM BRIDGE::Init(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "  " << "TM_BRIDGE::Init";
//  fH TR(P);
}

void TM BRIDGE::fH TR(BOC& P) {
  if(func report) cout << endl << "  " << "TM BRIDGE::fH TR";
  double mH_TR=P.Vnum(Id, "PSI")*P.Vnum(Id, "L");

  {
  ostringstream res; res << mH_TR;
  P.ModKey(Id,"H-TR",res.str());
```

```
  }

}

void TM_BRIDGE::Recommendation(PROJECTDefaults& D, PROJECTTables& T,
BOC& P) {
  if(func report)    cout    <<    endl    <<    "         "    <<
"TM_BRIDGE::Recommendation";

}



// End Object: TM_BRIDGE
```

GENERAL

The first object in the input file must be the GENERAL object. This specifies general data for the whole building.

Example input:

```
"SBEM-PROJECT" = GENERAL
   B-TYPE              = B1 Offices and Workshop businesses
   WEATHER             = LON
   UPRN                = "000000000000"
   B-INSP-DATE         = { 2008, 02, 01 }
   P-NAME              = "Example building"
   B-ADDRESS-0         = "Example building"
   B-ADDRESS-1         = "56 London Road"
   B-ADDRESS-2         = ""
   B-ADDRESS-3         = ""
   B-COUNTY            = "LONDON"
   B-POSTCODE          = "SW23 1HA"
   O-NAME              = "John Jones"
   O-TELEPHONE         = "987654321"
   O-ADDRESS           = "53 London Road"
   O-COUNTY            = "LONDON"
   O-POSTCODE          = "SW23 1HA"
   C-NAME              = "<insert name>"
   C-TELEPHONE         = "99999999999"
   C-ADDRESS           = "<insert address>"
   C-COUNTY            = "<insert county>"
   C-POSTCODE          = "XX XXX"
   C-REG-NUMBER        = "XXXX000000"
   C-ACCRED-SCHEME     = "<Accreditation scheme>"
   C-EMP-TRAD-NAME     = "<insert Employer/Trading Name>"
   C-EMP-TRAD-ADDRESS  = "<insert Employer/Trading Address>"
   C-REL-PART-DISC     = Not related to the owner
   INTERFACE           = "iSBEM"
   INTERFACE-VERSION   = "v4.1.a"
```

```
   PATH-FILE-INTERFACE =
"C:\NCM\iSBEM_v4.1.a\iNCT\nctproject.mdb"
   ACT-NOT             = ACT
   BUILDING-AREA       = 2900
   ELEC-POWER-FACTOR   = >0.95
   LIGHT-METERING      = 0.95
   BUILD-ORIENTATION   = 0
   PRODUCE-MESSAGES = YES
   ..
```

P-NAME
Specifies the project's name.

UPRN
Specifies the Unique Property Reference Number (UPRN) of the building. This keyword is included in the input if the COMPLIANCE object's EPC-TYPE keyword has the value 'EPC England', 'EPC Wales', or 'EPC Northern Ireland'. It consists of 12 digits.

MPRN
Specifies the Meter Point Registration Number (MPRN) of the building. This keyword appears in the input if the COMPLIANCE object's EPC-TYPE keyword has the value 'Republic of Ireland: Building Energy Rating'. It consists of 12 digits which must start with 0.

B-INSP-DATE
Specifies the inspection date of the building by the Energy Assessor, in the form shown below (enclosed by braces and separated by commas). This keyword appears in the input file if the COMPLIANCE object keyword EPC-TYPE has the value 'EPC England', 'EPC Wales', or 'EPC Northern Ireland'.

{ yyyy, mm, dd }

B-CONS-DATE
Specifies the construction date of the building, in the form shown below (enclosed by braces and separated by commas). This keyword appears in the input if the COMPLIANCE object's EPC-TYPE keyword has the value 'Republic of Ireland: Building Energy Rating'.

{ yyyy, mm, dd }

B-TYPE
Specifies the building type, as one of the value listed below. The default is 'B1 Offices and Workshop businesses'.

A1/A2 Retail and Financial/Professional services
A3/A4/A5 Restaurant and Cafes/Drinking Establishments and Hot Food takeaways
B1 Offices and Workshop businesses
B2 to B7 General Industrial and Special Industrial Groups

B8 Storage or Distribution
C1 Hotels
C2 Residential Institutions - Hospitals and Care Homes
C2 Residential Institutions - Residential schools
C2 Residential Institutions - Universities and colleges
C2A Secure Residential Institutions
Residential spaces
D1 Non-residential Institutions - Community/Day Centre
D1 Non-residential Institutions - Libraries Museums and Galleries
D1 Non-residential Institutions - Education
D1 Non-residential Institutions - Primary Health Care Building
D1 Non-residential Institutions - Crown and County Courts
D2 General Assembly and Leisure plus Night Clubs and Theatres
Others - Passenger terminals
Others - Emergency services
Others -Telephone exchanges
Others - Miscellaneous 24hr activities
Others - Car Parks 24 hrs
Others - Stand alone utility block

B-ADDRESS **(NO LONGER IN USE)**
Specifies one line of the building street address.

B-ADDRESS-0
Specifies one line of the building address.

B-ADDRESS-1
Specifies a first additional line of the building address.

B-ADDRESS-2
Specifies a second additional line of the building address.

B-ADDRESS-3
Specifies a third additional line of the building address.

B-LOC-DESCRIPTION
Provides a description of the building location.

B-CITY
Specifies the city where the building is located. This keyword is not included in the input if the COMPLIANCE object's BR-TYPE keyword has the value 'Republic of Ireland: Building Regulations Part L', or the COMPLIANCE object's EPC-TYPE keyword has the value 'Republic of Ireland: Building Energy Rating'.

B-POSTCODE
Specifies the postcode where the building is located. This keyword is not included in the input if the COMPLIANCE object's BR-TYPE keyword has the value 'Republic of Ireland: Building Regulations Part L', or the COMPLIANCE object's EPC-TYPE keyword has the value 'Republic of Ireland: Building Energy Rating'.

B-COUNTY

Specifies the county where the building is located. This keyword is included in the input if the COMPLIANCE object's BR-TYPE keyword has the value 'Republic of Ireland: Building Regulations Part L', or the COMPLIANCE object's EPC-TYPE keyword has the value 'Republic of Ireland: Building Energy Rating'.

O-NAME
Specifies the name of building owner.

O-TELEPHONE
Specifies the telephone number of the building owner.

O-ADDRESS
Specifies the street address of the building owner.

O-CITY
Specifies the city where the building owner resides. This keyword is not included in the input if the COMPLIANCE object's BR-TYPE keyword has the value 'Republic of Ireland: Building Regulations Part L', or the COMPLIANCE object's EPC-TYPE keyword has the value 'Republic of Ireland: Building Energy Rating'.

O-POSTCODE
Specifies the postcode where building owner resides. This keyword is not included in the input if the COMPLIANCE object's BR-TYPE keyword has the value 'Republic of Ireland: Building Regulations Part L', or the COMPLIANCE object's EPC-TYPE keyword has the value 'Republic of Ireland: Building Energy Rating'.

O-COUNTY
Specifies the building owner's county. This keyword is included in the input if the COMPLIANCE object's BR-TYPE keyword has the value 'Republic of Ireland: Building Regulations Part L', or the COMPLIANCE object's EPC-TYPE keyword has the value 'Republic of Ireland: Building Energy Rating'.

C-NAME
Specifies the Certifier's/Energy Assessor's name (Agent/Qualified Accredited Person in Scotland).

C-TELEPHONE
Specifies the Certifier's/Energy Assessor's telephone number (Agent/Qualified Accredited Person in Scotland).

C-ADDRESS
Specifies the Certifier's/Energy Assessor's street address (Agent/Qualified Accredited Person in Scotland).

C-CITY
Specifies the city where the Certifier/Energy Assessor (Agent/Qualified Accredited Person in Scotland) resides. This keyword is not included in the input if the COMPLIANCE object's BR-TYPE keyword has the value 'Republic of Ireland: Building Regulations Part L', or the COMPLIANCE object's EPC-TYPE keyword has the value 'Republic of Ireland: Building Energy Rating'.

C-POSTCODE
Specifies the postcode where the Certifier/Energy Assessor (Agent/Qualified Accredited Person in Scotland) resides. This keyword is not included in the input if the COMPLIANCE object's BR-TYPE keyword has the value 'Republic of Ireland: Building Regulations Part L', or the COMPLIANCE object's EPC-TYPE keyword has the value 'Republic of Ireland: Building Energy Rating'.

C-COUNTY
Specifies the county where the certifier resides. This keyword is included in the input if the COMPLIANCE object's BR-TYPE keyword has the value 'Republic of Ireland: Building Regulations Part L', or the COMPLIANCE object's EPC-TYPE keyword has the value 'Republic of Ireland: Building Energy Rating'.

C-ACCRED-SCHEME
Specifies the name of the Certifier's/Energy Assessor's accreditation scheme provider.

C-REG-NUMBER
Specifies the Certifier's/Energy Assessor's registration number within the accreditation scheme. This keyword is included in the input when the COMPLIANCE object's EPC-TYPE keyword is set to 'EPC England', 'EPC Wales', or 'EPC Northern Ireland'.

C-REG-NUMBER-S
Specifies the Certifier's/Energy Assessor's registration number within the accreditation scheme. This keyword is included in the input when the COMPLIANCE object's EPC-TYPE keyword is set to 'EPC Scotland'.

C-REG-NUMBER-IR
Specifies the Certifier's/Energy Assessor's registration number within the accreditation scheme. This keyword is included in the input file when the keyword EPC-TYPE is set to 'Republic of Ireland: Building Energy Rating'.

***C-COMPANY* (NO LONGER IN USE)**
Specifies the Certifier's company.

C-EMP-TRAD-NAME
Specifies the Certifier's/Energy Assessor's employer/trading name.

C-EMP-TRAD-ADDRESS
Specifies the Certifier's/Energy Assessor's employer/trading address.

C-EMP-TRAD-NUMBER
Specifies the Certifier's/Energy Assessor's employer/trading number. This keyword is included in the input when the COMPLIANCE object's EPC-TYPE keyword is set to 'Republic of Ireland: Building Energy Rating'.

C-REL-PART-DISC
Specifies the related party disclosure by the Energy Assessor, as one of the values listed below. This is a required keyword if the COMPLIANCE object's EPC-TYPE keyword has the value 'EPC England', 'EPC Wales', or 'EPC Northern Ireland'.

Not related to the owner
Relative of the owner
Relative of the professional dealing with the property transaction
Indirect relation to the owner (e.g. relative employed/contracted by owner)
Occupier of the property
Owner or Director of the organisation dealing with the property transaction
Employed by the organisation dealing with the property transaction
Financial interest in the property
Employed by the owner
Contracted by the owner to provide other Energy Assessment services
Contracted by the owner to provide other (non Energy Assessment) services
Previous relation to the owner (e.g. employee/contractor)

TRANSACTION-TYPE
Specifies the transaction type that prompted the generation of the EPC. This is a required keyword if the COMPLIANCE object's EPC-TYPE keyword has the value 'EPC England', 'EPC Wales', or 'EPC Northern Ireland'.

Mandatory issue (Marketed sale)
Mandatory issue (Non-marketed sale)
Mandatory issue (Property on construction)
Mandatory issue (Property to let)
Voluntary re-issue (A valid EPC is already lodged)
Voluntary (No legal requirement for an EPC)

C-QUALIFICATIONS
Specifies the qualifications of the energy assessor, as one of the values listed below. The default is 'NOS3'.

NOS3
NOS4
NOS5

C-INSURER
Specifies the Energy Assessor's insurance company.

C-INS-POL-NUMBER
Specifies the Energy Assessor's insurance policy number.

C-INS-EFF-DATE
Specifies the Energy Assessor's insurance policy effective start date.

C-INS-EXP-DATE
Specifies the Energy Assessor's insurance policy expiry date.

C-INS-PI-LIMIT
Specifies the Energy Assessor's insurance policy cover limit.

NOS-LEVEL

Specifies the complexity level of the project, as one of the values listed below. The default is 'Level 3'.

Level 3
Level 4
Level 5

INTERFACE-VAL
This keyword is for validation purposes. It specifies the name of the interface which is to be validated as the one actually used to generate the input file for SBEM, as one of the values listed below. The default is 'Not Approved'.

Not Approved
iSBEM
CARBON-CHECKER
HEVACOMP
IES
DESIGN-BUILDER
DIGITALENERGY
ECAT

INTERFACE
Name of the software interface used to produce the input file for SBEM. This keyword is required.

INTERFACE-VERSION
Version of the software interface used to produce the input file for SBEM. This keyword is required.

SOFT-COMP-NAME
Name of the company whose software tool has been approved. This keyword is required.

PATH-FILE-INTERFACE
Specifies the path location of the input file to the interface software.

WEATHER
Specifies the weather data to use for the energy calculations of the building (depending on its location), as one of the values listed below. This is a required keyword.

Value	Description
BEL	Belfast - *Only option for Northern Ireland Building Regulations*
BIR	Birmingham
CAR	Cardiff
DUB	Dublin - *Only available for Republic of Ireland Building Regulations*
EDI	Edinburgh - *Only available for England & Wales Building Regulations*
GLA	Scotland - *Only option for Scottish Building Regulations*
GLA	Glasgow - *Only available for England & Wales Building Regulations*
JER	Jersey – *Only available for Building Bye-laws Jersey Part 11*
LEE	Leeds
LON	London
MAN	Manchester

NEW	Newcastle
NOR	Norwich
NOT	Nottingham
PLY	Plymouth
SOU	Southampton
SWI	Swindon

ACT-NOT
Specifies whether the input file is for the actual or the notional building, using one of the values listed below. This is a required keyword.

Value	Description
ACT	Actual building
NOT	Notional building

***H-STACK* (NO LONGER IN USE)**
This is the building height (to eaves), in m. This is a required keyword.

BUILDING-AREA
Specifies the total floor area of the building, in m^2.

BUILDING-VOLUME
Specifies the volume of the building, in m^3. This keyword is included in the input if the energy calculations are for the Republic of Ireland.

MAX-STOREY
Specifies the maximum number of storeys in the building.

ELEC-POWER-FACTOR
Specifies the building's electrical power factor, as one of the values listed below. The default is '<0.9'.

<0.9
0.9 to 0.95
>0.95

LIGHT-METERING
This is a correction factor due to light metering in the building. If lighting systems have provision for metering, and the metering warns about out-of-range values, this keyword is equal to 0.95. If lighting systems have no provision for metering, or the metering does not warn about out-of-range values, this keyword is equal to 1, i.e., no correction. The default value is 1.

***MODULAR-2YEARS* (NO LONGER IN USE)**
This keyword specifies whether the building is a modular and/or portable building with planned time of use of 2 years or less. The value can be YES or NO. The default is NO.

BUILD-ORIENTATION
Specifies the orientation of the building in degrees from north, as one of the values listed below. The default value is 0.

0
45
90
135
180
225
270
315

GAS-AVAILABILITY
This keyword is included in the input file only if the COMPLIANCE object's EPC-TYPE keyword has the value 'EPC Scotland', or the COMPLIANCE object's BR-TYPE keyword has the value 'Scottish Building Regulations 2010'. It specifies whether natural gas is available on site. The value can be YES or NO. The default is YES.

RENEWABLE-MAIN
This keyword is included in the input file only if the COMPLIANCE object's EPC-TYPE keyword has the value 'EPC Scotland'. It can have one of the values listed below. The default is 'From calculation'.

Value	Description
From calculation	The main renewable source will be determined by SBEM based on the input parameters and the calculated results.
Other non-calculated	The main renewable source is one that cannot be modelled using SBEM.

PAUSE-KNOWN-ERRORS
This keyword specifies whether the execution of SBEM is required to pause if known errors are encountered. The value can be YES or NO. The default is YES.

DH-CO2F
Specifies the CO_2 emission factor if the fuel type 'District Heating' is used anywhere in the building, in $kgCO_2/kWh$. The default value is 0.293.

DH-PRIMF
Specifies the primary energy conversion factor if the fuel type 'District Heating' is used anywhere in the building, in kWh/kWh. The default value is 1.2. This keyword is included in the input file only if the COMPLIANCE object's EPC-TYPE keyword has the value 'Republic of Ireland: Building Energy Rating', or the COMPLIANCE object's BR-TYPE keyword has the value 'Republic of Ireland: Building Regulations Part L'.

PRODUCE-XML-MESSAGES **(NO LONGER IN USE)**
This keyword specifies whether the production of XML messages is required. The value can be YES or NO. The default is YES.

HTML-DR-REPORTS
This keyword specifies whether the production of the Data Reflection reports in html format, for both the actual and notional buildings, is required. The value can be YES or NO. The default is NO.

HTTP-PROXY-SERVER

Specifies the URL address of the proxy server if one is being used (by a Local Area Network) for connecting the user's computer to the Internet.

HTTP-PROXY-PORT
Specifies the port of the proxy server if one is being used by a (Local Area Network) for connecting user's computer to the Internet.

DLLEXE-CALL
Specifies whether "dll" files for the EPCgen and BERgen modules are called from within SBEM.exe, or their "exe" files need to be called separately following the execution of SBEM.exe. It can have one of the values listed below, and the default value is 'EXE'.

DLL
EXE

New

LANDMARK-ENVIRONMENT
Specifies the environment at Landmark to which EPCgen should connect, as one of the values listed below. The default value is 'www'. This keyword is for developers' and Accreditation Scheme Providers' testing purposes only following the implementation of changes that affect the register. Energy Assessors have no need to view this keyword or alter it from its default value.

Value	Description
www	Live Energy Assessor's register
uat	User Acceptance Testing
ote	Open Transaction Environment

* Note: For testers using iSBEM, in order for the parameter corresponding to this keyword to appear in the interface, you need to create a text file that contains the text 'Landmark Tests' (without the quotes) in the top line, name it 'as.tests' (with no other extension), and ensure it is saved within the 'iSBEM_v4.1.a' folder on your computer. To make the parameter invisible again, remove the file 'as.tests' from the iSBEM folder. Ensure that the parameter is set to 'www' before you make it invisible.

c++ code

```
//: general.cpp

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void GENERAL::Init(PROJECTTables& T, BOC &P) {
  if(func report) cout << endl << "  " << "GENERAL::Init";

  if(P.Vnum(Id,"DH-CO2F")!=-5555){
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6)) {
          T.ValueMod("tKGCO2 IR",1,13,P.Vnum(Id,"DH-CO2F"));
          T.ValueMod("tPRIM IR",1,13,P.Vnum(Id,"DH-PRIMF"));
      } // end if
      else if((P.Vcodid(cId,"BR-TYPE")==6)) {
             T.ValueMod("tKGCO2 JER",1,13,P.Vnum(Id,"DH-CO2F"));
             T.ValueMod("tPRIM_JER",1,13,P.Vnum(Id,"DH-PRIMF"));
           } // end if
           else // i.e., all UK
             T.ValueMod("tKGCO2",1,13,P.Vnum(Id,"DH-CO2F"));
```

```
  }

  // B-ADDRESS = B-ADDRESS-0 + B-ADDRESS-1 + B-ADDRESS-2 + B-ADDRESS-
3
    string bAddress;
    bool flag0, flag1, flag2, flag3;
    flag0=0; flag1=0; flag2=0; flag3=0;
    if(StripQuotes(P.Vtext(gId,"B-ADDRESS-0"))!="") flag0=1;
    if(StripQuotes(P.Vtext(gId,"B-ADDRESS-1"))!="") flag1=1;
    if(StripQuotes(P.Vtext(gId,"B-ADDRESS-2"))!="") flag2=1;
    if(StripQuotes(P.Vtext(gId,"B-ADDRESS-3"))!="") flag3=1;
    int c1, c2, c3;
    c1=flag0; c2=flag0+flag1; c3=flag0+flag1+flag2;
    if(!flag1) c1=0;
    if(!flag2) c2=0;
    if(!flag3) c3=0;

    { ostringstream os;
      char C=34;
      os << C;
      if(flag0) os << StripQuotes(P.Vtext(gId,"B-ADDRESS-0"));
      if(c1>0)  os << ", ";
      if(flag1) os << StripQuotes(P.Vtext(gId,"B-ADDRESS-1"));
      if(c2>0)  os << ", ";
      if(flag2) os << StripQuotes(P.Vtext(gId,"B-ADDRESS-2"));
      if(c3>0)  os << ", ";
      if(flag3) os << StripQuotes(P.Vtext(gId,"B-ADDRESS-3"));

      os << C;
      bAddress=os.str();
    }

 {
  ostringstream res; res << bAddress;
  P.ModKey(Id,"B-ADDRESS",res.str());
 }

  // Ensuring that a valid option for related party disclosure and
for transaction type have been input
  // for EPCs in England, Wales, and NI
  if   (   (P.Vcod(gId,"ACT-NOT")=="ACT")   &&   (   (P.Vcodid(cId,"EPC-
TYPE")==2)  ||  (P.Vcodid(cId,"EPC-TYPE")==3)  ||  (P.Vcodid(cId,"EPC-
TYPE")==5) ) ) {

    if(P.Vcodid(gId,"C-REL-PART-DISC")==13) { // 13=Not recorded.
      {ostringstream os; os << endl << endl << "ERROR: \"Related
Party Disclosure\" is undefined in " << P.vINP[Id].Name << endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
    } // end if

    if(P.Vcodid(gId,"TRANSACTION-TYPE")==7) { // 7=Not recorded.
      {ostringstream os; os << endl << endl << "ERROR: \"Transaction
Type\" is undefined in " << P.vINP[Id].Name << endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
    } // end if

  } // end if


  // The function below was moved here so it is executed early on and
the ratio is available to the renewables
  // to adjust their contributions accordingly
  fSC ACTUAL RATIO(T, P);

}

//  GENERAL::Notional(PROJECTDefaults&  D,  PROJECTTables&  T,  BOC  &P,
```

```
BOC&N)
void GENERAL::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P,
BOC &N) {
  if(func report) cout << endl << "  " << "GENERAL::Notional";
// Alternative one
  BObject Temp;
  Temp.Ntype=15;
  Temp.Type="GENERAL";
  Temp.Name="SBEM-PROJECT-NOTIONAL";
  KeywordValue KV;
 {
  ostringstream res; res << "NOT";
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make_pair("ACT-NOT",KV));
 }
 {
  ostringstream res; res << P.Vcod(Id,"WEATHER");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("WEATHER",KV));
 }
 {
  ostringstream res; res << P.Vtext(Id,"P-NAME");
  KV.Reset(); KV.Init(2,res.str());
  Temp.Prop.insert(make pair("P-NAME",KV));
 }
 {
  ostringstream res; res << P.Vcod(Id,"B-TYPE");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("B-TYPE",KV));
 }
 {
  ostringstream res; res << P.Vtext(Id,"B-ADDRESS");
  KV.Reset(); KV.Init(2,res.str());
  Temp.Prop.insert(make pair("B-ADDRESS",KV));
 }
 {
  ostringstream res; res << P.Vtext(Id,"B-CITY");
  KV.Reset(); KV.Init(2,res.str());
  Temp.Prop.insert(make_pair("B-CITY",KV));
 }
 {
  ostringstream res; res << P.Vtext(Id,"B-POSTCODE");
  KV.Reset(); KV.Init(2,res.str());
  Temp.Prop.insert(make pair("B-POSTCODE",KV));
 }
 {
  ostringstream res; res << P.Vtext(Id,"O-NAME");
  KV.Reset(); KV.Init(2,res.str());
  Temp.Prop.insert(make pair("O-NAME",KV));
 }
 {
  ostringstream res; res << P.Vtext(Id,"O-TELEPHONE");
  KV.Reset(); KV.Init(2,res.str());
  Temp.Prop.insert(make_pair("O-TELEPHONE",KV));
 }
 {
  ostringstream res; res << P.Vtext(Id,"O-ADDRESS");
  KV.Reset(); KV.Init(2,res.str());
  Temp.Prop.insert(make pair("O-ADDRESS",KV));
 }
 {
  ostringstream res; res << P.Vtext(Id,"O-CITY");
  KV.Reset(); KV.Init(2,res.str());
  Temp.Prop.insert(make pair("O-CITY",KV));
 }
 {
  ostringstream res; res << P.Vtext(Id,"O-POSTCODE");
  KV.Reset(); KV.Init(2,res.str());
  Temp.Prop.insert(make_pair("O-POSTCODE",KV));
```

```
  }
  {
   ostringstream res; res << P.Vtext(Id,"C-NAME");
   KV.Reset(); KV.Init(2,res.str());
   Temp.Prop.insert(make_pair("C-NAME",KV));
  }
  {
   ostringstream res; res << P.Vtext(Id,"C-TELEPHONE");
   KV.Reset(); KV.Init(2,res.str());
   Temp.Prop.insert(make pair("C-TELEPHONE",KV));
  }
  {
   ostringstream res; res << P.Vtext(Id,"C-ADDRESS");
   KV.Reset(); KV.Init(2,res.str());
   Temp.Prop.insert(make pair("C-ADDRESS",KV));
  }
  {
   ostringstream res; res << P.Vtext(Id,"C-CITY");
   KV.Reset(); KV.Init(2,res.str());
   Temp.Prop.insert(make_pair("C-CITY",KV));
  }
  {
   ostringstream res; res << P.Vtext(Id,"C-POSTCODE");
   KV.Reset(); KV.Init(2,res.str());
   Temp.Prop.insert(make pair("C-POSTCODE",KV));
  }
  {
   ostringstream res; res << P.Vtext(Id,"INTERFACE");
   KV.Reset(); KV.Init(2,res.str());
   Temp.Prop.insert(make pair("INTERFACE",KV));
  }
  {
   ostringstream res; res << P.Vtext(Id,"INTERFACE-VERSION");
   KV.Reset(); KV.Init(2,res.str());
   Temp.Prop.insert(make pair("INTERFACE-VERSION",KV));
  }
  {
   ostringstream res; res << P.Vtext(Id,"SOFT-COMP-NAME");
   KV.Reset(); KV.Init(2,res.str());
   Temp.Prop.insert(make pair("SOFT-COMP-NAME",KV));
  }
{
   ostringstream res; res << P.Vnum(Id,"BUILDING-AREA");
   KV.Reset(); KV.Init(1,res.str());
   Temp.Prop.insert(make_pair("BUILDING-AREA",KV));
  }
  {
   ostringstream               res;               res                <<
T.TValue("Not Elec Power Fac",1,P.Vint(Id,"NOT-CASE"));
   KV.Reset(); KV.Init(4,res.str());
   Temp.Prop.insert(make_pair("ELEC-POWER-FACTOR",KV));
  }
  {
   ostringstream               res;               res                <<
T.Value("Not Light Metering",1,P.Vint(Id,"NOT-CASE"));
   KV.Reset(); KV.Init(1,res.str());
   Temp.Prop.insert(make_pair("LIGHT-METERING",KV));
  }

//set notional fuel type - OIL (if there is no gas in the actual
building), or GAS otherwise
//unless the main heating fuel in the actual building is a low or
zero carbon heating technology (Biogas, Biomass, Waste Heat, or Heat
pump)
//when GAS must be used in the notional building
  {
   ostringstream res; res << P.Vcod(Id,"NOT-FUEL");
   KV.Reset(); KV.Init(4,res.str());
   Temp.Prop.insert(make_pair("NOT-FUEL",KV));
```

```
 }

// Building rotation in the notional building is always 0 as any
rotation defined in the actual building has
 // already been applied to the orientations of all the walls before
they are copied onto the notional building
 {
  ostringstream res; res << "0";
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("BUILD-ORIENTATION",KV));
 }

 if((P.Vcodid(cId,"BR-TYPE")==5)  ||  (P.Vcodid(cId,"EPC-TYPE")==6))  {
{
    ostringstream res; res << P.Vnum(Id,"BUILDING-VOLUME");
    KV.Reset(); KV.Init(1,res.str());
    Temp.Prop.insert(make_pair("BUILDING-VOLUME",KV));
   }
   {
    ostringstream res; res << P.Vtext(Id,"B-CONS-DATE");
    KV.Reset(); KV.Init(5,res.str());
    Temp.Prop.insert(make pair("B-CONS-DATE",KV));
   }
 } // end if

 // Saving the user input emission factor for district heating in the
actual building
 {
  ostringstream res; res << P.Vnum(Id,"DH-CO2F");
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("DH-CO2F",KV));
 }

 {
  ostringstream res; res << P.Vnum(Id,"MAX-STOREY");
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("MAX-STOREY",KV));
 }

 { // Intermittency calculation method
  ostringstream res; res << P.Vcod(Id,"INTERMIT-CALC");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("INTERMIT-CALC",KV));
 }


  N.vINP.push back(Temp);
}

//  GENERAL::Notional(PROJECTDefaults&  D,  PROJECTTables&  T,  BOC  &P,
BOC&N)

void GENERAL::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC &P) {
  if(func_report) cout << endl << "  " << "GENERAL::Calc";
  if(calc_report) cout << endl << P.vINP[Id].Name;
  fAREA(D, T, P);
  fENERGY_END_USE(D, T, P);
  fENERGY_FUEL_TYPE(D, T, P);
//  fMAIN_FUEL_TYPE(D, T, P);   // JO (14/02/09): Move analysis after
fBUILDING_ENVIRONMENT  to  set  "Other"  if  BUILDING-ENVIRONMENT  is
"Unconditioned"
//  fBUILDING_ENVIRONMENT(D, T, P);

  if (P.Vcod(Id,"ACT-NOT")=="ACT") {
    fENERGY_RENEWABLES(D, T, P);
    fKG_CO2_CHP(D, T, P);
    fKG_CO2_DISP(D, T, P);
  }
  fKG_CO2_END_USE(D, T, P);
```

```
  fKG CO2(D, T, P);
}

void GENERAL::fAREA(PROJECTDefaults& D, PROJECTTables& T, BOC &P) {
  if(func_report) cout << endl << "  " << "GENERAL::fAREA";
  double mAREA=0;
  for(int i=0;i<P.vINP.size();i++)
    if(P.vINP[i].Type=="HVAC-SYSTEM")
      mAREA+=P.Vnum(i,"AREA")*P.Vnum(i,"MULTIPLIER");
 {
  ostringstream res; res << mAREA;
  P.ModKey(Id,"AREA",res.str());
 }
}

void GENERAL::fENERGY END USE(PROJECTDefaults& D, PROJECTTables& T,
BOC& P) {
  if(func report) cout << endl << "  " << "GENERAL::fENERGY END USE";
//calculates project energy by end use and project total energy
//also writes KG-CO2 for each end use by summing up figures from HVAC
AND DHW (annual figures only)

 double mKWH HEAT M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH HEAT=0.0;
 double mKWH COOL M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH COOL=0.0;
 double mKWH AUX M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH AUX=0.0;
 double mKWH EQUP M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH EQUP=0.0;
 double mKWH LI M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH LI=0.0;
 double mKWH DHW M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH DHW=0.0;

 //v3.0.a dev R09 CD added price weighted energy for each end use
 //using factor from table tFuel Price Weighting (depends on relative
prices of fuels)
 //used in calculation of energy impact for recommendations
 double mKWH HEAT PRICE WEIGHTED=0.0;
 double mKWH COOL PRICE WEIGHTED=0.0;
 double mKWH_AUX_PRICE_WEIGHTED=0.0;
 double mKWH LI PRICE WEIGHTED=0.0;
 double mKWH DHW PRICE WEIGHTED=0.0;

 double mCO2 HEAT=0.0;
 double mCO2 COOL=0.0;
 double mCO2_AUX=0.0;
 double mCO2 LI=0.0;
 double mCO2 DHW=0.0;

 deque<double> mKWH HEAT M HVAC;
 deque<double> mKWH COOL M HVAC;
 deque<double> mKWH_AUX_M_HVAC;
 deque<double> mKWH EQUP M HVAC;
 deque<double> mKWH LI M HVAC;

 // 10 April 2010 JO: Shell and Core new approach. Not counting
energy for systems within catch.
 for(int i=0;i<12;i++) {
   mKWH HEAT M HVAC[i]=0.0;
   mKWH COOL M HVAC[i]=0.0;
   mKWH_AUX_M_HVAC[i]=0.0;
   mKWH EQUP M HVAC[i]=0.0;
   mKWH LI M HVAC[i]=0.0;
 }

  deque<double> mKWH DHW SEC PUMP M;
  deque<double> mKWH_SES_PUMP_M;
```

```
  deque<double> mKWH TSC M;

  for(int i=0;i<P.vINP.size();i++) {
    if(P.vINP[i].Type=="HVAC-SYSTEM") {
      mKWH_HEAT_M_HVAC=P.Vlnum_Mul(i,"KWH-HEAT-
M",P.Vnum(i,"MULTIPLIER"));
      mKWH COOL M HVAC=P.Vlnum Mul(i,"KWH-COOL-
M",P.Vnum(i,"MULTIPLIER"));
      mKWH AUX M HVAC=P.Vlnum Mul(i,"KWH-AUX-
M",P.Vnum(i,"MULTIPLIER"));
      mKWH_EQUP_M_HVAC=P.Vlnum_Mul(i,"KWH-EQUP-
M",P.Vnum(i,"MULTIPLIER"));
      mKWH LI M HVAC=P.Vlnum Mul(i,"KWH-LI-
M",P.Vnum(i,"MULTIPLIER"));

      mCO2 HEAT+=P.Vnum(i,"KG-CO2-HEAT")*P.Vnum(i,"MULTIPLIER");
      mCO2_COOL+=P.Vnum(i,"KG-CO2-COOL")*P.Vnum(i,"MULTIPLIER");
      mCO2 AUX+=P.Vnum(i,"KG-CO2-AUX")*P.Vnum(i,"MULTIPLIER");
      mCO2 LI+=P.Vnum(i,"KG-CO2-LI")*P.Vnum(i,"MULTIPLIER");

      for (int j=0;j<12;j++)   {
        mKWH HEAT M[j]+=mKWH HEAT M HVAC[j];
        mKWH HEAT+=mKWH HEAT M HVAC[j];
        //v3.0.a dev R09 CD added price weighted energy for each end
use
        //using factor from table tFuel Price Weighting (depends on
relative prices of fuels)
        //used in calculation of energy impact for recommendations

mKWH HEAT PRICE WEIGHTED+=mKWH HEAT M HVAC[j]*T.Value("tFuel Price We
ighting",1,P.Vcodid(i,"FUEL-TYPE"));
        mKWH COOL M[j]+=mKWH COOL M HVAC[j];
        mKWH COOL+=mKWH COOL M HVAC[j];
        //v3.0.a dev R09 CD added price weighted energy for each end
use
        // In the next line, the codid of the cooling fuel type is
matched to the codid of the full list of fuel types
        // (as in heating) so that the correct values of emission
factors, primary energy factors, etc, can be retrieved from
        // the tables.

mKWH_COOL_PRICE_WEIGHTED+=mKWH_COOL_M_HVAC[j]*T.Value("tFuel_Price_We
ighting",1,(int)T.Value("tFUEL TYPE COOL",1,P.Vcodid(i,"FUEL-TYPE-
COOL")));

        mKWH AUX M[j]+=mKWH AUX M HVAC[j];
        mKWH AUX+=mKWH AUX M HVAC[j];
        //v3.0.a dev R09 CD added price weighted energy for each end
use

mKWH_AUX_PRICE_WEIGHTED+=mKWH_AUX_M_HVAC[j]*T.Value("tFuel_Price_Weig
hting",1,(int)D.Value("dELEC CODID"));
//          mKWH EQUP M HVAC[j]=0.0; //Added to eliminate EQUIP from
CO2 emissions
        mKWH EQUP M[j]+=mKWH EQUP M HVAC[j];
        mKWH EQUP+=mKWH EQUP M HVAC[j];
        mKWH_LI_M[j]+=mKWH_LI_M_HVAC[j];
        mKWH LI+=mKWH LI M HVAC[j];
        //v3.0.a dev R09 CD added price weighted energy for each end
use

mKWH LI PRICE WEIGHTED+=mKWH LI M HVAC[j]*T.Value("tFuel Price Weight
ing",1,(int)D.Value("dELEC_CODID"));
      }
    }
    else if(P.vINP[i].Type=="DHW-GENERATOR") {
//                        mKWH DHW SEC PUMP M=P.Vlnum Mul(i,"SEC-PUMP-
ENERGY",P.Vnum(i,"MULTIPLIER")); CORRECTED in v1.1.b
      // Check if the hot water system is actually serving any zones
```

```
before taking account of its secondary circulation energy - AN
(28/01/2009)
      if (P.Vnum(i,"AREA-SERVED")!=0.0) {
        mKWH DHW SEC PUMP M=P.Vlnum(i,"SEC-PUMP-ENERGY");
        for (int j=0;j<12;j++)   {
          mKWH AUX M[j]+=mKWH DHW SEC PUMP M[j];
          mKWH AUX+=mKWH DHW SEC PUMP M[j];

mKWH AUX PRICE WEIGHTED+=mKWH DHW SEC PUMP M[j]*T.Value("tFuel Price
Weighting",1,(int)D.Value("dELEC CODID"));
        } // end for
      } // end if
    } // end if
    else if(P.vINP[i].Type=="SES") {
      if (P.Vnum(i,"AREA-SERVED")!=0.0) {
        mKWH SES PUMP M=P.Vlnum Mul(i,"KWH-SES-AUX-
M",P.Vnum(i,"MULTIPLIER"));
        for (int j=0;j<12;j++)   {
          mKWH AUX M[j]+=mKWH SES PUMP M[j];
          mKWH_AUX+=mKWH_SES_PUMP_M[j];

mKWH AUX PRICE WEIGHTED+=mKWH SES PUMP M[j]*T.Value("tFuel Price Weig
hting",1,(int)D.Value("dELEC CODID"));
        } // end for
      } // end if
    } // end if
    else if(P.vINP[i].Type=="TSC") {
      mKWH TSC M=P.Vlnum(i,"KWH-AUX");
      for (int j=0;j<12;j++)   {
        mKWH AUX M[j]+=mKWH TSC M[j];
        mKWH AUX+=mKWH TSC M[j];

mKWH AUX PRICE WEIGHTED+=mKWH TSC M[j]*T.Value("tFuel Price Weighting
",1,(int)D.Value("dELEC CODID"));
      } // end for
    } // end if

  } // end for


 {
  ostringstream res; res << mCO2_HEAT;
  P.ModKey(Id,"KG-CO2-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH HEAT;
  P.ModKey(Id,"KWH-HEAT",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH HEAT M[i] << ", ";
  res << mKWH HEAT M[11] << " }";
  P.ModKey(Id,"KWH-HEAT-M",res.str());
 }
 {
  ostringstream res; res << mKWH_HEAT*3.6;
  P.ModKey(Id,"Q-HEAT",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_HEAT_M[i]*3.6 << ", ";
  res << mKWH HEAT M[11]*3.6 << " }";
  P.ModKey(Id,"Q-HEAT-M",res.str());
 }
 {
  ostringstream res; res << mCO2 COOL;
  P.ModKey(Id,"KG-CO2-COOL",res.str());
```

```
 }
 {
  ostringstream res; res << mKWH COOL;
  P.ModKey(Id,"KWH-COOL",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH COOL M[i] << ", ";
  res << mKWH COOL M[11] << " }";
  P.ModKey(Id,"KWH-COOL-M",res.str());
 }
 {
  ostringstream res; res << mKWH_COOL*3.6;
  P.ModKey(Id,"Q-COOL",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_COOL_M[i]*3.6 << ", ";
  res << mKWH COOL M[11]*3.6 << " }";
  P.ModKey(Id,"Q-COOL-M",res.str());
 }
 {
  ostringstream res; res << mCO2 AUX;
  P.ModKey(Id,"KG-CO2-AUX",res.str());
 }
 {
  ostringstream res; res << mKWH AUX;
  P.ModKey(Id,"KWH-AUX",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH AUX M[i] << ", ";
  res << mKWH AUX M[11] << " }";
  P.ModKey(Id,"KWH-AUX-M",res.str());
 }
 {
  ostringstream res; res << mKWH AUX*3.6;
  P.ModKey(Id,"Q-AUX",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH AUX M[i]*3.6 << ", ";
  res << mKWH AUX M[11]*3.6 << " }";
  P.ModKey(Id,"Q-AUX-M",res.str());
 }
 {
  ostringstream res; res << mKWH_EQUP;
  P.ModKey(Id,"KWH-EQUP",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_EQUP_M[i] << ", ";
  res << mKWH EQUP M[11] << " }";
  P.ModKey(Id,"KWH-EQUP-M",res.str());
 }
 {
  ostringstream res; res << mKWH EQUP*3.6;
  P.ModKey(Id,"Q-EQUP",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH EQUP M[i]*3.6 << ", ";
  res << mKWH_EQUP_M[11]*3.6 << " }";
```

```
  P.ModKey(Id,"Q-EQUP-M",res.str());
 }
 {
  ostringstream res; res << mKWH LI;
  P.ModKey(Id,"KWH-LI",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH LI M[i] << ", ";
  res << mKWH_LI_M[11] << " }";
  P.ModKey(Id,"KWH-LI-M",res.str());
 }
 {
  ostringstream res; res << mCO2 LI;
  P.ModKey(Id,"KG-CO2-LI",res.str());
 }
 {
  ostringstream res; res << mKWH LI*3.6;
  P.ModKey(Id,"Q-LI",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH LI M[i]*3.6 << ", ";
  res << mKWH LI M[11]*3.6 << " }";
  P.ModKey(Id,"Q-LI-M",res.str());
 }

 deque<double> mKWH DHW M DHW;

 for(int i=0;i<P.vINP.size();i++)
   if(P.vINP[i].Type=="DHW-GENERATOR") {
      mCO2 DHW+=P.Vnum(i,"KG-CO2-DHW");
//                                mKWH DHW M DHW=P.Vlnum Mul(i,"KWH-DHW-
M",P.Vnum(i,"MULTIPLIER"));
     mKWH DHW M DHW=P.Vlnum(i,"KWH-DHW-M");
     for (int j=0;j<12;j++) {
       mKWH_DHW_M[j]+=mKWH_DHW_M_DHW[j];
       mKWH DHW+=mKWH DHW M DHW[j];

mKWH_DHW_PRICE_WEIGHTED+=mKWH_DHW_M_DHW[j]*T.Value("tFuel_Price_Weigh
ting",1,P.Vcodid(i,"FUEL-TYPE"));
     }
   }

 {
  ostringstream res; res << mCO2_DHW;
  P.ModKey(Id,"KG-CO2-DHW",res.str());
 }
 {
  ostringstream res; res << mKWH DHW;
  P.ModKey(Id,"KWH-DHW",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH DHW M[i] << ", ";
  res << mKWH DHW M[11] << " }";
  P.ModKey(Id,"KWH-DHW-M",res.str());
 }
 {
  ostringstream res; res << mKWH_DHW*3.6;
  P.ModKey(Id,"Q-DHW",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_DHW_M[i]*3.6 << ", ";
```

```
  res << mKWH DHW M[11]*3.6 << " }";
  P.ModKey(Id,"Q-DHW-M",res.str());
 }

  //total building energy:
  double mKWH TOTAL=0.0;
  //v3.0.a dev R09 CD added price weighted energy for each end use
  //using factor from table tFuel Price Weighting (depends on
relative prices of fuels)
  //used in calculation of energy impact for recommendations
  double mKWH_TOTAL_PRICE_WEIGHTED=0.0;

  mKWH TOTAL=mKWH HEAT+mKWH COOL+mKWH AUX+mKWH LI+mKWH DHW;

mKWH TOTAL PRICE WEIGHTED=mKWH HEAT PRICE WEIGHTED+mKWH COOL PRICE WE
IGHTED+mKWH AUX PRICE WEIGHTED+mKWH LI PRICE WEIGHTED+mKWH DHW PRICE
WEIGHTED;
 {
  ostringstream res; res << mKWH TOTAL;
  P.ModKey(Id,"KWH-TOTAL",res.str());
 }
 {
  ostringstream res; res << mKWH TOTAL;
  P.ModKey(Id,"KWH-JER",res.str());
 }
 {
  ostringstream res; res << mKWH TOTAL*3.6;
  P.ModKey(Id,"Q-TOTAL",res.str());
 }
 {
  ostringstream res; res << mKWH TOTAL PRICE WEIGHTED*3.6;
  P.ModKey(Id,"Q-TOTAL-P-WEIGHTED",res.str());
 }

}

void GENERAL::fENERGY FUEL TYPE(PROJECTDefaults& D, PROJECTTables& T,
BOC& P) {
  if(func_report)    cout    <<    endl    <<    "         "    <<
"GENERAL::fENERGY FUEL TYPE";
//energy used by fuel type

 double mKWH NATGAS M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH NATGAS=0.0;
 double mKWH_LPG_M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH LPG=0.0;
 double mKWH BIOGAS M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH_BIOGAS=0.0;
 double mKWH OIL M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH OIL=0.0;
 double mKWH_COAL_M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH COAL=0.0;
 double mKWH ANTHRACITE M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH_ANTHRACITE=0.0;
 double mKWH SMOKELESS M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH SMOKELESS=0.0;
 double mKWH_DUELFUEL_M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH DUELFUEL=0.0;
 double mKWH BIOMASS M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH_BIOMASS=0.0;
 double mKWH SUPELEC M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH SUPELEC=0.0;
 double mKWH_WASTEHEAT_M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH WASTEHEAT=0.0;
 double mKWH DISTRICT HEATING M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH_DISTRICT_HEATING=0.0;

double mKWH NATGAS heat=0.0;
 double mKWH_LPG_heat=0.0;
```

```
 double mKWH BIOGAS heat=0.0;
 double mKWH OIL heat=0.0;
 double mKWH COAL heat=0.0;
 double mKWH ANTHRACITE heat=0.0;
 double mKWH_SMOKELESS_heat=0.0;
 double mKWH DUELFUEL heat=0.0;
 double mKWH BIOMASS heat=0.0;
 double mKWH_SUPELEC_heat=0.0;
 double mKWH WASTEHEAT heat=0.0;
 double mKWH DISTRICT HEATING heat=0.0;

 deque<double> mKWH NATGAS M HVAC;
 deque<double> mKWH LPG M HVAC;
 deque<double> mKWH_BIOGAS_M_HVAC;
 deque<double> mKWH OIL M HVAC;
 deque<double> mKWH COAL M HVAC;
 deque<double> mKWH_ANTHRACITE_M_HVAC;
 deque<double> mKWH SMOKELESS M HVAC;
 deque<double> mKWH DUELFUEL M HVAC;
 deque<double> mKWH_BIOMASS_M_HVAC;
 deque<double> mKWH SUPELEC M HVAC;
 deque<double> mKWH WASTEHEAT M HVAC;
 deque<double> mKWH DISTRICT HEATING M HVAC;

double mKWH NATGAS PRIM IR=0.0;
 double mKWH LPG PRIM IR=0.0;
 double mKWH BIOGAS PRIM IR=0.0;
 double mKWH OIL PRIM IR=0.0;
 double mKWH COAL PRIM IR=0.0;
 double mKWH ANTHRACITE PRIM IR=0.0;
 double mKWH SMOKELESS PRIM IR=0.0;
 double mKWH DUELFUEL PRIM IR=0.0;
 double mKWH BIOMASS PRIM IR=0.0;
 double mKWH SUPELEC PRIM IR=0.0;
 double mKWH WASTEHEAT PRIM IR=0.0;
 double mKWH DISTRICT HEATING PRIM IR=0.0;
double mKWH PRIM IR=0.0;

double mKWH_NATGAS_PRIM_JER=0.0;
 double mKWH LPG PRIM JER=0.0;
 double mKWH BIOGAS PRIM JER=0.0;
 double mKWH_OIL_PRIM_JER=0.0;
 double mKWH COAL PRIM JER=0.0;
 double mKWH ANTHRACITE PRIM JER=0.0;
 double mKWH_SMOKELESS_PRIM_JER=0.0;
 double mKWH DUELFUEL PRIM JER=0.0;
 double mKWH BIOMASS PRIM JER=0.0;
 double mKWH_SUPELEC_PRIM_JER=0.0;
 double mKWH WASTEHEAT PRIM JER=0.0;
 double mKWH DISTRICT HEATING PRIM JER=0.0;
double mKWH_PRIM_JER=0.0;


 double iMultiplier;


  deque<double> mKWH_DHW_SEC_PUMP_M;
  double mKWH SEC M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
  double mKWH SEC=0;

  // adding the SES pump energy if applicable
  deque<double> mKWH SES PUMP M;
  double mKWH_SES_M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
  double mKWH SES=0;

  // adding the TSC pump energy if applicable
  deque<double> mKWH TSC AUX M;
  double mKWH TSC M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
  double mKWH_TSC=0;
```

```
 for(int i=0;i<P.vINP.size();i++)
   switch(P.vINP[i].Ntype) {
     case 1: {
     //HVAC
                iMultiplier=P.Vnum(i,"MULTIPLIER");
                mKWH NATGAS M HVAC=P.Vlnum Mul(i,"KWH-NATGAS-
M",iMultiplier);
                mKWH LPG M HVAC=P.Vlnum Mul(i,"KWH-LPG-
M",iMultiplier);
                mKWH_BIOGAS_M_HVAC=P.Vlnum_Mul(i,"KWH-BIOGAS-
M",iMultiplier);
                mKWH OIL M HVAC=P.Vlnum Mul(i,"KWH-OIL-
M",iMultiplier);
                mKWH COAL M HVAC=P.Vlnum Mul(i,"KWH-COAL-
M",iMultiplier);
                mKWH_ANTHRACITE_M_HVAC=P.Vlnum_Mul(i,"KWH-ANTHRACITE-
M",iMultiplier);
                mKWH SMOKELESS M HVAC=P.Vlnum Mul(i,"KWH-SMOKELESS-
M",iMultiplier);
                mKWH DUELFUEL M HVAC=P.Vlnum Mul(i,"KWH-DUELFUEL-
M",iMultiplier);
                mKWH BIOMASS M HVAC=P.Vlnum Mul(i,"KWH-BIOMASS-
M",iMultiplier);
                mKWH SUPELEC M HVAC=P.Vlnum Mul(i,"KWH-SUPELEC-
M",iMultiplier);
                mKWH WASTEHEAT M HVAC=P.Vlnum Mul(i,"KWH-WASTEHEAT-
M",iMultiplier);
                mKWH DISTRICT HEATING M HVAC=P.Vlnum Mul(i,"KWH-
DISTRICT-HEATING-M",iMultiplier);

                // New parameters to save the annual heating fuel ONLY
(space heating and DHW only).
                mKWH NATGAS heat+=P.Vnum(i,"KWH-NATGAS-
HEAT")*iMultiplier;
                mKWH LPG heat+=P.Vnum(i,"KWH-LPG-HEAT")*iMultiplier;
                mKWH BIOGAS heat+=P.Vnum(i,"KWH-BIOGAS-
HEAT")*iMultiplier;
                mKWH_OIL_heat+=P.Vnum(i,"KWH-OIL-HEAT")*iMultiplier;
                mKWH COAL heat+=P.Vnum(i,"KWH-COAL-HEAT")*iMultiplier;
                mKWH ANTHRACITE heat+=P.Vnum(i,"KWH-ANTHRACITE-
HEAT")*iMultiplier;
                mKWH SMOKELESS heat+=P.Vnum(i,"KWH-SMOKELESS-
HEAT")*iMultiplier;
                mKWH_DUELFUEL_heat+=P.Vnum(i,"KWH-DUELFUEL-
HEAT")*iMultiplier;
                mKWH BIOMASS heat+=P.Vnum(i,"KWH-BIOMASS-
HEAT")*iMultiplier;
                mKWH SUPELEC heat+=P.Vnum(i,"KWH-SUPELEC-
HEAT")*iMultiplier;
                mKWH_WASTEHEAT_heat+=P.Vnum(i,"KWH-WASTEHEAT-
HEAT")*iMultiplier;
                mKWH DISTRICT HEATING heat+=P.Vnum(i,"KWH-DISTRICT-
HEATING-HEAT")*iMultiplier;
//                          mKWH HEATPUMP heat+=P.Vnum(i,"KWH-HEATPUMP-
HEAT")*iMultiplier;

                // Include the energy used by any DHW generators that
are served by the HVAC if its heat source is a heat pump
//                 deque<int> mDHW_SERVED_REF=P.Vlrefid(i,"DHW-SERVED-
REF");
//                            if  ((P.Vcodid(i,"HEAT-SOURCE")==12)  ||
(P.Vcodid(i,"HEAT-SOURCE")==13) || (P.Vcodid(i,"HEAT-SOURCE")==18) ||
(P.Vcodid(i,"HEAT-SOURCE")==19)) // if the heat source of the HVAC is
a heat pump
//                   for (int j=0;j< mDHW_SERVED_REF.size();j++)
//
mKWH HEATPUMP heat+=P.Vnum(mDHW SERVED REF[j],"KWH-DHW");
```

```
                for (int j=0;j<12;j++)   {
                  mKWH NATGAS M[j]+=mKWH NATGAS M HVAC[j];
                  mKWH NATGAS+=mKWH NATGAS M HVAC[j];
                  mKWH LPG M[j]+=mKWH LPG M HVAC[j];
                  mKWH_LPG+=mKWH_LPG_M_HVAC[j];
                  mKWH BIOGAS M[j]+=mKWH BIOGAS M HVAC[j];
                  mKWH BIOGAS+=mKWH BIOGAS M HVAC[j];
                  mKWH_OIL_M[j]+=mKWH_OIL_M_HVAC[j];
                  mKWH OIL+=mKWH OIL M HVAC[j];
                  mKWH COAL M[j]+=mKWH COAL M HVAC[j];
                  mKWH_COAL+=mKWH_COAL_M_HVAC[j];
                  mKWH ANTHRACITE M[j]+=mKWH ANTHRACITE M HVAC[j];
                  mKWH ANTHRACITE+=mKWH ANTHRACITE M HVAC[j];
                  mKWH_SMOKELESS_M[j]+=mKWH_SMOKELESS_M_HVAC[j];
                  mKWH SMOKELESS+=mKWH SMOKELESS M HVAC[j];
                  mKWH DUELFUEL M[j]+=mKWH DUELFUEL M HVAC[j];
                  mKWH_DUELFUEL+=mKWH_DUELFUEL_M_HVAC[j];
                  mKWH BIOMASS M[j]+=mKWH BIOMASS M HVAC[j];
                  mKWH BIOMASS+=mKWH BIOMASS M HVAC[j];
                  mKWH_SUPELEC_M[j]+=mKWH_SUPELEC_M_HVAC[j];
                  mKWH SUPELEC+=mKWH SUPELEC M HVAC[j];
                  mKWH WASTEHEAT M[j]+=mKWH WASTEHEAT M HVAC[j];
                  mKWH WASTEHEAT+=mKWH WASTEHEAT M HVAC[j];

mKWH DISTRICT HEATING M[j]+=mKWH DISTRICT HEATING M HVAC[j];

mKWH DISTRICT HEATING+=mKWH DISTRICT HEATING M HVAC[j];
                }
     }
     break;
     case 12: {
     //DHW GENERATOR
//                  deque<double> mKWH DHW FUEL=P.Vlnum Mul(i,"KWH-DHW-
M",P.Vnum(i,"MULTIPLIER"));
                deque<double> mKWH DHW FUEL=P.Vlnum(i,"KWH-DHW-M");
                for (int m=0;m<12;m++) {
                  switch (P.Vcodid(i,"FUEL-TYPE"))   {
                  //nat gas
                  case 1: {
                    mKWH NATGAS M[m]+=mKWH DHW FUEL[m];
                    mKWH NATGAS+=mKWH DHW FUEL[m];
                    mKWH_NATGAS_heat+=mKWH_DHW_FUEL[m];
                    }
                    break;
                  //LPG
                  case 2: {
                    mKWH LPG M[m]+=mKWH DHW FUEL[m];
                    mKWH_LPG+=mKWH_DHW_FUEL[m];
                    mKWH LPG heat+=mKWH DHW FUEL[m];
                    }
                    break;
                  //biogas
                  case 3: {
                    mKWH_BIOGAS_M[m]+=mKWH_DHW_FUEL[m];
                    mKWH BIOGAS+=mKWH DHW FUEL[m];
                    mKWH BIOGAS heat+=mKWH DHW FUEL[m];
                    }
                    break;
                  //oil
                  case 4: {
                    mKWH OIL M[m]+=mKWH DHW FUEL[m];
                    mKWH OIL+=mKWH DHW FUEL[m];
                    mKWH_OIL_heat+=mKWH_DHW_FUEL[m];
                    }
                    break;
                  //coal
                  case 5: {
                    mKWH COAL M[m]+=mKWH DHW FUEL[m];
                    mKWH_COAL+=mKWH_DHW_FUEL[m];
```

```
                    mKWH COAL heat+=mKWH DHW FUEL[m];
                    }
                    break;
                   //anthracite
                  case 6: {
                    mKWH ANTHRACITE M[m]+=mKWH DHW FUEL[m];
                    mKWH ANTHRACITE+=mKWH DHW FUEL[m];
                    mKWH_ANTHRACITE_heat+=mKWH_DHW_FUEL[m];
                    }
                    break;
                   //smokeless fuel
                  case 7: {
                    mKWH SMOKELESS M[m]+=mKWH DHW FUEL[m];
                    mKWH_SMOKELESS+=mKWH_DHW_FUEL[m];
                    mKWH SMOKELESS heat+=mKWH DHW FUEL[m];
                    }
                    break;
                   //duel fuel
                  case 8: {
                    mKWH_DUELFUEL_M[m]+=mKWH_DHW_FUEL[m];
                    mKWH DUELFUEL+=mKWH DHW FUEL[m];
                    mKWH DUELFUEL heat+=mKWH DHW FUEL[m];
                    }
                    break;
                  //biomass
                  case 9: {
                    mKWH BIOMASS M[m]+=mKWH DHW FUEL[m];
                    mKWH BIOMASS+=mKWH DHW FUEL[m];
                    mKWH BIOMASS heat+=mKWH DHW FUEL[m];
                    }
                    break;
                  //grid supplied electricity
                  case 10: {
                    mKWH SUPELEC M[m]+=mKWH DHW FUEL[m];
                    mKWH SUPELEC+=mKWH DHW FUEL[m];
                    mKWH SUPELEC heat+=mKWH DHW FUEL[m];
                    }
                    break;
                  //waste heat
                  case 12: {
                    mKWH WASTEHEAT M[m]+=mKWH DHW FUEL[m];
                    mKWH_WASTEHEAT+=mKWH_DHW_FUEL[m];
                    mKWH WASTEHEAT heat+=mKWH DHW FUEL[m];
                    }
                    break;
                  //district heating
                  case 13: {
                    mKWH_DISTRICT_HEATING_M[m]+=mKWH_DHW_FUEL[m];
                    mKWH DISTRICT HEATING+=mKWH DHW FUEL[m];
                    mKWH DISTRICT HEATING heat+=mKWH DHW FUEL[m];
                    }
                    break;
                  }
//                   if (P.Vcod(i,"HEAT-GEN-TYPE")=="Heat pump")  // if
the generator type for DHW is a heat pump
//                        mKWH HEATPUMP heat+=mKWH DHW FUEL[m];

                 }
                 if (P.Vnum(i,"AREA-SERVED")!=0.0) {
                   mKWH_DHW_SEC_PUMP_M=P.Vlnum(i,"SEC-PUMP-ENERGY");
                   for (int j=0;j<12;j++)   {
                     mKWH SEC M[j]+=mKWH DHW SEC PUMP M[j];
                     mKWH_SEC+=mKWH_DHW_SEC_PUMP_M[j];
                   } // end for
                 } // end if

     }
     break;
     case 20: {
```

```
     //CHP
                deque<double> mKWH HEAT FUEL=P.Vlnum(i,"KWH-HEAT-M");
                deque<double>        mKWH HEATCOOL FUEL=P.Vlnum(i,"KWH-
HEATCOOL-M");
                deque<double> mKWH_DHW_FUEL=P.Vlnum(i,"KWH-DHW-M");
                double mKWH FUEL[12]= { 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0 };
                for (int m=0;m<12;m++) {

mKWH FUEL[m]=mKWH HEAT FUEL[m]+mKWH HEATCOOL FUEL[m]+mKWH DHW FUEL[m]
;
                  switch (P.Vcodid(i,"FUEL-TYPE"))   {
                  //nat gas
                  case 1: {
                    mKWH NATGAS M[m]+=mKWH FUEL[m];
                    mKWH NATGAS+=mKWH FUEL[m];
                    mKWH_NATGAS_heat+=mKWH_FUEL[m];
                    }
                    break;
                  //LPG
                  case 2: {
                    mKWH LPG M[m]+=mKWH FUEL[m];
                    mKWH LPG+=mKWH FUEL[m];
                    mKWH LPG heat+=mKWH FUEL[m];
                    }
                    break;
                  //biogas
                  case 3: {
                    mKWH BIOGAS M[m]+=mKWH FUEL[m];
                    mKWH BIOGAS+=mKWH FUEL[m];
                    mKWH BIOGAS heat+=mKWH FUEL[m];
                    }
                    break;
                  //oil
                  case 4: {
                    mKWH OIL M[m]+=mKWH FUEL[m];
                    mKWH OIL+=mKWH FUEL[m];
                    mKWH OIL heat+=mKWH FUEL[m];
                    }
                    break;
                  //coal
                  case 5: {
                    mKWH COAL M[m]+=mKWH FUEL[m];
                    mKWH COAL+=mKWH FUEL[m];
                    mKWH_COAL_heat+=mKWH_FUEL[m];
                    }
                    break;
                   //anthracite
                  case 6: {
                    mKWH ANTHRACITE M[m]+=mKWH FUEL[m];
                    mKWH_ANTHRACITE+=mKWH_FUEL[m];
                    mKWH ANTHRACITE heat+=mKWH FUEL[m];
                    }
                    break;
                   //smokeless fuel
                  case 7: {
                    mKWH_SMOKELESS_M[m]+=mKWH_FUEL[m];
                    mKWH SMOKELESS+=mKWH FUEL[m];
                    mKWH SMOKELESS heat+=mKWH FUEL[m];
                    }
                    break;
                   //duel fuel
                  case 8: {
                    mKWH DUELFUEL M[m]+=mKWH FUEL[m];
                    mKWH DUELFUEL+=mKWH FUEL[m];
                    mKWH_DUELFUEL_heat+=mKWH_FUEL[m];
                    }
                    break;
                  //biomass
```

```
                  case 9: {
                    mKWH BIOMASS M[m]+=mKWH FUEL[m];
                    mKWH BIOMASS+=mKWH FUEL[m];
                    mKWH BIOMASS heat+=mKWH FUEL[m];
                    }
                    break;
                  //grid supplied electricity
                  case 10: {
                    mKWH SUPELEC M[m]+=mKWH FUEL[m];
                    mKWH SUPELEC+=mKWH FUEL[m];
                    mKWH_SUPELEC_heat+=mKWH_FUEL[m];
                    }
                    break;
                  //waste heat
                  case 12: {
                    mKWH WASTEHEAT M[m]+=mKWH FUEL[m];
                    mKWH_WASTEHEAT+=mKWH_FUEL[m];
                    mKWH WASTEHEAT heat+=mKWH FUEL[m];
                    }
                    break;
                  //district heating
                  case 13: {
                    mKWH DISTRICT HEATING M[m]+=mKWH FUEL[m];
                    mKWH DISTRICT HEATING+=mKWH FUEL[m];
                    mKWH DISTRICT HEATING heat+=mKWH FUEL[m];
                    }
                    break;
                  }
                }
     }
     break;
     case 13: {
     //SES
               if (P.Vnum(i,"AREA-SERVED")!=0.0) {
                 mKWH SES PUMP M=P.Vlnum Mul(i,"KWH-SES-AUX-
M",P.Vnum(i,"MULTIPLIER"));
                 for (int j=0;j<12;j++)   {
                   mKWH SES M[j]+=mKWH SES PUMP M[j];
                   mKWH_SES+=mKWH_SES_PUMP_M[j];
                 } // end for
               } // end if


     }
     break;
     case 24: {
     //TSC
                 mKWH_TSC_AUX_M=P.Vlnum(i,"KWH-AUX");
                 for (int j=0;j<12;j++)   {
                   mKWH TSC M[j]+=mKWH TSC AUX M[j];
                   mKWH_TSC+=mKWH_TSC_AUX_M[j];
                 } // end for


     }
     break;
  }

  for(int i=0;i<12;i++) {
    mKWH_SUPELEC_M[i]+=mKWH_SEC_M[i];
    mKWH SUPELEC+=mKWH SEC M[i];
  }

  for(int i=0;i<12;i++) {
    mKWH SUPELEC M[i]+=mKWH SES M[i];
    mKWH_SUPELEC+=mKWH_SES_M[i];
  }

  for(int i=0;i<12;i++) {
```

```
    mKWH SUPELEC M[i]+=mKWH TSC M[i];
    mKWH SUPELEC+=mKWH TSC M[i];
  }


 {
  ostringstream res; res << mKWH NATGAS;
  P.ModKey(Id,"KWH-NATGAS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH NATGAS M[i] << ", ";
  res << mKWH_NATGAS_M[11] << " }";
  P.ModKey(Id,"KWH-NATGAS-M",res.str());
 }
 {
  ostringstream res; res << mKWH NATGAS*3.6;
  P.ModKey(Id,"Q-NATGAS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH NATGAS M[i]*3.6 << ", ";
  res << mKWH NATGAS M[11]*3.6 << " }";
  P.ModKey(Id,"Q-NATGAS-M",res.str());
 }
 {
  ostringstream res; res << mKWH NATGAS heat*3.6;
  P.ModKey(Id,"Q-NATGAS-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH LPG;
  P.ModKey(Id,"KWH-LPG",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_LPG_M[i] << ", ";
  res << mKWH LPG M[11] << " }";
  P.ModKey(Id,"KWH-LPG-M",res.str());
 }
 {
  ostringstream res; res << mKWH LPG*3.6;
  P.ModKey(Id,"Q-LPG",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH LPG M[i]*3.6 << ", ";
  res << mKWH_LPG_M[11]*3.6 << " }";
  P.ModKey(Id,"Q-LPG-M",res.str());
 }
 {
  ostringstream res; res << mKWH LPG heat*3.6;
  P.ModKey(Id,"Q-LPG-HEAT",res.str());
 }

 {
  ostringstream res; res << mKWH_BIOGAS;
  P.ModKey(Id,"KWH-BIOGAS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_BIOGAS_M[i] << ", ";
  res << mKWH BIOGAS M[11] << " }";
  P.ModKey(Id,"KWH-BIOGAS-M",res.str());
 }
```

```
{
 ostringstream res; res << mKWH BIOGAS*3.6;
 P.ModKey(Id,"Q-BIOGAS",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<11;i++)
   res << mKWH_BIOGAS_M[i]*3.6 << ", ";
 res << mKWH BIOGAS M[11]*3.6 << " }";
 P.ModKey(Id,"Q-BIOGAS-M",res.str());
}
{
 ostringstream res; res << mKWH BIOGAS heat*3.6;
 P.ModKey(Id,"Q-BIOGAS-HEAT",res.str());
}

{
 ostringstream res; res << mKWH OIL;
 P.ModKey(Id,"KWH-OIL",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<11;i++)
   res << mKWH OIL M[i] << ", ";
 res << mKWH OIL M[11] << " }";
 P.ModKey(Id,"KWH-OIL-M",res.str());
}
{
 ostringstream res; res << mKWH OIL*3.6;
 P.ModKey(Id,"Q-OIL",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<11;i++)
   res << mKWH OIL M[i]*3.6 << ", ";
 res << mKWH OIL M[11]*3.6 << " }";
 P.ModKey(Id,"Q-OIL-M",res.str());
}
{
 ostringstream res; res << mKWH OIL heat*3.6;
 P.ModKey(Id,"Q-OIL-HEAT",res.str());
}

{
 ostringstream res; res << mKWH_COAL;
 P.ModKey(Id,"KWH-COAL",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<11;i++)
   res << mKWH_COAL_M[i] << ", ";
 res << mKWH COAL M[11] << " }";
 P.ModKey(Id,"KWH-COAL-M",res.str());
}
{
 ostringstream res; res << mKWH COAL*3.6;
 P.ModKey(Id,"Q-COAL",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<11;i++)
   res << mKWH COAL M[i]*3.6 << ", ";
 res << mKWH_COAL_M[11]*3.6 << " }";
 P.ModKey(Id,"Q-COAL-M",res.str());
}
{
 ostringstream res; res << mKWH COAL heat*3.6;
 P.ModKey(Id,"Q-COAL-HEAT",res.str());
}
```

```
{
 ostringstream res; res << mKWH ANTHRACITE;
 P.ModKey(Id,"KWH-ANTHRACITE",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<11;i++)
   res << mKWH ANTHRACITE M[i] << ", ";
 res << mKWH ANTHRACITE M[11] << " }";
 P.ModKey(Id,"KWH-ANTHRACITE-M",res.str());
}
{
 ostringstream res; res << mKWH_ANTHRACITE*3.6;
 P.ModKey(Id,"Q-ANTHRACITE",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<11;i++)
   res << mKWH_ANTHRACITE_M[i]*3.6 << ", ";
 res << mKWH ANTHRACITE M[11]*3.6 << " }";
 P.ModKey(Id,"Q-ANTHRACITE-M",res.str());
}
{
 ostringstream res; res << mKWH ANTHRACITE heat*3.6;
 P.ModKey(Id,"Q-ANTHRACITE-HEAT",res.str());
}

{
 ostringstream res; res << mKWH SMOKELESS;
 P.ModKey(Id,"KWH-SMOKELESS",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<11;i++)
   res << mKWH SMOKELESS M[i] << ", ";
 res << mKWH SMOKELESS M[11] << " }";
 P.ModKey(Id,"KWH-SMOKELESS-M",res.str());
}
{
 ostringstream res; res << mKWH SMOKELESS*3.6;
 P.ModKey(Id,"Q-SMOKELESS",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<11;i++)
   res << mKWH SMOKELESS M[i]*3.6 << ", ";
 res << mKWH_SMOKELESS_M[11]*3.6 << " }";
 P.ModKey(Id,"Q-SMOKELESS-M",res.str());
}
{
 ostringstream res; res << mKWH SMOKELESS heat*3.6;
 P.ModKey(Id,"Q-SMOKELESS-HEAT",res.str());
}

{
 ostringstream res; res << mKWH_DUELFUEL;
 P.ModKey(Id,"KWH-DUELFUEL",res.str());
}
{
 ostringstream res; res << "{ ";
 for (int i=0;i<11;i++)
   res << mKWH_DUELFUEL_M[i] << ", ";
 res << mKWH DUELFUEL M[11] << " }";
 P.ModKey(Id,"KWH-DUELFUEL-M",res.str());
}
{
 ostringstream res; res << mKWH DUELFUEL*3.6;
 P.ModKey(Id,"Q-DUELFUEL",res.str());
```

```
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_DUELFUEL_M[i]*3.6 << ", ";
  res << mKWH DUELFUEL M[11]*3.6 << " }";
  P.ModKey(Id,"Q-DUELFUEL-M",res.str());
 }
 {
  ostringstream res; res << mKWH DUELFUEL heat*3.6;
  P.ModKey(Id,"Q-DUELFUEL-HEAT",res.str());
 }

 {
  ostringstream res; res << mKWH BIOMASS;
  P.ModKey(Id,"KWH-BIOMASS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH BIOMASS M[i] << ", ";
  res << mKWH BIOMASS M[11] << " }";
  P.ModKey(Id,"KWH-BIOMASS-M",res.str());
 }
 {
  ostringstream res; res << mKWH BIOMASS*3.6;
  P.ModKey(Id,"Q-BIOMASS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH BIOMASS M[i]*3.6 << ", ";
  res << mKWH BIOMASS M[11]*3.6 << " }";
  P.ModKey(Id,"Q-BIOMASS-M",res.str());
 }
 {
  ostringstream res; res << mKWH BIOMASS heat*3.6;
  P.ModKey(Id,"Q-BIOMASS-HEAT",res.str());
 }

 {
  ostringstream res; res << mKWH_SUPELEC;
  P.ModKey(Id,"KWH-SUPELEC",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_SUPELEC_M[i] << ", ";
  res << mKWH SUPELEC M[11] << " }";
  P.ModKey(Id,"KWH-SUPELEC-M",res.str());
 }
 {
  ostringstream res; res << mKWH SUPELEC*3.6;
  P.ModKey(Id,"Q-SUPELEC",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH SUPELEC M[i]*3.6 << ", ";
  res << mKWH_SUPELEC_M[11]*3.6 << " }";
  P.ModKey(Id,"Q-SUPELEC-M",res.str());
 }
 {
  ostringstream res; res << mKWH SUPELEC heat*3.6;
  P.ModKey(Id,"Q-SUPELEC-HEAT",res.str());
 }

 {
  ostringstream res; res << mKWH_WASTEHEAT;
```

```
  P.ModKey(Id,"KWH-WASTEHEAT",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH WASTEHEAT M[i] << ", ";
  res << mKWH WASTEHEAT M[11] << " }";
  P.ModKey(Id,"KWH-WASTEHEAT-M",res.str());
 }
 {
  ostringstream res; res << mKWH_WASTEHEAT*3.6;
  P.ModKey(Id,"Q-WASTEHEAT",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_WASTEHEAT_M[i]*3.6 << ", ";
  res << mKWH WASTEHEAT M[11]*3.6 << " }";
  P.ModKey(Id,"Q-WASTEHEAT-M",res.str());
 }
 {
  ostringstream res; res << mKWH WASTEHEAT heat*3.6;
  P.ModKey(Id,"Q-WASTEHEAT-HEAT",res.str());
 }

 {
  ostringstream res; res << mKWH DISTRICT HEATING;
  P.ModKey(Id,"KWH-DISTRICT-HEATING",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH DISTRICT HEATING M[i] << ", ";
  res << mKWH DISTRICT HEATING M[11] << " }";
  P.ModKey(Id,"KWH-DISTRICT-HEATING-M",res.str());
 }
 {
  ostringstream res; res << mKWH DISTRICT HEATING*3.6;
  P.ModKey(Id,"Q-DISTRICT-HEATING",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH DISTRICT HEATING M[i]*3.6 << ", ";
  res << mKWH_DISTRICT_HEATING_M[11]*3.6 << " }";
  P.ModKey(Id,"Q-DISTRICT-HEATING-M",res.str());
 }
 {
  ostringstream res; res << mKWH DISTRICT HEATING heat*3.6;
  P.ModKey(Id,"Q-DISTRICT-HEATING-HEAT",res.str());
 }

// {
//  ostringstream res; res << mKWH_HEATPUMP_heat*3.6;
//  P.ModKey(Id,"Q-HEATPUMP-HEAT",res.str());
// }


  if((P.Vcodid(cId,"BR-TYPE")==5)  ||  (P.Vcodid(cId,"EPC-TYPE")==6)){
mKWH_NATGAS_PRIM_IR=P.Vnum(Id,"KWH-NATGAS")*T.Value("tPRIM_IR",1,1);
     mKWH LPG PRIM IR=P.Vnum(Id,"KWH-LPG")*T.Value("tPRIM IR",1,2);
     mKWH BIOGAS PRIM IR=P.Vnum(Id,"KWH-
BIOGAS")*T.Value("tPRIM_IR",1,3);
     mKWH OIL PRIM IR=P.Vnum(Id,"KWH-OIL")*T.Value("tPRIM IR",1,4);
     mKWH COAL PRIM IR=P.Vnum(Id,"KWH-COAL")*T.Value("tPRIM IR",1,5);
     mKWH_ANTHRACITE_PRIM_IR=P.Vnum(Id,"KWH-
ANTHRACITE")*T.Value("tPRIM IR",1,6);
     mKWH SMOKELESS PRIM IR=P.Vnum(Id,"KWH-
SMOKELESS")*T.Value("tPRIM_IR",1,7);
```

```
     mKWH DUELFUEL PRIM IR=P.Vnum(Id,"KWH-
DUELFUEL")*T.Value("tPRIM IR",1,8);
     mKWH BIOMASS PRIM IR=P.Vnum(Id,"KWH-
BIOMASS")*T.Value("tPRIM IR",1,9);
     mKWH_SUPELEC_PRIM_IR=P.Vnum(Id,"KWH-
SUPELEC")*T.Value("tPRIM IR",1,10);
     // note that grid-displaced is not included here - it is
calculated separately for renewables in function fENERGY_RENEWABLES
     mKWH WASTEHEAT PRIM IR=P.Vnum(Id,"KWH-
WASTEHEAT")*T.Value("tPRIM IR",1,12);
     mKWH_DISTRICT_HEATING_PRIM_IR=P.Vnum(Id,"KWH-DISTRICT-
HEATING")*T.Value("tPRIM IR",1,13);


mKWH PRIM IR=mKWH NATGAS PRIM IR+mKWH LPG PRIM IR+mKWH BIOGAS PRIM IR
+mKWH OIL PRIM IR+mKWH COAL PRIM IR+mKWH ANTHRACITE PRIM IR+mKWH SMOK
ELESS_PRIM_IR+mKWH_DUELFUEL_PRIM_IR+mKWH_BIOMASS_PRIM_IR+mKWH_SUPELEC
 PRIM IR+mKWH WASTEHEAT PRIM IR+mKWH DISTRICT HEATING PRIM IR;
  }
  else if((P.Vcodid(cId,"BR-TYPE")==6)) { // && P.Vcodid(cId,"JERSEY-
PRIM")==1){  //Jersey only
     mKWH NATGAS PRIM JER=P.Vnum(Id,"KWH-
NATGAS")*T.Value("tPRIM JER",1,1);
     mKWH LPG PRIM JER=P.Vnum(Id,"KWH-LPG")*T.Value("tPRIM JER",1,2);
     mKWH BIOGAS PRIM JER=P.Vnum(Id,"KWH-
BIOGAS")*T.Value("tPRIM JER",1,3);
     mKWH OIL PRIM JER=P.Vnum(Id,"KWH-OIL")*T.Value("tPRIM JER",1,4);
     mKWH COAL PRIM JER=P.Vnum(Id,"KWH-
COAL")*T.Value("tPRIM JER",1,5);
     mKWH ANTHRACITE PRIM JER=P.Vnum(Id,"KWH-
ANTHRACITE")*T.Value("tPRIM JER",1,6);
     mKWH SMOKELESS PRIM JER=P.Vnum(Id,"KWH-
SMOKELESS")*T.Value("tPRIM JER",1,7);
     mKWH DUELFUEL PRIM JER=P.Vnum(Id,"KWH-
DUELFUEL")*T.Value("tPRIM JER",1,8);
     mKWH BIOMASS PRIM JER=P.Vnum(Id,"KWH-
BIOMASS")*T.Value("tPRIM JER",1,9);
     mKWH SUPELEC PRIM JER=P.Vnum(Id,"KWH-
SUPELEC")*T.Value("tPRIM_JER",1,10);
     // note that grid-displaced is not included here - it is
calculated separately for renewables in function fENERGY RENEWABLES
     mKWH_WASTEHEAT_PRIM_JER=P.Vnum(Id,"KWH-
WASTEHEAT")*T.Value("tPRIM JER",1,12);
     mKWH DISTRICT HEATING PRIM JER=P.Vnum(Id,"KWH-DISTRICT-
HEATING")*T.Value("tPRIM_JER",1,13);


mKWH_PRIM_JER=mKWH_NATGAS_PRIM_JER+mKWH_LPG_PRIM_JER+mKWH_BIOGAS_PRIM
 JER+mKWH OIL PRIM JER+mKWH COAL PRIM JER+mKWH ANTHRACITE PRIM JER+mK
WH SMOKELESS PRIM JER+mKWH DUELFUEL PRIM JER+mKWH BIOMASS PRIM JER+mK
WH_SUPELEC_PRIM_JER+mKWH_WASTEHEAT_PRIM_JER+mKWH_DISTRICT_HEATING_PRI
M JER;
  }

//Total primary energy for RoI due to NON-renewables
 {
  ostringstream res; res << mKWH_PRIM_IR;
  P.ModKey(Id,"KWH-PRIM-IR",res.str());
 }

//Total primary energy for Jersey due to NON-renewables
 {
  ostringstream res; res << mKWH_PRIM_JER;
  P.ModKey(Id,"KWH-PRIM-JER",res.str());
 }

}

void GENERAL::fSC_ACTUAL_RATIO(PROJECTTables& T, BOC& P) {
```

```
  if(func report)     cout     <<     endl     <<     "          "     <<
"GENERAL::fSC ACTUAL RATIO";
//Shell and core assumed ratio
  double mAREA ASSUMED=0.0;
  double mAREA_ACTUAL=0.0;

  for(int i=0;i<P.vINP.size();i++) {
    if(P.vINP[i].Ntype==9) {// ZONES
      //if(P.Vcodid(i,"SHELL-ASSUMED")==1)
      if((P.Vcodid(cId,"SHELL-AND-CORE")==1    &&    P.Vcodid(cId,"EPC-
TYPE")==1   &&   P.Vcodid(cId,"BR-STAGE")==2   &&   P.Vcodid(i,"SHELL-
ASSUMED")==1))
        mAREA ASSUMED+=P.Vnum(i,"AREA");
      else
        mAREA ACTUAL+=P.Vnum(i,"AREA");
    }
  }
  double mSC ACTUAL RATIO=1.0;
  if(mAREA ASSUMED+mAREA ACTUAL>0.0)
mSC_ACTUAL_RATIO=mAREA_ACTUAL/(mAREA_ASSUMED+mAREA_ACTUAL);

 {
  ostringstream res; res << mSC ACTUAL RATIO;
  P.ModKey(cId,"SC-ACTUAL-RATIO",res.str());
 }
}

void  GENERAL::fENERGY RENEWABLES(PROJECTDefaults&  D,  PROJECTTables&
T, BOC& P) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"GENERAL::fENERGY RENEWABLES";

  double mKWH DISP M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
  double mKWH DISP=0.0;
  double mQ DISP M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
  double mQ DISP=0.0;
  deque<double> temp;

  double  mKWH_PRIM_IR=P.Vnum(Id,"KWH-PRIM-IR");    //  primary  energy
due to NON-renewables in RoI
  double  mKWH PRIM JER=P.Vnum(Id,"KWH-PRIM-JER");   // primary energy
due to NON-renewables in Jersey
  double  mKWH JER=P.Vnum(Id,"KWH-JER");   // delivered energy due to
NON-renewables in Jersey

 for(int i=0;i<P.vINP.size();i++)
   if(P.vINP[i].Type=="CCHP-GENERATOR") {
     temp=P.Vlnum_Mul(i,"KWH-DISP-M",1);  //P.Vnum(i,"MULTIPLIER"));
     mKWH DISP=P.Vnum(i,"KWH-DISP");      //*P.Vnum(i,"MULTIPLIER");
     for (int j=0;j<12;j++) mKWH DISP M[j]+=temp[j];
   }
   else if(P.vINP[i].Type=="WIND-GENERATOR") {
     temp=P.Vlnum Mul(i,"KWH-DISP-M",P.Vnum(i,"MULTIPLIER"));
     mKWH_DISP+=P.Vnum(i,"KWH-DISP")*P.Vnum(i,"MULTIPLIER");
     for (int j=0;j<12;j++) mKWH DISP M[j]+=temp[j];
   }
   else if(P.vINP[i].Type=="PVS") {
     temp=P.Vlnum Mul(i,"KWH-DISP-M",P.Vnum(i,"MULTIPLIER"));
     mKWH DISP+=P.Vnum(i,"KWH-DISP")*P.Vnum(i,"MULTIPLIER");
     for (int j=0;j<12;j++) mKWH_DISP_M[j]+=temp[j];
   }

 mQ_DISP=mKWH_DISP*3.6;
 for (int j=0;j<12;j++) mQ DISP M[j]=mKWH DISP M[j]*3.6;


 // Deducting the primary energy displaced due to renewables for RoI
 if((P.Vcodid(cId,"BR-TYPE")==5)   ||   (P.Vcodid(cId,"EPC-TYPE")==6))
//Republic of Ireland only
```

```
     mKWH PRIM IR-=mKWH DISP*T.Value("tPRIM IR",1,11);
 // Deducting the primary energy displaced due to renewables for
Jersey
 else if((P.Vcodid(cId,"BR-TYPE")==6)) {  // && P.Vcodid(cId,"JERSEY-
PRIM")==1)  //Jersey
        mKWH PRIM JER-=mKWH DISP*T.Value("tPRIM JER",1,11);
        mKWH JER-=mKWH DISP;  // net delivered energy
      } //
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH DISP M[i] << ", ";
  res << mKWH DISP M[11] << " }";
  P.ModKey(Id,"KWH-DISP-M",res.str());
 }
 {
  ostringstream res; res << mKWH_DISP;
  P.ModKey(Id,"KWH-DISP",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ DISP M[i] << ", ";
  res << mQ DISP M[11] << " }";
  P.ModKey(Id,"Q-DISP-M",res.str());
 }
 {
  ostringstream res; res << mQ DISP;
  P.ModKey(Id,"Q-DISP",res.str());
 }

//Total primary energy for RoI MINUS any energy displaced due to
renewables
 {
  ostringstream res; res << mKWH PRIM IR;
  P.ModKey(Id,"KWH-PRIM-IR",res.str());
 }
//Total primary energy for Jersey MINUS any energy displaced due to
renewables
 {
  ostringstream res; res << mKWH PRIM JER;
  P.ModKey(Id,"KWH-PRIM-JER",res.str());
 }
//Total delivered energy for Jersey MINUS any energy displaced due to
renewables
 {
  ostringstream res; res << mKWH JER;
  P.ModKey(Id,"KWH-JER",res.str());
 }

}


void GENERAL::fKG CO2 CHP(PROJECTDefaults& D, PROJECTTables& T, BOC
&P) {
  if(func report) cout << endl << "  " << "GENERAL::fKG CO2 CHP";
//kgCO2 produced by CHP fuel use is written to general keywords KG-
CO2-CHP-M (monthly) and KG-CO2-CHP (annual)

  deque<double> mKG_CO2_CHP_M;
  double mKG CO2 CHP=0.0;

    if (P.Vint(Id,"CHP-POS")>0) {
      mKG CO2 CHP M=P.Vlnum(P.Vint(Id,"CHP-POS"),"KG-CO2-M");
      mKG CO2 CHP=P.Vnum(P.Vint(Id,"CHP-POS"),"KG-CO2");
    }
      else {
      for (int i=0;i<12;i++)  mKG CO2 CHP M.push back(0.0);
    }
```

```
  {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
     res << mKG_CO2_CHP_M[i] << ", ";
  res << mKG CO2 CHP M[11] << " }";
  P.ModKey(Id,"KG-CO2-CHP-M",res.str());  //kg CO2
  }
  {
  ostringstream res; res << mKG CO2 CHP;
  P.ModKey(Id,"KG-CO2-CHP",res.str());    //kg CO2
  }
}

void GENERAL::fKG CO2 DISP(PROJECTDefaults& D, PROJECTTables& T, BOC
&P) {
  if(func_report) cout << endl << "  " << "GENERAL::fKG_CO2_DISP";

  double mKG CO2 DISP M[12]={ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };
  double mKG CO2 DISP=0.0;
  double mKG CO2 PV M[12]={ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };
  double mKG CO2 CHP M[12]={ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };
  double mKG CO2 WIND M[12]={ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };
  double mKG CO2 PV=0.0;
  double mKG CO2 CHP=0.0;
  double mKG CO2 WIND=0.0;
  deque<double> temp;

  for (int i=0;i<P.vINP.size();i++) {
    if (P.vINP[i].Type=="PVS") {
        temp=P.Vlnum Mul(i,"KG-CO2-DISP-M",P.Vnum(i,"MULTIPLIER"));
        for (int m=0;m<12;m++) mKG CO2 PV M[m]+=temp[m];
        mKG CO2 PV+=P.Vnum(i,"KG-CO2-DISP")*P.Vnum(i,"MULTIPLIER");
    }
    if (P.vINP[i].Type=="CCHP-GENERATOR") {
        temp=P.Vlnum Mul(i,"KG-CO2-DISP-M",1);
//P.Vnum(i,"MULTIPLIER"));
        for (int m=0;m<12;m++) mKG_CO2_CHP_M[m]+=temp[m];
        mKG CO2 CHP+=P.Vnum(i,"KG-CO2-DISP");
//*P.Vnum(i,"MULTIPLIER");
    }
    if (P.vINP[i].Type=="WIND-GENERATOR") {
        temp=P.Vlnum Mul(i,"KG-CO2-DISP-M",P.Vnum(i,"MULTIPLIER"));
        for (int m=0;m<12;m++) mKG_CO2_WIND_M[m]+=temp[m];
        mKG CO2 WIND+=P.Vnum(i,"KG-CO2-DISP")*P.Vnum(i,"MULTIPLIER");
    }
  }

//total KG-CO2-DISP:
  for (int i=0;i<12;i++) {

mKG CO2 DISP M[i]=mKG CO2 PV M[i]+mKG CO2 CHP M[i]+mKG CO2 WIND M[i];
    mKG_CO2_DISP+=mKG_CO2_DISP_M[i];
  }

{
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKG_CO2_DISP_M[i] << ", ";
  res << mKG CO2 DISP M[11] << " }";
  P.ModKey(Id,"KG-CO2-DISP-M",res.str());
 }
 {
  ostringstream res; res << mKG CO2 DISP;
  P.ModKey(Id,"KG-CO2-DISP",res.str());
```

```
 }

}


void GENERAL::fKG CO2 END USE(PROJECTDefaults& D, PROJECTTables& T,
BOC& P) {
  if(func_report) cout << endl << "  " << "GENERAL::fKG_CO2_END_USE";
//calculates project KG CO2 by end use


}

void GENERAL::fKG_CO2(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "GENERAL::fKG CO2";

//for each fuel type, annual values
 double mKG CO2 NATGAS=0.0;
 double mKG CO2 LPG=0.0;
 double mKG_CO2_BIOGAS=0.0;
 double mKG CO2 OIL=0.0;
 double mKG CO2 COAL=0.0;
 double mKG CO2 ANTHRACITE=0.0;
 double mKG CO2 SMOKELESS=0.0;
 double mKG CO2 DUELFUEL=0.0;
 double mKG CO2 BIOMASS=0.0;
 double mKG CO2 SUPELEC=0.0;
 double mKG CO2 WASTEHEAT=0.0;
 double mKG CO2 DISTRICT HEATING=0.0;

 //kg CO2 produced by CHP
double mKG CO2 DISP=P.Vnum(Id,"KG-CO2-DISP");

//total
 double mKG CO2=0.0;

 if((P.Vcodid(cId,"BR-TYPE")==5)  ||  (P.Vcodid(cId,"EPC-TYPE")==6)){
//Republic of Ireland only
     //use conversion factor from table to calc kG CO2 for each fuel
type, USE PRIMARY ENERGY IN KWH FOR CALCULATION:
     mKG CO2 NATGAS=P.Vnum(Id,"KWH-NATGAS")*T.Value("tKGCO2 IR",1,1);
     mKG_CO2_LPG=P.Vnum(Id,"KWH-LPG")*T.Value("tKGCO2_IR",1,2);
     mKG CO2 BIOGAS=P.Vnum(Id,"KWH-BIOGAS")*T.Value("tKGCO2 IR",1,3);
     mKG CO2 OIL=P.Vnum(Id,"KWH-OIL")*T.Value("tKGCO2 IR",1,4);
     mKG_CO2_COAL=P.Vnum(Id,"KWH-COAL")*T.Value("tKGCO2_IR",1,5);
     mKG CO2 ANTHRACITE=P.Vnum(Id,"KWH-
ANTHRACITE")*T.Value("tKGCO2 IR",1,6);
     mKG_CO2_SMOKELESS=P.Vnum(Id,"KWH-
SMOKELESS")*T.Value("tKGCO2 IR",1,7);
     mKG CO2 DUELFUEL=P.Vnum(Id,"KWH-
DUELFUEL")*T.Value("tKGCO2_IR",1,8);
     mKG CO2 BIOMASS=P.Vnum(Id,"KWH-
BIOMASS")*T.Value("tKGCO2 IR",1,9);
    //                                mKG_CO2_SUPELEC=P.Vnum(Id,"KWH-
SUPELEC")*T.Value("tKGCO2 IR",1,10)*(1-
T.Value("tPOWER FACTOR",1,P.Vcodid(Id,"ELEC-POWER-FACTOR")));      //
v1.0.b Addition of power factor correction
    // Applying the correction for electric power factor is now done
on ZONE calculated energy
    // rather than CO2 so would now be included in KWH-SUPELEC - AN
18/01/08
     mKG CO2 SUPELEC=P.Vnum(Id,"KWH-
SUPELEC")*T.Value("tKGCO2_IR",1,10);
     mKG CO2 WASTEHEAT=P.Vnum(Id,"KWH-
WASTEHEAT")*T.Value("tKGCO2 IR",1,12);
     mKG_CO2_DISTRICT_HEATING=P.Vnum(Id,"KWH-DISTRICT-
HEATING")*T.Value("tKGCO2 IR",1,13);
 }
 else if((P.Vcodid(cId,"BR-TYPE")==6)) { //Jersey
```

```
     //use conversion factor from table to calc kG CO2 for each fuel
type, USE PRIMARY ENERGY IN KWH FOR CALCULATION:
     mKG CO2 NATGAS=P.Vnum(Id,"KWH-
NATGAS")*T.Value("tKGCO2 JER",1,1);
     mKG_CO2_LPG=P.Vnum(Id,"KWH-LPG")*T.Value("tKGCO2_JER",1,2);
     mKG CO2 BIOGAS=P.Vnum(Id,"KWH-
BIOGAS")*T.Value("tKGCO2 JER",1,3);
     mKG_CO2_OIL=P.Vnum(Id,"KWH-OIL")*T.Value("tKGCO2_JER",1,4);
     mKG CO2 COAL=P.Vnum(Id,"KWH-COAL")*T.Value("tKGCO2 JER",1,5);
     mKG CO2 ANTHRACITE=P.Vnum(Id,"KWH-
ANTHRACITE")*T.Value("tKGCO2_JER",1,6);
     mKG CO2 SMOKELESS=P.Vnum(Id,"KWH-
SMOKELESS")*T.Value("tKGCO2 JER",1,7);
     mKG_CO2_DUELFUEL=P.Vnum(Id,"KWH-
DUELFUEL")*T.Value("tKGCO2 JER",1,8);
     mKG CO2 BIOMASS=P.Vnum(Id,"KWH-
BIOMASS")*T.Value("tKGCO2_JER",1,9);
    //                              mKG CO2 SUPELEC=P.Vnum(Id,"KWH-
SUPELEC")*T.Value("tKGCO2 JER",1,10)*(1-
T.Value("tPOWER_FACTOR",1,P.Vcodid(Id,"ELEC-POWER-FACTOR")));      //
v1.0.b Addition of power factor correction
    // Applying the correction for electric power factor is now done
on ZONE calculated energy
    // rather than CO2 so would now be included in KWH-SUPELEC - AN
18/01/08
     mKG CO2 SUPELEC=P.Vnum(Id,"KWH-
SUPELEC")*T.Value("tKGCO2 JER",1,10);
     mKG CO2 WASTEHEAT=P.Vnum(Id,"KWH-
WASTEHEAT")*T.Value("tKGCO2 JER",1,12);
     mKG CO2 DISTRICT HEATING=P.Vnum(Id,"KWH-DISTRICT-
HEATING")*T.Value("tKGCO2 JER",1,13);
 }
 else { // all UK
     //use conversion factor from table to calc kG CO2 for each fuel
type, USE PRIMARY ENERGY IN KWH FOR CALCULATION:
     mKG CO2 NATGAS=P.Vnum(Id,"KWH-NATGAS")*T.Value("tKGCO2",1,1);
     mKG CO2 LPG=P.Vnum(Id,"KWH-LPG")*T.Value("tKGCO2",1,2);
     mKG CO2 BIOGAS=P.Vnum(Id,"KWH-BIOGAS")*T.Value("tKGCO2",1,3);
     mKG_CO2_OIL=P.Vnum(Id,"KWH-OIL")*T.Value("tKGCO2",1,4);
     mKG CO2 COAL=P.Vnum(Id,"KWH-COAL")*T.Value("tKGCO2",1,5);
     mKG CO2 ANTHRACITE=P.Vnum(Id,"KWH-
ANTHRACITE")*T.Value("tKGCO2",1,6);
     mKG CO2 SMOKELESS=P.Vnum(Id,"KWH-
SMOKELESS")*T.Value("tKGCO2",1,7);
     mKG_CO2_DUELFUEL=P.Vnum(Id,"KWH-
DUELFUEL")*T.Value("tKGCO2",1,8);
     mKG CO2 BIOMASS=P.Vnum(Id,"KWH-BIOMASS")*T.Value("tKGCO2",1,9);
    //                              mKG_CO2_SUPELEC=P.Vnum(Id,"KWH-
SUPELEC")*T.Value("tKGCO2",1,10)*(1-
T.Value("tPOWER FACTOR",1,P.Vcodid(Id,"ELEC-POWER-FACTOR")));      //
v1.0.b Addition of power factor correction
    // Applying the correction for electric power factor is now done
on ZONE calculated energy
    // rather than CO2 so would now be included in KWH-SUPELEC - AN
18/01/08
     mKG CO2 SUPELEC=P.Vnum(Id,"KWH-SUPELEC")*T.Value("tKGCO2",1,10);
     mKG_CO2_WASTEHEAT=P.Vnum(Id,"KWH-
WASTEHEAT")*T.Value("tKGCO2",1,12);
     mKG CO2 DISTRICT HEATING=P.Vnum(Id,"KWH-DISTRICT-
HEATING")*T.Value("tKGCO2",1,13);
 }


mKG CO2=mKG CO2 NATGAS+mKG CO2 LPG+mKG CO2 BIOGAS+mKG CO2 OIL+mKG CO2
 COAL+mKG CO2 ANTHRACITE+mKG CO2 SMOKELESS+mKG CO2 DUELFUEL+mKG CO2 B
IOMASS+mKG_CO2_SUPELEC+mKG_CO2_WASTEHEAT+mKG_CO2_DISTRICT_HEATING-
mKG CO2 DISP;

 {
```

```
  ostringstream res; res << mKG_CO2_NATGAS;
  P.ModKey(Id,"KG-CO2-NATGAS",res.str());
 }
 {
  ostringstream res; res << mKG_CO2_LPG;
  P.ModKey(Id,"KG-CO2-LPG",res.str());
 }
 {
  ostringstream res; res << mKG_CO2_BIOGAS;
  P.ModKey(Id,"KG-CO2-BIOGAS",res.str());
 }
 {
  ostringstream res; res << mKG_CO2_OIL;
  P.ModKey(Id,"KG-CO2-OIL",res.str());
 }
 {
  ostringstream res; res << mKG_CO2_COAL;
  P.ModKey(Id,"KG-CO2-COAL",res.str());
 }
 {
  ostringstream res; res << mKG_CO2_ANTHRACITE;
  P.ModKey(Id,"KG-CO2-ANTHRACITE",res.str());
 }
 {
  ostringstream res; res << mKG_CO2_SMOKELESS;
  P.ModKey(Id,"KG-CO2-SMOKELESS",res.str());
 }
 {
  ostringstream res; res << mKG_CO2_DUELFUEL;
  P.ModKey(Id,"KG-CO2-DUELFUEL",res.str());
 }
 {
  ostringstream res; res << mKG_CO2_BIOMASS;
  P.ModKey(Id,"KG-CO2-BIOMASS",res.str());
 }
 {
  ostringstream res; res << mKG_CO2_SUPELEC;
  P.ModKey(Id,"KG-CO2-SUPELEC",res.str());
 }
 {
  ostringstream res; res << mKG_CO2_WASTEHEAT;
  P.ModKey(Id,"KG-CO2-WASTEHEAT",res.str());
 }
 {
  ostringstream res; res << mKG_CO2_DISTRICT_HEATING;
  P.ModKey(Id,"KG-CO2-DISTRICT-HEATING",res.str());
 }
 {
  ostringstream res; res << mKG_CO2;
  P.ModKey(Id,"KG-CO2",res.str());
 }
//write CO2 EXCLUDING displaced CO2:
 {
  ostringstream res; res << (mKG_CO2+mKG_CO2_DISP);
  P.ModKey(Id,"KG-CO2-EXCL-RENEWABLES",res.str());
 }
}


void GENERAL::fMAIN_FUEL_TYPE(PROJECTDefaults& D, PROJECTTables& T,
BOC& P) {
  if(func_report) cout << endl << "  " << "GENERAL::fMAIN_FUEL_TYPE";

  deque<double> mFUEL;
  mFUEL.push_back(P.Vnum(Id,"Q-NATGAS-HEAT"));
  mFUEL.push_back(P.Vnum(Id,"Q-LPG-HEAT"));
  mFUEL.push_back(P.Vnum(Id,"Q-BIOGAS-HEAT"));
  mFUEL.push_back(P.Vnum(Id,"Q-OIL-HEAT"));
  mFUEL.push_back(P.Vnum(Id,"Q-COAL-HEAT"));
```

```
  mFUEL.push back(P.Vnum(Id,"Q-ANTHRACITE-HEAT"));
  mFUEL.push back(P.Vnum(Id,"Q-SMOKELESS-HEAT"));
  mFUEL.push back(P.Vnum(Id,"Q-DUELFUEL-HEAT"));
  mFUEL.push back(P.Vnum(Id,"Q-BIOMASS-HEAT"));
  mFUEL.push_back(P.Vnum(Id,"Q-SUPELEC-HEAT"));
  mFUEL.push back(0); //for grid displaced elec
  mFUEL.push back(P.Vnum(Id,"Q-WASTEHEAT-HEAT"));
  mFUEL.push_back(P.Vnum(Id,"Q-DISTRICT-HEATING-HEAT"));

  int mMain=14;   // AN (28/01/09): codid 14 for keyword MAIN-FUEL-
TYPE corresponds to the option "Undefined" (will be the option if
there is no heating or hot water in the building)
                  // JO (14/02/09): From v3.3.c it will be "Other"
instead of "Undefined"
  double mMax=0.0;
  for(int i=0;i<mFUEL.size();i++)
    if(mFUEL[i]>mMax) { mMax=mFUEL[i]; mMain=i+1; }

  if(P.Vcodid(Id,"BUILDING-ENVIRONMENT")==6) mMain=14;

 {
  ostringstream res; res << P.Vcod(Id,"MAIN-FUEL-TYPE",mMain);    //
Main heating fuel ONLY (space heating and DHW only)
  P.ModKey(Id,"MAIN-FUEL-TYPE",res.str());
 }


 // !!! Calculate the total thermal output due to heat pumps in the
building (in HVACs and HWSs)
 // Annual heating (space heating and DHW only) attributed to heat
pumps.
 double mKWH HEATPUMP heat=0.0;
 double iMultiplier;

 for(int i=0;i<P.vINP.size();i++) {
   switch(P.vINP[i].Ntype) {
     case 1: { //HVAC

                iMultiplier=P.Vnum(i,"MULTIPLIER");
                deque<int>    mDHW SERVED REF=P.Vlrefid(i,"DHW-SERVED-
REF");

                // if the heat source of the HVAC is a heat pump
                if      (      (P.Vcodid(i,"HEAT-SOURCE")==12)      ||
(P.Vcodid(i,"HEAT-SOURCE")==13) ||
                    (P.Vcodid(i,"HEAT-SOURCE")==18)                ||
(P.Vcodid(i,"HEAT-SOURCE")==19) ) {

                  // Check if the fuel used by the heat pump is the
same as the calculated main heating fuel
                  if (P.Vcodid(i,"FUEL-TYPE")==mMain) {
                    mKWH HEATPUMP heat+=2*P.Vnum(i,"KWH-HEATPUMP-
HEAT")*iMultiplier;
                    // Include the energy used by any DHW generators
that are served by the HVAC
                    for (int j=0;j< mDHW SERVED REF.size();j++)

mKWH HEATPUMP heat+=2*P.Vnum(mDHW SERVED REF[j],"KWH-DHW");
                  } // end if

                  else { // i.e., the fuel used by the heat pump is
NOT the same as the calculated main heating fuel
                    mKWH_HEATPUMP_heat+=P.Vnum(i,"KWH-HEATPUMP-
HEAT")*iMultiplier;
                    // Include the energy used by any DHW generators
that are served by the HVAC
                    for (int j=0;j< mDHW SERVED REF.size();j++)

mKWH_HEATPUMP_heat+=P.Vnum(mDHW_SERVED_REF[j],"KWH-DHW");
```

```
                  } // end else

                } // end if
     }
     break;
     case 12: { //DHW GENERATOR

                deque<double> mKWH_DHW_FUEL=P.Vlnum(i,"KWH-DHW-M");
                for (int m=0;m<12;m++) {
                  if (P.Vcodid(i,"HEAT-GEN-TYPE")==7)  // 7=Heat pump

                    // Check if the fuel used by the heat pump is the
same as the calculated main heating fuel
                    if (P.Vcodid(i,"FUEL-TYPE")==mMain)
                      mKWH HEATPUMP heat+=2*mKWH DHW FUEL[m];
                    else // i.e., the fuel used by the heat pump is
NOT the same as the calculated main heating fuel
                      mKWH HEATPUMP heat+=mKWH DHW FUEL[m];
                } // end for
     }
     break;
   } // end switch

 } // end for

 {
  ostringstream res; res << mKWH HEATPUMP heat*3.6;
  P.ModKey(Id,"Q-HEATPUMP-HEAT",res.str());
 }

 {
  ostringstream res; res << mKWH HEATPUMP heat/2.0;
  P.ModKey(Id,"KWH-HEAT-PUMP",res.str());
 }



  //!! Check for availability of gas on site, even if not used in the
actual building
  if( (P.Vcodid(cId,"BR-TYPE")==3) || (P.Vcodid(cId,"EPC-TYPE")==4) )
{
    // 1=OIL , 1=YES
    if    (    (P.Vcodid(Id,"NOT-FUEL")==1)    &&    (P.Vcodid(Id,"GAS-
AVAILABILITY")==1) )
      {
       ostringstream res; res << "GAS";     // In which case, set
notional fuel to gas
       P.ModKey(Id,"NOT-FUEL",res.str());
      }
  } // end if
  //Set notional fuel type - OIL (if there is no gas in the actual
building), or GAS otherwise (OIL is default in matrix)
  //unless the main heating fuel in the actual building is a low or
zero carbon heating technology (Biogas, Biomass, Waste Heat, or Heat
pump)
  //when GAS must be used in the notional building
  else {
    if   (   (P.Vcod(Id,"MAIN-FUEL-TYPE")=="Biogas")||(P.Vcod(Id,"MAIN-
FUEL-TYPE")=="Biomass")||
       (P.Vcod(Id,"MAIN-FUEL-TYPE")=="Waste    Heat")||((P.Vnum(Id,"Q-
HEATPUMP-HEAT"))>mMax) )
        // if the main heating fuel is biogas, biomass, or waste
heat, OR if the main heating source is a heat pump
      {
       ostringstream res; res << "GAS";
       P.ModKey(Id,"NOT-FUEL",res.str());
      }
  }
```

```
}


void  PROJECT::fMAIN FUEL TYPE(PROJECTDefaults&  D,  PROJECTTables&  T,
BOC& P) {
  if(func report) cout << endl << "  " << "PROJECT::fMAIN FUEL TYPE";
   dGEN[0].fMAIN FUEL TYPE(D,T,P);
}

void        PROJECT::fBUILDING ENVIRONMENT(PROJECTDefaults&        D,
PROJECTTables& T, BOC& P) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"PROJECT::fBUILDING ENVIRONMENT";

 double mENV[5]={ 0.0, 0.0, 0.0, 0.0, 0.0};

// ZONE SERVICE-STRATEGY categories are:
          // 1)Heated and Naturally Ventilated
          // 2)Heated and Mechanically Ventilated
          // 3)Air Conditioned
          // 4)No heating or cooling
          // From 01 Jan 08
          // 5)Mixed-mode and Naturally Ventilated
          // 6)Mixed-mode and Mechanically Ventilated

 for (int i=0;i<vHVAC.size();i++)
   for(int j=0;j<vHVAC[i].vZ.size();j++) {
     switch(P.Vint(vHVAC[i].vZ[j].Id,"SERVICE-STRATEGY")) {
       case   1:   {   mENV[0]+=P.Vnum(vHVAC[i].vZ[j].Id,"KG-CO2");   }
break;
       case   2:   {   mENV[1]+=P.Vnum(vHVAC[i].vZ[j].Id,"KG-CO2");   }
break;
       case   3:   {   mENV[2]+=P.Vnum(vHVAC[i].vZ[j].Id,"KG-CO2");   }
break;
       case   5:   {   mENV[3]+=P.Vnum(vHVAC[i].vZ[j].Id,"KG-CO2");   }
break;
       case   6:   {   mENV[4]+=P.Vnum(vHVAC[i].vZ[j].Id,"KG-CO2");   }
break;
     }
   }

  int mMain=6;  // JO (14/02/09): 6 means "Unconditioned". It used to
be "Undefined"
  double mMax=0;
  for(int i=0;i<5;i++)
    if(mENV[i]>mMax) { mMax=mENV[i]; mMain=i+1; }

 {
  ostringstream res; res << P.Vcod(gId,"BUILDING-ENVIRONMENT",mMain);
  P.ModKey(gId,"BUILDING-ENVIRONMENT",res.str());
 }

}


// End Object: general
```

GLASS

Specifies the type of glass in a window or rooflight. U-name must be specified so that the GLASS may be referenced in a subsequent WINDOW instruction.

Example input:

```
"Double" = GLASS
```

```
U-VALUE              = 2.1
TOT-SOL-TRANS        = 0.821
LIG-SOL-TRANS        = 0.898
..
```

U-VALUE
Specifies the overall conductance of the surface including the inside and outside air films, in W/m^2K. For the GLASS object, the U-value should include the glazing, in a vertical inclination, the frame, and any bars that form part of the glazing system. This keyword is required.

TOT-SOL-TRANS
This is the total solar energy transmittance of the glazing, defined as the time-averaged ratio of energy passing through the un-shaded element to that incident upon it (In NEN 2916:1998 and is adopted from 6.9.3 of NEN 5128.). The default value is 1.

LIG-SOL-TRANS
This is the visible solar energy transmittance of the glazing, expressed as a fraction of the visible solar energy incident on the window. The default value is 1.

c++ code

```
//: glass.cpp

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void GLASS::TEST() {
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test GLASS" << endl;
//  cout << "Fin de Test GLASS" << endl;
  BEEMObject::TEST();
}

void GLASS::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "GLASS::Calc";
  if(calc report) cout << endl << P.vINP[Id].Name;

}

void GLASS::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P,
BOC &N, int not case) {
  if(func report) cout << endl << "   " << "GLASS::Notional   " <<
not_case;

// Notional Glass
  {
    BObject Temp;
    Temp.Ntype=3;
    Temp.Type="GLASS";
    Temp.Name="Notional Glass";
    KeywordValue KV;
   {
    ostringstream res; res << T.Value("NotGlass L Value",1,not case);
//values have been changed in v1.2.a to use l(perp), we multiply by
0.9 in window to obtain figure averaged over all angles
    KV.Reset(); KV.Init(1,res.str());
    Temp.Prop.insert(make pair("LIG-SOL-TRANS",KV));
   }
```

```
    {
     ostringstream res; res << T.Value("NotGlass G Value",1,not case);
//values have been changed in v1.2.a to use g(perp), we multiply by
0.9 in window to obtain figure averaged over all angles
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make pair("TOT-SOL-TRANS",KV));
    }
    {
     ostringstream              res;              res              <<
T.Value("NotGlass U Value",1,not case)*T.Value("Not U Value Factor",1
,not_case);   // accounting for alpha value
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make pair("U-VALUE",KV));
    }
     N.vINP.push back(Temp);
   }

  if (not case>=9) { // if this notional building is for E, W, or NI
   // Notional Rooflight Glass
   {
     BObject Temp;
     Temp.Ntype=3;
     Temp.Type="GLASS";
     Temp.Name="Notional Rooflight Glass";
     KeywordValue KV;
    {
     ostringstream res; res << T.Value("NotGlass L Value",1,11);
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make pair("LIG-SOL-TRANS",KV));
    }
    {
     ostringstream res; res << T.Value("NotGlass G Value",1,11);
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make pair("TOT-SOL-TRANS",KV));
    }
    {
     ostringstream              res;              res              <<
T.Value("NotGlass U Value",1,11)*T.Value("Not U Value Factor",1,11);
// accounting for alpha value
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make pair("U-VALUE",KV));
    }
     N.vINP.push back(Temp);
   }
  } // end if

// Notional Display Glass
   {
     BObject Temp;
     Temp.Ntype=3;
     Temp.Type="GLASS";
     Temp.Name="Notional Display Glass";
     KeywordValue KV;
    {
     ostringstream res; res << T.Value("NotGlass L Value",2,not case);
//display glazing does not contribute to daylight in 2010 E&W. This
is redundant with WINDOW::Init()
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make pair("LIG-SOL-TRANS",KV));
    }
    {
     ostringstream res; res << T.Value("NotGlass G Value",2,not case);
//values have been changed in v1.2.a to use g(perp), we multiply by
0.9 in window to obtain figure averaged over all angles:
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make_pair("TOT-SOL-TRANS",KV));
    }
    {
     ostringstream res; res << T.Value("NotGlass U Value",2,not case);
```

```
// AN (25/11/08): For England, Wales, and NI, the U-value of notional
display window was modified to 5.7 W/m2K according to published NCM
Modelling Guide Oct08
    KV.Reset(); KV.Init(1,res.str());
    Temp.Prop.insert(make_pair("U-VALUE",KV));
   }
    N.vINP.push back(Temp);
  }
// */two notional glass types added CD
}

void GLASS::Recommendation(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func_report) cout << endl << "  " << "GLASS::Recommendation";

}


// End Object: GLASS
```

HVAC-SYSTEM

Specifies information associated with an HVAC system. A U-name must be specified, but it is used only for output reports. It is the first object in the hierarchy of geometry definitions, followed by ZONE, WALL, (WALL-TM-BRIDGE), WINDOW, (WINDOW-TM-BRIDGE), or DOOR, (DOOR-TM-BRIDGE).

Example input:

```
"HVAC for the example building" = HVAC-SYSTEM
   ISBEM-ID            = 11
   TYPE                = Single-duct VAV
   HEAT-SOURCE         = LTHW boiler
   CHP                 = NO
   FUEL-TYPE           = Natural Gas
   HEAT-REC-SYSTEM     = Plate heat exchanger (Recuperator)
   HEAT-REC-SEFF       = 0.65
   COOL-SSEER          = 2.489
   MIXED-MODE          = YES
   HEAT-SSEFF          = 0.807
   AUX-ENERGY-CORR     = 0.551
   HEAT-GEN-SEFF       = 0.89
   COOL-GEN-SEER       = 3.5
   COOL-GEN-EER        = 3.125
   METERING            = 0.95
   SFP-CHECK           = YES
   SFP                 = 2.2
   DUCT-LEAKAGE-PC     = 21
   CHILLER-TYPE        = Air cooled chiller
   ..
```

***MULTIPLIER* (NO LONGER IN USE)**

This keyword specifies how many HVAC-SYSTEMS with this description exist in the project. E.g., if the value of multiplier is 3, there are 3 HVAC-SYSTEMS each with identical keyword values in the project. This enables repeating HVAC-SYSTEMS to be entered quickly and easily. The default value is 1.

ISBEM-ID
This keyword specifies the interface's ID number for the HVAC-SYSTEM object, and it is used when calculated values relating to that object need to be reported within the interface.

TYPE
Specifies the type of HVAC system, as one of the values listed below. The default is 'Single-duct VAV'.

No Heating or Cooling
Central heating using water: radiators
Central heating using water: convectors
Central heating using water: floor heating
Central heating using air distribution
Other local room heater - fanned
Unflued radiant heater
Flued radiant heater
Multiburner radiant heaters
Flued forced-convection air heaters
Unflued forced-convection air heaters
Single-duct VAV
Dual-duct VAV
Indoor packaged cabinet (VAV)
Fan coil systems
Induction system
Constant volume system (fixed fresh air rate)
Constant volume system (variable fresh air rate)
Multizone (hot deck/cold deck)
Terminal reheat (constant volume)
Dual duct (constant volume)
Chilled ceilings or passive chilled beams and displacement ventilation
Active chilled beams
Water loop heat pump
Variable refrigerant flow (NO LONGER IN USE)
Split or multi-split system
Split or multi-split system with ventilation (NO LONGER IN USE)
Single room cooling system
Other local room heater - unfanned

***HG-TYPE* (NOT IN USE)**
This is a number representing the heating system type according to the Heating Guide produced by Faber Maunsell.

HEAT-SOURCE
Specifies the heat source for the HVAC system, as one of the values listed below. The default is 'LTHW boiler' (low temperature hot water boiler).

LTHW boiler
MTHW boiler
HTHW boiler
Air heater
Unflued gas warm air heater
Unitary radiant heater
Radiant heater
Unflued radiant heater
Direct or storage electric heater
Heat pump (gas/oil): air source
Heat pump (gas/oil): ground or water source
Room heater
District heating
Direct gas firing
Heat pump (electric): air source
Heat pump (electric): ground or water source

BH-HEAT-SOURCE
This keyword in included in the input if the HVAC system is bi-valent (or multi-valent), i.e., it is served by more than one heat generator. It specifies a list of the secondary heat sources that provide heating, to support the primary heat source already defined in the HVAC-SYSTEM object's HEAT-SOURCE keyword, in the form shown below (enclosed by braces and separated by commas). If the HVAC system is served by only one heat source, then this keyword is <u>not</u> included in the input.

{ "sec-heat-source$_1$", "sec-heat-source$_2$", .., .., .., .., "sec-heat-source$_n$" } where sec-heat-source$_i$ is one of the values listed below.

LTHW boiler
MTHW boiler
HTHW boiler
Air heater
Unflued gas warm air heater
Unitary radiant heater
Radiant heater
Unflued radiant heater
Direct or storage electric heater
Heat pump (gas/oil): air source
Heat pump (gas/oil): ground or water source
Room heater
District heating
Direct gas firing
Heat pump (electric): air source
Heat pump (electric): ground or water source

CHP
This keyword specifies whether the HVAC system uses a CCHP generator to provide a proportion of its heat load. The value can be YES or NO. The default is NO.

FUEL-TYPE
Specifies the fuel type for the HVAC system's heat source, as one of the values listed below. The default is 'Oil'.

Natural Gas
LPG
Biogas
Oil
Coal
Anthracite
Smokeless Fuel (inc Coke)
Dual Fuel Appliances (Mineral + Wood)
Biomass
Grid Supplied Electricity
Waste Heat
District Heating

BH-FUEL-TYPE
This keyword in included in the input if the HVAC system is bi-valent (or multi-valent), i.e., it is served by more than one heat generator. It specifies a list of the fuel types supplying the secondary heat sources that provide heating (to support the primary heat source already defined in the HVAC-SYSTEM object's HEAT-SOURCE keyword and which is powered by the fuel defined in the HVAC-SYSTEM object's FUEL-TYPE keyword), in the form shown below (enclosed by braces and separated by commas). The list of fuel types in this keyword must correspond, in number and in order, to the heat sources listed in the HVAC-SYSTEM object's BH-HEAT-SOURCE keyword. If the HVAC system is served by only one heat source (and one fuel type), then this keyword is not included in the input.

{ "sec-fuel-type$_1$", "sec-fuel-type$_2$", .., .., .., .., "sec-fuel-type$_n$" } where sec-fuel-type$_i$ is one of the values listed below.

Natural Gas
LPG
Biogas
Oil
Coal
Anthracite
Smokeless Fuel (inc Coke)
Dual Fuel Appliances (Mineral + Wood)
Biomass
Grid Supplied Electricity
Waste Heat
District Heating

FUEL-TYPE-COOL
Specifies the fuel type for the cooling system, as one of the values listed below. The default is 'Grid Supplied Electricity'.

Natural Gas
LPG
Biogas
Oil
Grid Supplied Electricity

HEAT-REC-SYSTEM
Specifies the type of heat recovery system for the HVAC system, as one of the values listed below. The default is 'No heat recovery'.

No heat recovery
Plate heat exchanger (Recuperator)
Heat pipes
Thermal wheel
Run around coil

HEAT-REC-SEFF
Specifies the seasonal efficiency of the heat recovery system. The default values for the different types of heat recovery system options in the HVAC-SYSTEM object's HEAT-REC-SYSTEM keyword are listed below.

Heat Recovery System	Default Efficiency (based on CIBSE suggestions)
No heat recovery	0
Plate heat exchanger (Recuperator)	0.65
Heat pipes	0.6
Thermal wheel	0.65
Run around coil	0.5

HEAT-REC-VAR-EFF
This keyword is included in the input if the HVAC-SYSTEM object's HEAT-REC-SYSTEM keyword has a value other than 'No heat recovery'. It indicates whether the seasonal efficiency of the heat recovery system is variable. The value can be YES or NO. The default is NO.

F-CTRL-VENT-TYPE **(NO LONGER IN USE)**
Specifies whether the HVAC system has flow reduction or recirculation, as one of the values listed below. The default is 'No recirculation'.

no recirculation
less than 10%
10% to 20%
20% to 30%
30% to 40%
40% to 50%
50% to 60%
60% to 70%
70% to 80%
80% to 90%

more than 90%

COOL-SSEER
Seasonal system energy efficiency ratio for cooling of the HVAC system. Defined as cooling demand / cooling energy.

HEAT-SSEFF
Seasonal system efficiency for heating of the HVAC system. Defined as heating demand / heating energy.

BH-HEAT-SSEFF
This keyword in included in the input if the HVAC system is bi-valent (or multi-valent), i.e., it is served by more than one heat generator. It specifies a list of the seasonal system efficiencies for heating of the secondary heat sources that provide heating (to support the primary heat source already defined in the HVAC-SYSTEM object's HEAT-SOURCE keyword and whose seasonal heating system efficiency is defined in the HVAC-SYSTEM object's HEAT-SSEFF keyword), in the form shown below (enclosed by braces and separated by commas). The list of seasonal heating system efficiencies in this keyword must correspond, in number and in order, to the heat sources listed in the HVAC-SYSTEM object's BH-HEAT-SOURCE keyword. If the HVAC system is served by only one heat source (and therefore has one seasonal heating system efficiency), then this keyword is not included in the input.

{ sec-heat-sseff$_1$, sec-heat-sseff$_2$, .., .., .., .., sec-heat-sseff$_n$ } where sec-heat-sseff$_i$ is a value for the seasonal system efficiency for heating.

AUX-ENERGY **(NO LONGER IN USE)**
Auxiliary energy consumption required by the HVAC system, in kWh/m^2.

AUX-ENERGY-CORR
Auxiliary energy adjustment factor for the HVAC system due to duct leakage, AHU leakage, and controls provisions.

HEAT-GEN-SEFF ***(RENAMED FROM SYS-SEFF)***
This is the effective seasonal heat generating efficiency of the heat generator of the HVAC system. The default value is 0.4.

BH-HEAT-GEN-SEFF
This keyword in included in the input if the HVAC system is bi-valent (or multi-valent), i.e., it is served by more than one heat generator. It specifies a list of the effective seasonal heat generating efficiencies of the secondary heat sources that provide heating (to support the primary heat source already defined in the HVAC-SYSTEM object's HEAT-SOURCE keyword and whose effective seasonal heat generating efficiency is defined in the HVAC-SYSTEM object's HEAT-GEN-SEFF keyword), in the form shown below (enclosed by braces and separated by commas). The list of effective seasonal heat generator efficiencies in this keyword must correspond, in number and in order, to the heat sources listed in the HVAC-SYSTEM object's BH-HEAT-SOURCE keyword. If the HVAC system is served by only one heat source (and therefore has one effective seasonal heat generator efficiency), then this keyword is not included in the input.

{ sec-heat-gen-seff$_1$, sec-heat-gen-seff$_2$, .., .., .., .., sec-heat-gen-seff$_n$ } where sec-heat-gen-seff$_i$ is a value for the effective seasonal heat generating efficiency for the heat generator.

BH-LOAD
This keyword in included in the input if the HVAC system is bi-valent (or multi-valent), i.e., it is served by more than one heat generator. It specifies a list of the partial heating loads of the secondary heat sources that provide heating (to support the primary heat source already defined in the HVAC-SYSTEM object's HEAT-SOURCE keyword), in the form shown below (enclosed by braces and separated by commas). The list of partial loads in this keyword must correspond, in number and in order, to the heat sources listed in the HVAC-SYSTEM object's BH-HEAT-SOURCE keyword. If the HVAC system is served by only one heat source (and therefore providing the whole load), then this keyword is not included in the input.

{ sec-heat-load$_1$, sec-heat-load$_2$, .., .., .., .., sec-heat-load$_n$ } where sec-heat-load$_i$ is a value for the partial load in decimal form.

EFF-DEFAULT-HEAT
This keyword indicates whether the user has used the default effective seasonal heat generating efficiency for the heat generator of the HVAC system (the value of this keyword is YES) or manually entered a value for the efficiency (the value of this keyword is NO). The default is YES.

RADIANT-EFF
This is the ratio of the radiant heat output to energy input (for radiant HVAC systems from the options in the HVAC-SYSTEM object's TYPE keyword). The default value is 0.

COOL-GEN-SEER *(RENAMED FROM COOL-GEN-SEFF)*
This is the seasonal energy efficiency ratio for the cooling generator of the HVAC system. The default value is 0.4.

COOL-GEN-EER
This is the nominal energy efficiency ratio for the cooling generator of the HVAC system. The default value is 0.4.

EFF-DEFAULT-COOL
This keyword indicates whether the user has used the default cooling seasonal energy efficiency ratio for the cooling generator of the HVAC system (the value of this keyword is YES) or manually entered a value for the efficiency (the value of this keyword is NO). The default is YES.

CHILLER-TYPE
Specifies the type of the chiller of the HVAC system, as one of the values listed below. The default is 'Not applicable'.

Air cooled chiller
Water cooled chiller
Remote-condenser chiller

Not applicable
Heat pump (gas/oil)
Heat pump (electric)

METERING
This is a correction factor due to metering during the operation of the HVAC system. If the HVAC system has provision for metering, and the metering warns about out-of-range values, this keyword is equal to 0.95. If the HVAC system has no provision for metering, or the metering does not warn about out-of-range values, this keyword is equal to 1, i.e., no correction. The default value is 1.

***SYS-CTRL-MIN-LIST* (NO LONGER IN USE)**
This is a list of minimum system requirements for the building HVAC system type.

***SYS-CTRL-MIN* (NO LONGER IN USE)**
This keyword states whether the system complies with the minimum system control requirements. The value can be YES or NO. The default is NO.

***SYS-CTRL-ADD-LIST* (NO LONGER IN USE)**
This keyword contains a list of any additional system control elements which are included in the building HVAC system.

***SYS-CTRL-ADD-CRED* (NO LONGER IN USE)**
This keyword contains any additional system credits gained for additional system control elements listed above.

SFP-CHECK
This keyword specifies whether the HVAC system allows the user to enter a value for the specific fan power of the air distribution system (the HVAC system provides mechanical ventilation), i.e., a compliance check is required. The value can be YES or NO. The default is NO.

SFP
This keyword in included in the input if the HVAC-SYSTEM object's SFP-CHECK keyword has the value of YES. It specifies the specific fan power of the air distribution system, in W/(l/s).

DUCT-LEAKAGE-PC
This keyword specifies the percentage of air leakage through the ductwork and air handling unit if mechanical ventilation is provided by the HVAC system. The default value is 10.

***AUX-ENERGY/KWH* (NO LONGER IN USE)**
This keyword in included in the input file if the HVAC system TYPE keyword is set to either 'Flued forced-convection air heaters' or 'Unflued forced-convection air heaters'. It specifies the auxiliary energy of fanned warm air heaters, in kWh, per kWh of heating. The default value is 0.02.

DHW-SERVED-REF
This keyword in included in the input if the HVAC system serves one or more hot water systems. It specifies a list of U-name(s) previously defined in a DHW-GENERATOR

instruction(s) to define the hot water generator(s) served by the HVAC system, in the form shown below (enclosed by braces and separated by commas). The keyword is used in the generation of the recommendations report for the EPC. If the HVAC system does not serve any hot water generators (none of the DHW-GENERATOR objects in the input file have their keyword HEAT-GEN-TYPE set to 'Using Central Heating boiler' in reference to this HVAC system), then the keyword is not included in the input.

{ "dhw-gen$_1$", "dhw-gen$_2$", .., .., .., .., "dhw-gen$_n$" } where dhw-gen$_i$ is the U-name of a DHW-GENERATOR object previously defined in the input file.

CEN-TIME-CTRL
Specifies whether the heating system has centralised time control. The value can be YES or NO. The default is NO.

ROOM-TIME-CTRL
Specifies whether the heating system has local (i.e., room by room) time control. The value can be YES or NO. The default is NO.

ROOM-TEMP-CTRL
Specifies whether the heating system has local (i.e., room by room) temperature control. The value can be YES or NO. The default is NO.

OPTIMUM-START
Specifies whether the heating system has optimum start/stop control. The value can be YES or NO. The default is NO.

WEATHER-COMP
Specifies whether the heating system has weather compensation control. The value can be YES or NO. The default is NO.

MIXED-MODE
Specifies whether the HVAC system has a mixed mode operation strategy for cooling. The value can be YES or NO. The default is NO.

Modified

VARIABLE-SPEED-PUMP
This keyword specifies whether the HVAC system uses variable speed pumping, as one of the values listed below. The default is 'NO'.

Value	Description
NO	No variable speed pumping
LTHW	Variable speed pumping applies to LTHW (low temperature hot water) only
BOTH	Variable speed pumping applies to both LTHW and CHW (chilled water)

VARIABLE-SPEED-PUMP-TYPE
This keyword specifies the type of variable speed pumping, if applicable (i.e., the HVAC-SYSTEM object's VARIABLE-SPEED-PUMP keyword has a value other than NO), as one of the values listed below. The default is 'With differential sensor across pump'.

With differential sensor across pump
With differential sensor in the system
With multiple pressure sensors in the system

c++ code

```
//: hvac.cpp

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;


void HVAC::TEST() {
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test HVAC" << endl;
//  cout << "Fin de Test HVAC" << endl;
  BEEMObject::TEST();
  for(int i=0;i<vZ.size();i++)
    vZ[i].TEST();
}


void HVAC::Reset() {
  if(func report) cout << endl << "  " << "HVAC::Reset";
  dZ.clear();
//  vZ.clear();
}


void HVAC::DtoV() {
  if(func report) cout << endl << "  " << "HVAC::DtoV";
//  for(int i=0;i<dZ.size();i++)  dZ[i].DtoV();
//  vZ.reserve(dZ.size());
//  vZ.assign(dZ.begin(),dZ.end());
//  dZ.clear();
}


void HVAC::Init fuel(PROJECTDefaults& D, PROJECTTables& T, BOC& P,
PROJECTDefaults& HV, PROJECTTables& HVG) {
  if(func report) cout << endl << "  " << "HVAC::Init fuel";
//This function determines whether the HVAC-System or the associated
CHP system uses gas as the fuel type.
//This is so that SBEM can use oil as the notional fuel if there is
NO gas anywhere in the project.
//if HVAC or CHP fuel is gas, set general keyword "NOT-FUEL" to "GAS"
(ID=2)
//Note:  There  is  a  similar  function  in  'dhw generator':
DHW_GENERATOR::Init_fuel()

  //for HVAC SYSTEM:
  if (P.Vcod(Id,"FUEL-TYPE")=="Natural Gas")
    //if HVAC fuel is gas, set general keyword "NOT-FUEL" to "GAS"
(ID=2):
    {
     ostringstream res; res << "GAS";
     P.ModKey(gId,"NOT-FUEL",res.str());
    }
  else
    //if HVAC system has CHP associated with it:
    if (P.Vcod(Id,"CHP")=="YES")
      //for CHP SYSTEM:
      if(P.Vcod(P.Vint(gId,"CHP-POS"),"FUEL-TYPE")=="Natural Gas")
        //if HVAC fuel is gas, set general keyword "NOT-FUEL" to
"GAS" (ID=2):
```

```
        {
         ostringstream res; res << "GAS";
         P.ModKey(gId,"NOT-FUEL",res.str());
        }

}

void HVAC::Init(PROJECTDefaults& D, PROJECTTables& T, BOC& P,
PROJECTDefaults& HV, PROJECTTables& HVG) {
  if(func report) cout << endl << "  " << "HVAC::Init";
// Code below get the right Heating guide ID depending on Heat Source
and Fuel Type in separate function HVAC::fHVAC GUIDE NO...


//v2.0.a: HVAC systems "Variable refrigerant flow" (COD25) or "Split
or multi-split system with ventilation" (COD27)
//are no longer to be used in SBEM.
//If these systems are used, change to Split or multi-split system
(COD26)
//and add a warning in the SBEM log file
  if ((P.Vcodid(Id,"TYPE")==25) || (P.Vcodid(Id,"TYPE")==27)) {
  //add warning to log:
    {ostringstream os; os << endl << endl << "WARNING: HVAC SYSTEM
type [" << P.Vcod(Id,"TYPE") << "] has been chosen for HVAC system ["
<< P.vINP[Id].Name << endl << "This system is no longer used and has
been replaced by HVAC system 'Split or multi-split system'" << endl;
       SendLog(os.str()); WriteLog();}
  //change system type:
    {
    ostringstream res; res << "Split or multi-split system";
    P.ModKey(Id,"TYPE",res.str());
    }
  }
//v2.0.a: Heat recovery systems types have been updated in matrix -
if old system types have
//been used, convert these to new types and add a warning in the SBEM
log file
  if ((P.Vcodid(Id,"HEAT-REC-SYSTEM")==3) || (P.Vcodid(Id,"HEAT-REC-
SYSTEM")==4)) {
  //add warning to log:
    {ostringstream os; os << endl << endl << "WARNING: Heat Recovery
system type [" << P.Vcod(Id,"HEAT-REC-SYSTEM") << "] has been chosen
for HVAC system [" << P.vINP[Id].Name << "]" << endl << "This type is
no longer used and has been replaced by type 'Run around coil'" <<
endl;
       SendLog(os.str()); WriteLog();}
  //change system type:
    {
    ostringstream res; res << "Run around coil";
    P.ModKey(Id,"HEAT-REC-SYSTEM",res.str());
    }
  }

//check if you have COOL-SSEER, HEAT-SSEFF and AUX-ENERGY values if
system has cooling, heating, aux:

 if        ((T.Value("tSYS COOL",1,P.Vcodid(Id,"TYPE"))==1)        &&
(P.Vnum(Id,"COOL-SSEER")==-6666)) {
      {ostringstream os; os << endl << endl << "ERROR: Keyword COOL-
SSEER has not been defined in HVAC [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
 if        ((T.Value("tSYS HEAT",1,P.Vcodid(Id,"TYPE"))==1)        &&
(P.Vnum(Id,"HEAT-SSEFF")==-6666)) {
      {ostringstream os; os << endl << endl << "ERROR: Keyword HEAT-
SSEFF has not been defined in HVAC [" << P.vINP[Id].Name << "]" <<
endl;
```

```
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
 if (P.Vcod(gId,"ACT-NOT")=="ACT") {
    if       ((T.Value("tSYS AUX",1,P.Vcodid(Id,"TYPE"))==1)       &&
(P.Vnum(Id,"AUX-ENERGY-CORR")==-6666)) {
       {ostringstream os; os << endl << endl << "ERROR: Keyword AUX-
ENERGY-CORR has not been defined in HVAC [" << P.vINP[Id].Name << "]"
<< endl;
       SendLog(os.str()); WriteLog();}
       assert(1==0);
    }
 }


// v3.0.a R24 01/01/08 Mix mode check.
 if (P.Vcod(Id,"MIXED-MODE")=="YES")
   if (T.Value("tMIXED MODE",1,P.Vcodid(Id,"TYPE"))==0) {
      {ostringstream os; os << endl << endl << "ERROR: HVAC [" <<
P.vINP[Id].Name << "] cannot have MIXED MODE operation" << endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }

// Set multiplier to 1 to prevent any tool inputing multiplers
different to 1

 {
  ostringstream res; res << 1;
  P.ModKey(Id,"MULTIPLIER",res.str());
 }


 // Check if the actual building has a bivalent heating system, and
if yes, add the "main" system parameters to
 // the deque of supplementary systems. Otherwise, the main system is
the only system.
  if (P.Vcod(gId,"ACT-NOT")=="ACT") {   // if this is the actual
building

    deque<string> BH HS=P.Vltext(Id,"BH-HEAT-SOURCE");
    deque<string> BH_FT=P.Vltext(Id,"BH-FUEL-TYPE");
    deque<double> BH SYS EFF=P.Vlnum(Id,"BH-HEAT-SSEFF");
    deque<double> BH GEN EFF=P.Vlnum(Id,"BH-HEAT-GEN-SEFF");
    deque<double> BH_LOAD=P.Vlnum(Id,"BH-LOAD");

    // JO added, 12Nov2010, Preventing District heating in Bivalent
systems
    deque<int> toerase;
    for(int i=0;i<BH HS.size();i++)
      if(BH_HS[i]=="District heating") toerase.push_front(i);
    for(int i=0;i<toerase.size();i++) {
      BH HS.erase(BH HS.begin()+toerase[i]);
      BH_FT.erase(BH_FT.begin()+toerase[i]);
      BH SYS EFF.erase(BH SYS EFF.begin()+toerase[i]);
      BH GEN EFF.erase(BH GEN EFF.begin()+toerase[i]);
      BH_LOAD.erase(BH_LOAD.begin()+toerase[i]);
    }

    if (P.Vcodid(Id,"HEAT-SOURCE")==15) {
      BH HS.clear();
      BH FT.clear();
      BH_SYS_EFF.clear();
      BH GEN EFF.clear();
      BH LOAD.clear();
      BH_HS.push_back("-5555");
      BH FT.push back("-5555");
      BH SYS EFF.push back(-5555);
      BH_GEN_EFF.push_back(-5555);
```

```
      BH LOAD.push back(-5555);
    }

    // Heat sources
    if (BH_HS[0]=="-5555")
      BH HS[0]=P.Vcod(Id,"HEAT-SOURCE");
    else
      BH_HS.push_back(P.Vcod(Id,"HEAT-SOURCE"));

    // Fuel types
    if (BH_FT[0]=="-5555")
      BH FT[0]=P.Vcod(Id,"FUEL-TYPE");
    else
      BH_FT.push_back(P.Vcod(Id,"FUEL-TYPE"));

    // System efficiency
    if (BH_SYS_EFF[0]==-5555)
      BH SYS EFF[0]=P.Vnum(Id,"HEAT-SSEFF");
    else
      BH_SYS_EFF.push_back(P.Vnum(Id,"HEAT-SSEFF"));

    // Generator efficiency
    if (BH GEN EFF[0]==-5555)
      BH GEN EFF[0]=P.Vnum(Id,"HEAT-GEN-SEFF");
    else
      BH GEN EFF.push back(P.Vnum(Id,"HEAT-GEN-SEFF"));

    // Heat load
    double main load=1.0;

    if (BH LOAD[0]==-5555)
      BH LOAD[0]=main load;         // i.e., main system is the only
system and it provides all the load
    else {
      for(int i=0;i<BH LOAD.size();i++)
        main load-=BH LOAD[i];
      if (main load>0.0)
        BH LOAD.push back(main load);
    } // end else

    { // Save the updated deque of heat sources
     ostringstream res; res << "{ ";
     for (int i=0;i<BH HS.size()-1;i++)
       res << "\"" << BH HS[i] << "\", ";
     res << "\"" << BH_HS[BH_HS.size()-1] << "\" }";
     P.ModKey(Id,"BH-HEAT-SOURCE",res.str());
    }

    { // Save the updated deque of fuel types
     ostringstream res; res << "{ ";
     for (int i=0;i<BH_FT.size()-1;i++)
       res << "\"" << BH FT[i] << "\", ";
     res << "\"" << BH FT[BH FT.size()-1] << "\" }";
     P.ModKey(Id,"BH-FUEL-TYPE",res.str());
    }

    { // Save the updated deque of system efficiencies
     ostringstream res; res << "{ ";
     for (int i=0;i<BH SYS EFF.size()-1;i++)
       res << BH_SYS_EFF[i] << ", ";
     res << BH SYS EFF[BH SYS EFF.size()-1] << " }";
     P.ModKey(Id,"BH-HEAT-SSEFF",res.str());
    }

    { // Save the updated deque of generator efficiencies
     ostringstream res; res << "{ ";
     for (int i=0;i<BH GEN EFF.size()-1;i++)
       res << BH GEN EFF[i] << ", ";
     res << BH_GEN_EFF[BH_GEN_EFF.size()-1] << " }";
```

```
     P.ModKey(Id,"BH-HEAT-GEN-SEFF",res.str());
    }

    { // Save the updated deque of generator efficiencies
     ostringstream res; res << "{ ";
     for (int i=0;i<BH LOAD.size()-1;i++)
       res << BH LOAD[i] << ", ";
     res << BH_LOAD[BH_LOAD.size()-1] << " }";
     P.ModKey(Id,"BH-LOAD",res.str());
    }

    // !!! Calculate the equivalent system HVAC efficiency in the
actual building based on the combination of bivalent
    // !!! systems
    if (BH HS.size()>1) { // If there is more than one heat source
defined (i.e., it is a bivalent system)

      double mSSEFF=0.0;   // system efficiency
      double mGEFF=0.0;    // generator efficiency
      int FT_Id;

      for(int i=0;i<BH HS.size();i++) {

        FT Id=FuelTypeId(BH FT[i]);   // get CODID of fuel type

        mSSEFF+=BH LOAD[i]*T.Value("tKGCO2",1,FT Id)/BH SYS EFF[i];
        mGEFF+=BH LOAD[i]*T.Value("tKGCO2",1,FT Id)/BH GEN EFF[i];

      } // end for

      // Save the system efficiency equivalent to the combination of
bivalent systems
      {
       ostringstream             res;             res             <<
T.Value("tKGCO2",1,P.Vcodid(Id,"FUEL-TYPE"))/mSSEFF;
       P.ModKey(Id,"HEAT-SSEFF",res.str());
      }

      // Save the generator efficiency equivalent to the combination
of bivalent systems
      {
       ostringstream             res;             res             <<
T.Value("tKGCO2",1,P.Vcodid(Id,"FUEL-TYPE"))/mGEFF;
       P.ModKey(Id,"HEAT-GEN-SEFF",res.str());
      }

    } // end if
  } // end if

} // end Init


string HVAC::FuelNot2010(int HS, int FT, string sFT) {
  string mNot_Fuel;
  // 18=Heat pump (electric): air source , 19=Heat pump (electric):
ground or water source
  // 10=Grid Supplied Electricity
  if ( ((HS==18) || (HS==19)) && (FT==10) )
    mNot Fuel=sFT;   // same as heating fuel in the actual building

  else { // i.e., not an electric heat pump
    // Check if the heating fuel in the actual building is one whose
emission factor is >= that of oil,
    // i.e., >= 0.306 kg/kWh
    //  4=Oil,  5=Coal,  6=Anthracite,  7=Smokeless  Fuel  (inc  Coke),
10=Grid Supplied Electricity
    if ( (FT==4) || (FT==5) || (FT==6) || (FT==7) || (FT==10) )
      mNot Fuel="Oil";
```

```
    else // i.e., the heating fuel in the actual building is one
whose emission factor < that of oil,
       // i.e., < 0.306 kg/kWh
      mNot Fuel=sFT;   // same as heating fuel in the actual building
  }
   return(mNot Fuel);
}

void HVAC::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P, BOC
&N, int i) {
  if(func_report) cout << endl << "  " << "HVAC::Notional  " << i;
//HVAC system for the notional building

  BObject Temp;
  Temp.Ntype=1;
  Temp.Type="HVAC-SYSTEM";
  Temp.Name=P.vINP[i].Name;
  KeywordValue KV;
 {
  ostringstream res; res << P.Vcod(i,"TYPE");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("TYPE",KV));
 }
 {
  ostringstream res; res << P.Vcod(i,"HEAT-SOURCE");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("HEAT-SOURCE",KV));
 }

  if((P.Vcodid(cId,"BR-TYPE")==5)  ||  (P.Vcodid(cId,"EPC-TYPE")==6)){
//Republic of Ireland only
       // if fuel in the actual building is a low or zero carbon
heating technology (Biogas, Biomass, Waste Heat, or Heat pump), use
natural gas instead
    if               ((P.Vcodid(i,"FUEL-TYPE")==3)||(P.Vcodid(i,"FUEL-
TYPE")==9)||(P.Vcodid(i,"FUEL-TYPE")==12)||
        (P.Vcodid(i,"HEAT-SOURCE")==12)||(P.Vcodid(i,"HEAT-
SOURCE")==13)||(P.Vcodid(i,"HEAT-SOURCE")==18)||(P.Vcodid(i,"HEAT-
SOURCE")==19)) {
       {
        ostringstream res; res << "Natural Gas";
        KV.Reset(); KV.Init(4,res.str());
        Temp.Prop.insert(make pair("FUEL-TYPE",KV));
       }
    }
    else  // use the same fuel as in the actual building
       {
        ostringstream res; res << P.Vcod(i,"FUEL-TYPE");
        KV.Reset(); KV.Init(4,res.str());
        Temp.Prop.insert(make pair("FUEL-TYPE",KV));
       }
  }
  else  if((P.Vcodid(cId,"BR-TYPE")==6)){
    //if elec is used in the actual building, set notional fuel to
elec, otherwise use LPG
    if (P.Vcodid(i,"FUEL-TYPE")==10)
       {
        ostringstream res; res << P.Vcod(i,"FUEL-TYPE");
        KV.Reset(); KV.Init(4,res.str());
        Temp.Prop.insert(make_pair("FUEL-TYPE",KV));
       }
    else
       {
        ostringstream res; res << "LPG";
        KV.Reset(); KV.Init(4,res.str());
        Temp.Prop.insert(make_pair("FUEL-TYPE",KV));
       }
  }
  else { // all UK
```

```
    // Check if this is the 2010 notional
    if (P.Vint(gId,"NOT-CASE")>=9)  { // 9=EW NOT 2010


//      string mNot Fuel;
//         // 18=Heat pump (electric): air source , 19=Heat pump
(electric): ground or water source
//      // 10=Grid Supplied Electricity
//        if ( ((P.Vcodid(i,"HEAT-SOURCE")==18) || (P.Vcodid(i,"HEAT-
SOURCE")==19)) && (P.Vcodid(i,"FUEL-TYPE")==10) )
//         mNot Fuel=P.Vcod(i,"FUEL-TYPE");   // same as heating fuel
in the actual building
//
//      else { // i.e., not an electric heat pump
//         // Check if the heating fuel in the actual building is one
whose emission factor is >= that of oil,
//        // i.e., >= 0.306 kg/kWh
//           // 4=Oil, 5=Coal, 6=Anthracite, 7=Smokeless Fuel (inc
Coke), 10=Grid Supplied Electricity, 14=Other
//           if ( (P.Vcodid(i,"FUEL-TYPE")==4) || (P.Vcodid(i,"FUEL-
TYPE")==5) || (P.Vcodid(i,"FUEL-TYPE")==6)
//              || (P.Vcodid(i,"FUEL-TYPE")==7) || (P.Vcodid(i,"FUEL-
TYPE")==10) || (P.Vcodid(i,"FUEL-TYPE")==14) )
//          mNot Fuel="Oil";
//
//         else // i.e., the heating fuel in the actual building is
one whose emission factor < that of oil,
//           // i.e., < 0.306 kg/kWh
//             mNot Fuel=P.Vcod(i,"FUEL-TYPE");    // same as heating
fuel in the actual building
//
//      } // end else

      // Save the heating fuel type
      {
        ostringstream   res;   res   <<   FuelNot2010(P.Vcodid(i,"HEAT-
SOURCE"), P.Vcodid(i,"FUEL-TYPE"), P.Vcod(i,"FUEL-TYPE"));
        KV.Reset(); KV.Init(4,res.str());
        Temp.Prop.insert(make pair("FUEL-TYPE",KV));
      }
    } // end if

    else { // i.e., not 2010 notional
      //if gas is used anywhere in the actual building, set notional
fuel to gas, otherwise use oil
      //keyword   NOT-FUEL   is   set   to   gas   in   funtions
DHW GENERATOR::Init fuel() and HVAC::Init fuel() if there is gas in
the actual building
      if (P.Vcod(gId,"NOT-FUEL")=="GAS")
         {
          ostringstream res; res << "Natural Gas";
          KV.Reset(); KV.Init(4,res.str());
          Temp.Prop.insert(make_pair("FUEL-TYPE",KV));
         }
      else
         {
          ostringstream res; res << "Oil";
          KV.Reset(); KV.Init(4,res.str());
          Temp.Prop.insert(make_pair("FUEL-TYPE",KV));
         }

      // Determine if the fuel type for this HVAC system in the
actual building is grid electricity if the calculation is
      // for Scotland so that the keyword can be accessed from within
the notional zones in order to trigger the
      // adjustment to the heating demand of the 2010 Scottish
notional building
      if((P.Vcodid(cId,"BR-TYPE")==3)      ||      (P.Vcodid(cId,"EPC-
```

```
TYPE")==4)) {   //for Scotland only
        if   (P.Vcodid(i,"FUEL-TYPE")==10)   //   10=Grid   Supplied
Electricity
         {
          ostringstream res; res << "YES";  // actual building's fuel
type for this HVAC is electricity
          KV.Reset(); KV.Init(4,res.str());
          Temp.Prop.insert(make_pair("ACT-ELEC-HEAT",KV));
         }
      } // end if
    } // end else

  } // end else

 // The notional fuel for cooling is always grid electricity
 {
  ostringstream res; res << "Grid Supplied Electricity";
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("FUEL-TYPE-COOL",KV));
 }


 {
  ostringstream res;
 // Check if this is the 2010 notional as it affects the cooling
efficiency which in this case, depends on the DRIVER2A
 // keyword of the zone in the actual building. As such, the
efficiency is actually determined at zone level. The HVAC
 // parameters, however, are filled in below for reporting in the
data reflection report and the HVAC-SYSTEM-DATA object
 // for the Technical Data Sheet in BRUKL.

 if (P.Vint(gId,"NOT-CASE")>=9)   { // 9=EW NOT 2010
   // if the HVAC system operates mixed-mode cooling
   if (P.Vcod(Id,"MIXED-MODE")=="YES")
     res  <<  T.Value("Not Space Cool SEER",2,P.Vint(gId,"NOT-CASE"));
// mixed-mode SEER
   else // i.e., not mixed-mode
     res  <<  T.Value("Not_Space_Cool_SEER",1,P.Vint(gId,"NOT-CASE"));
// regular SEER
 } // end if

 else { // i.e., not 2010 notional
  // Modified to allow Notional building with Mixed-Mode operation
strategy.
  // Check all together with the rest of the new code for assignment
of efficiencies.
  if(P.Vcod(i,"MIXED-MODE")=="YES")
    res << T.Value("Not HVAC Eff",4,1);
  else
    res                                                              <<
T.Value("Not HVAC Eff",T.Vint("tNOT GROUP",P.Vcodid(i,"TYPE"),P.Vint(
gId,"NOT-CASE")),1); // JO 24 Feb 09

 } // end else
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make_pair("COOL-SSEER",KV));
 }

//use correct notional heating eff
//this is modified in zone if you don't have cooling and have mech
ventilation
 {
  ostringstream res;
 // Check if this is the 2010 notional as it affects the heating
efficiency which in this case, depends on the DRIVER2A
 // keyword of the zone and the heating fuel of the HVAC system in
the actual building. As such, the efficiency is actually
 // determined at zone level. The HVAC parameters, however, are
```

```
filled in below for reporting in the data reflection report
 // and the HVAC-SYSTEM-DATA object for the Technical Data Sheet in
BRUKL.
 if (P.Vint(gId,"NOT-CASE")>=9)   { // 9=EW NOT 2010
   //--- Define efficiency
   // check if the heat source in the actual building is provided by
an electric heat pump
   // 18=Heat pump (electric): air source , 19=Heat pump (electric):
ground or water source
   // 10=Grid Supplied Electricity
   if   (   ((P.Vcodid(Id,"HEAT-SOURCE")==18)   ||   (P.Vcodid(Id,"HEAT-
SOURCE")==19)) && (P.Vcodid(Id,"FUEL-TYPE")==10) )
     res     <<     T.Value("Not Space Heat SEff 4",1,P.Vint(gId,"NOT-
CASE"));

   else { // i.e., not an electric heat pump
     // check if the heat source in the actual building is a non-
electric heat pump
     // 12=Heat pump (gas/oil): air source , 13=Heat pump (gas/oil):
ground or water source
     if   ((P.Vcodid(Id,"HEAT-SOURCE")==12)   ||   (P.Vcodid(Id,"HEAT-
SOURCE")==13))
       res    <<    T.Value("Not Space Heat SEff 3",1,P.Vint(gId,"NOT-
CASE"));

     else { // i.e., heat source is not a heat pump
       // Check if the heating fuel and heat source in the actual
building is district heating
       // 13=District Heating , 15=District Heating
       if   ((P.Vcodid(Id,"FUEL-TYPE")==13)   &&   (P.Vcodid(Id,"HEAT-
SOURCE")==15) ) {
         // Heating efficiency depends on the emission factor input
by the user for district heating
         if ((P.Vnum(gId,"DH-CO2F"))<D.Value("dNOT DISTRICT HEAT1"))
           res << T.Value("Not Space Heat SEff 1",1,9);
           else {
             if                                          ((P.Vnum(gId,"DH-
CO2F"))>D.Value("dNOT DISTRICT HEAT2"))
               res << T.Value("Not_Space_Heat_SEff_2",1,9);
             else // i.e., 0.15 <= emission factor <= 0.2
               res           <<           (((P.Vnum(gId,"DH-CO2F"))-
D.Value("dNOT_DISTRICT_HEAT1"))*(T.Value("Not_Space_Heat_SEff_2",1,9)
-
T.Value("Not Space Heat SEff 1",1,9))/(D.Value("dNOT DISTRICT HEAT2")
-
D.Value("dNOT DISTRICT HEAT1")))+T.Value("Not Space Heat SEff 1",1,9)
;
           } // end else
       } // end if

       else { // i.e., fuel type is not district heating
         // Check if the main heating fuel in the actual building is
one whose emission factor <= 0.156 kg/kWh
         // 3=Biogas, 9=Biomass, 12=Waste heat
         if  (  (P.Vcodid(Id,"FUEL-TYPE")==3)  ||  (P.Vcodid(Id,"FUEL-
TYPE")==9) || (P.Vcodid(Id,"FUEL-TYPE")==12) )
           res  <<  T.Value("Not_Space_Heat_SEff_1",1,P.Vint(gId,"NOT-
CASE"));
         else // i.e., the main heating fuel in the actual building
is one whose emission factor > 0.156 kg/kWh
           res  <<  T.Value("Not Space Heat SEff 2",1,P.Vint(gId,"NOT-
CASE"));
       } // end else
     } // end else

   } // end else

 } // end if
```

```
 else { // i.e., not 2010 notional
   if(P.Vcod(i,"MIXED-MODE")=="YES")
     // JO 24 Feb 09: New tables => new code
     res                                                    <<
T.Value("Not_HVAC_Eff",5+3+T.Vint("tNOT_GROUP",P.Vcodid(i,"TYPE"),P.V
int(gId,"NOT-CASE")),1); // JO 24 Feb 09
   else
     // JO 24 Feb 09: New tables => new code
     res                                                    <<
T.Value("Not HVAC Eff",5+T.Vint("tNOT GROUP",P.Vcodid(i,"TYPE"),P.Vin
t(gId,"NOT-CASE")),1); // JO 24 Feb 09
 } // end else

  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("HEAT-SSEFF",KV));
 }

//use correct notional aux energy
//change this in zone: if you don't have cooling and have mech
ventilation

// For 2010 notional, there is heat recovery if there is mechanical
ventilation in the zone, but because it depends
// on the DRIVER2A keyword for the ZONE, the heat recovery for this
type of notional building is re-set at zone level
 {
  ostringstream               res;               res              <<
T.TValue("Not Heat Recovery Sys",1,P.Vint(gId,"NOT-CASE"));
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("HEAT-REC-SYSTEM",KV));
 }
//v2.0.a - now using heat recovery efficiency from iSBEM: keyword
HEAT-REC-SEFF
//for no heat recovery HEAT-REC-SEFF=0
 {
  ostringstream               res;               res              <<
T.Value("Not Heat Recovery Seff",1,P.Vint(gId,"NOT-CASE"));
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make_pair("HEAT-REC-SEFF",KV));
 }
 {
  ostringstream               res;               res              <<
T.TValue("Not Var Heat Recovery",1,P.Vint(gId,"NOT-CASE"));
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make_pair("HEAT-REC-VAR-EFF",KV));
 }
 {
  ostringstream res; res << P.Vnum(i,"HEAT-GEN-SEFF");
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("HEAT-GEN-SEFF",KV));
 }
 {
  ostringstream res; res << P.Vnum(i,"COOL-GEN-SEER");
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("COOL-GEN-SEER",KV));
 }
 {
  ostringstream res; res << P.Vnum(i,"MULTIPLIER");
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make_pair("MULTIPLIER",KV));
 }

// Copy Mixed-mode strategy over Notional building
 {
  ostringstream res; res << P.Vcod(i,"MIXED-MODE");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("MIXED-MODE",KV));
 }
```

```
// For auxiliary energy we set the central SFP at HVAC level
 {
  ostringstream res; res << T.Value("tNOT SFP",1,P.Vint(gId,"NOT-
CASE"));
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("SFP",KV));
 }
 {
  ostringstream res; res <<
T.Value("Not HVAC Metering",1,P.Vint(gId,"NOT-CASE"));
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("METERING",KV));
 }

 // Copy over details of the bivalent systems for this HVAC to the
notional building
 {
  ostringstream res; res << P.Vtext(i,"BH-HEAT-SOURCE");
  KV.Reset(); KV.Init(5,res.str());
  Temp.Prop.insert(make_pair("BH-HEAT-SOURCE",KV));
 }
 {
  ostringstream res; res << P.Vtext(i,"BH-FUEL-TYPE");
  KV.Reset(); KV.Init(5,res.str());
  Temp.Prop.insert(make pair("BH-FUEL-TYPE",KV));
 }
 {
  ostringstream res; res << P.Vtext(i,"BH-HEAT-SSEFF");
  KV.Reset(); KV.Init(5,res.str());
  Temp.Prop.insert(make pair("BH-HEAT-SSEFF",KV));
 }
 {
  ostringstream res; res << P.Vtext(i,"BH-HEAT-GEN-SEFF");
  KV.Reset(); KV.Init(5,res.str());
  Temp.Prop.insert(make pair("BH-HEAT-GEN-SEFF",KV));
 }
 {
  ostringstream res; res << P.Vtext(i,"BH-LOAD");
  KV.Reset(); KV.Init(5,res.str());
  Temp.Prop.insert(make pair("BH-LOAD",KV));
 }

  N.vINP.push back(Temp);
}


void HVAC::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC& P,
PROJECTTables& HVG) {
  if(func report) cout << endl << "  " << "HVAC::Calc";
  if(calc report) cout << endl << P.vINP[Id].Name;
//calls functions for HVAC calculations

  for (int i=0;i<vZ.size();i++) {              // Before the whole
calculation. P_EFF_AGR has to be calculated
    //SPACE::fVENT ZONE HVAC() sets heat recovery option for zone
from parent HVAC and ventilation type keyword: "Q-V-TYPE"
    vZ[i].fVENT_ZONE_HVAC(D,T,P);
    //sets keyword "F-VENT-M"
    //sets ventilation flow rate keywords for zone: "U-V-M" for mech
ventilation and "U-V-N" for natural ventilation
    //these are air flow rates in l/s/m2
    vZ[i].fU V M(D, T, P);
  }

  //temperature gradient correction for HVAC system:
  fK_GRAD(D, T, P);
  //calculates area served by HVAC - sum of [zone areas multiplied by
zone multiplier]
  fAREA(D, T, P);
```

```
  //carry out calculation for each zone served by HVAC system
  for (int i=0;i<vZ.size();i++) {
    vZ[i].Calc(D, T, P);

    { ostringstream logtext; logtext << "."; SendLog(logtext.str());
}
  }

//calculations for this HVAC system:

//new function - finds system numbers from HVAC guide (for actual
building only)
if (P.Vcod(gId,"ACT-NOT")=="ACT") fHVAC_GUIDE_NO(D, T, P, HVG);
//calculate energy required for heating:
//following function now calls zone function and adds up the energy
calculated at zone level:
  fQ PRIM HEAT(D, T, P);
//calculate energy required for cooling:
//following function now calls zone function and adds up the energy
calculated at zone level:
  fQ PRIM COOL(D, T, P);
//calculate auxiliary energy required:
//following function adds up the energy calculated at zone level:
  fQ AUX(D, T, P);
//calculate energy required for equipment (info only, not included in
part L calc):
//following function now calls zone function and adds up the energy
calculated at zone level:
  fQ PRIM EQUP(D, T, P);
//calculate energy required for lighting:
//following function now calls zone function and adds up the energy
calculated at zone level:
  fQ PRIM LI(D, T, P);
//total energy required by fuel type:
//following function now calls zone function and adds up the energy
calculated at zone level:
  fFUEL TYPE(D, T, P);
}

void HVAC::fSolarGainCheck(PROJECTDefaults& D, PROJECTTables& T, BOC&
P, PROJECTTables& HVG) {
  if(func report) cout << endl << "  " << "HVAC::fSolarGainCheck";
  if(calc report) cout << endl << P.vINP[Id].Name;
  //carry out calculation for each zone served by HVAC system
  for (int i=0;i<vZ.size();i++) {
    vZ[i].fSolarGainCheck(D, T, P);
    { ostringstream logtext; logtext << "."; SendLog(logtext.str());
}
  }
}

double  HVAC::fAUX ENERGY CORR(PROJECTDefaults&  D,  PROJECTTables&  T,
BOC& P) {
  if(func report) cout << endl << "  " << "HVAC::fAUX ENERGY CORR";
//returns auxiliary energy for notional HVAC

  double mAUX ENERGY CORR=0.0;
  double mAUX ENERGY=0.0;
  double mAREA=P.Vnum(Id,"AREA");

   // JO 23Sep07 Modified to include Reference and Typical building
efficiencies through
   // keyword NOT-TYPE of general object
   //                                                        double
mAUX_ENERGY_DEF=T.Value("Not_HVAC_Eff",10+T.Vint("tNOT_GROUP",P.Vcodi
d(Id,"TYPE"),1),1);
   // JO 24 Feb 09: New tables => new code
  //double
```

```
mAUX ENERGY DEF=T.Value("Not HVAC Eff",T.Vint("tNOT GROUP",P.Vcodid(I
d,"TYPE"),1)+5*(P.Vcodid(gId,"NOT-TYPE")-1),3);
  double
mAUX ENERGY DEF=T.Value("Not HVAC Eff",T.Vint("tNOT GROUP",P.Vcodid(I
d,"TYPE"),P.Vint(gId,"NOT-CASE"))+5*(3-1),P.Vint(gId,"NOT-CASE"));
// JO 24 Feb 09
  //notional HVAC efficiency for heated and mechanically ventilated
zones:
//  double mAUX ENERGY 3=T.Value("Not HVAC Eff",3,3);
   // JO 24 Feb 09: New tables => new code
  //double
mAUX ENERGY 3=T.Value("Not HVAC Eff",3+5*(P.Vcodid(gId,"NOT-TYPE")-
1),3);
  double                  mAUX_ENERGY_3=T.Value("Not_HVAC_Eff",3+5*(3-
1),P.Vint(gId,"NOT-CASE"));
  for (int i=0;i<vZ.size();i++) {
    if(T.Vint("tNOT_GROUP",P.Vcodid(Id,"TYPE"),P.Vint(vZ[i].Id,"NOT-
CASE"))==1) {       //if zone has either mechanical supply or extract
serving more than one room:
      //if              ((P.Vcod(vZ[i].Id,"Q-V-TYPE")=="Mechanical")||
((P.Vcod(vZ[i].Id,"VENT-MECH-EXH")=="YES")                            &&
(P.Vcodid(vZ[i].Id,"VENT-MECH-EXH-NCM")==1)))

      if ((P.Vcod(vZ[i].Id,"Q-V-TYPE")=="Mechanical"))
        mAUX ENERGY=mAUX ENERGY 3;
      else
        mAUX ENERGY=mAUX ENERGY DEF;
    }
    else
      mAUX ENERGY=mAUX ENERGY DEF;

    // Note that condition is the negation of the following in order
to include only the as built zones
    // 1=YES , 1=None , 2=As built , 1=YES
    if(!(P.Vcodid(cId,"SHELL-AND-CORE")==1    &&    P.Vcodid(cId,"EPC-
TYPE")==1 && P.Vcodid(cId,"BR-STAGE")==2 && P.Vcodid(vZ[i].Id,"SHELL-
ASSUMED")==1))

mAUX_ENERGY_CORR+=(P.Vnum(vZ[i].Id,"AREA")*P.Vnum(vZ[i].Id,"MULTIPLIE
R")/mAREA*mAUX ENERGY);
  }
  return(mAUX_ENERGY_CORR);
}


void HVAC::fK GRAD(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "   " << "HVAC::fK GRAD";
//temperature gradient correction for HVAC system

 double mGRAD=0.0;

 mGRAD=T.Value("tGRAD",1,P.Vcodid(Id,"TYPE"));

 // Keep temperature gradient in notional building as the default 0
W/K. The value will be later changed at zone level
 // depending on whether the zone is side-lit/not-lit or top-lit.
// if (P.Vcod(gId,"ACT-NOT")=="NOT") {
//   mGRAD=0.0;
// }

 {
  ostringstream res; res << mGRAD;   // in W/K
  P.ModKey(Id,"GRAD",res.str());
 }
}


void HVAC::fAREA(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "   " << "HVAC::fAREA";
```

```
//calculates area served by HVAC - sum of [zone areas multiplied by
zone multiplier]

 double mAREA=0;
 for (int i=0;i<vZ.size();i++)
   // Note that condition is the negation of the following in order
to include only the as built zones
   // 1=YES , 1=None , 2=As built , 1=YES
   if(!(P.Vcodid(cId,"SHELL-AND-CORE")==1    &&    P.Vcodid(cId,"EPC-
TYPE")==1 && P.Vcodid(cId,"BR-STAGE")==2 && P.Vcodid(vZ[i].Id,"SHELL-
ASSUMED")==1))
     mAREA+=P.Vnum(vZ[i].Id,"AREA")*P.Vnum(vZ[i].Id,"MULTIPLIER");
 {
  ostringstream res; res << mAREA;
  P.ModKey(Id,"AREA",res.str());
 }

}


void HVAC::fHVAC GUIDE NO(PROJECTDefaults& D, PROJECTTables& T, BOC&
P, PROJECTTables& HVG) {
  if(func report) cout << endl << "  " << "HVAC::fHVAC GUIDE NO";
//writes 3 keywords to represent the HVAC guide numbers for the
system, these are passed to BRUKL:
//HVACGUIDE-HEAT (heating), HVACGUIDE-COOL (cooling), HVACGUIDE-SFP
(SFP)
//HVACGUIDE-DHW (DHW) is a DHW-GENERATOR keyword, and is done in dhw-
generator.cpp

  int mSYSTEM TYPE;
  int mHEAT SOURCE;
  int mFUEL TYPE;
  double mHVACGUIDE HEAT=-5555;
  double mHVACGUIDE COOL=-5555;
  double mHVACGUIDE SFP=-5555;

//HEATING
//find HVAC guide heating system number (will be -5555 if system is
not in the HVAC guide, or there is no min eff defined in HVAC guide)
//only do this if system has heating:

  if (T.Value("tSYS HEAT",1,P.Vcodid(Id,"TYPE"))==1) {

    mSYSTEM_TYPE=P.Vcodid(Id,"TYPE");
    mHEAT SOURCE=P.Vcodid(Id,"HEAT-SOURCE");
    mFUEL TYPE=P.Vcodid(Id,"FUEL-TYPE");

//use HVAC system type, heat source and fuel type IDs to create
unique table name:
//table name is prefixed "tHVACGUIDE_HEAT":
    string HVACGUIDE HEAT name="tHVACGUIDE HEAT";
    char HVACGUIDE HEAT name char[20];
    char *line = "_";

//add system type Id to table name:
    sprintf(HVACGUIDE_HEAT_name_char,"%s%d","",mSYSTEM_TYPE);
    { string s(HVACGUIDE HEAT name char); HVACGUIDE HEAT name+=s; }
//add underscore then heat source Id to table name:
    sprintf(HVACGUIDE_HEAT_name_char,"%s%d",line,mHEAT_SOURCE);
    { string s(HVACGUIDE HEAT name char); HVACGUIDE HEAT name+=s; }
//add underscore then fuel type Id to table name:
    sprintf(HVACGUIDE_HEAT_name_char,"%s%d",line,mFUEL_TYPE);
    { string s(HVACGUIDE HEAT name char); HVACGUIDE HEAT name+=s; }

//you now have the complete table name to look up in the HVACdefault
tables:
//the table has the following information (1 row and either 2 or 3
columns):
```

```
//-first column:
//       value=1->no extra checks (your system type, heat source and
fuel type combination gives you a unique HVAC guid system and no
further checks need to be carried out),
//       value=2->check heat recovery (the HVAC guide type depends on
whether system has heat recovery or not)
//       value=3->check chiller type (the HVAC guide type depends on
type of chiller, only air and water-cooled are considered in the HVAC
guide, remote condenser falls outside guide)
//-this is followed by either (if no extra checks) 1 value of HVAC
guide number or (if there are any extra checks) 2 values of HVAC
guide number

    if                          (HVG.Value(HVACGUIDE HEAT name,1,1)==1)
mHVACGUIDE HEAT=HVG.Value(HVACGUIDE HEAT name,1,2);

  }
  else {
    mHVACGUIDE HEAT=-1111;
  }

 {
  ostringstream res; res << mHVACGUIDE HEAT;
  P.ModKey(Id,"HVACGUIDE-HEAT",res.str());
 }

//COOLING
//find HVAC guide cooling system number (will be -5555 if system is
not in the HVAC guide, or there is no min eff defined in HVAC guide)
//only do this if system has cooling:
  if (T.Value("tSYS COOL",1,P.Vcodid(Id,"TYPE"))==1) {

    mSYSTEM TYPE=P.Vcodid(Id,"TYPE");
    mHEAT SOURCE=P.Vcodid(Id,"HEAT-SOURCE");
    mFUEL TYPE=P.Vcodid(Id,"FUEL-TYPE");

//use HVAC system type, heat source and fuel type IDs to create
unique table name:
//table name is prefixed "tHVACGUIDE_COOL":
    string HVACGUIDE COOL name="tHVACGUIDE COOL";
    char HVACGUIDE COOL name char[20];
    char *line = "_";

//add system type Id to table name:
    sprintf(HVACGUIDE_COOL_name_char,"%s%d","",mSYSTEM_TYPE);
    { string s(HVACGUIDE COOL name char); HVACGUIDE COOL name+=s; }
//add underscore then heat source Id to table name:
    sprintf(HVACGUIDE_COOL_name_char,"%s%d",line,mHEAT_SOURCE);
    { string s(HVACGUIDE COOL name char); HVACGUIDE COOL name+=s; }
//add underscore then fuel type Id to table name:
    sprintf(HVACGUIDE_COOL_name_char,"%s%d",line,mFUEL_TYPE);
    { string s(HVACGUIDE COOL name char); HVACGUIDE COOL name+=s; }

//you now have the complete table name to look up in the HVACdefault
tables:
//the table has the following information (1 row and either 2 or 3
columns):
//-first column:
//       value=1->no extra checks (your system type, heat source and
fuel type combination gives you a unique HVAC guid system and no
further checks need to be carried out),
//       value=2->check heat recovery (the HVAC guide type depends on
whether system has heat recovery or not)
//       value=3->check chiller type (the HVAC guide type depends on
type of chiller, only air and water-cooled are considered in the HVAC
guide, remote condenser falls outside guide)
//-this is followed by either (if no extra checks) 1 value of HVAC
guide number or (if there are any extra checks) 2 values of HVAC
guide number
```

```
    if (HVG.Value(HVACGUIDE COOL name,1,1)==1)
      mHVACGUIDE COOL=HVG.Value(HVACGUIDE COOL name,1,2);
    else if (HVG.Value(HVACGUIDE COOL name,1,1)==3) {
    //check chiller type
      if   (P.Vcodid(Id,"CHILLER-TYPE")==1   ||   P.Vcodid(Id,"CHILLER-
TYPE")==5)
        mHVACGUIDE COOL=HVG.Value(HVACGUIDE COOL name,1,2);      //air
cooled
      else if (P.Vcodid(Id,"CHILLER-TYPE")==2)
        mHVACGUIDE COOL=HVG.Value(HVACGUIDE COOL name,1,3);    //water
cooled
      else
        mHVACGUIDE_COOL=-5555; //remote condenser chiller does not
have a minimum efficiency in HVAC guide
    }
  }
  else {
    mHVACGUIDE COOL=-1111;  //no cooling
  }

   {
    ostringstream res; res << mHVACGUIDE COOL;
    P.ModKey(Id,"HVACGUIDE-COOL",res.str());
   }

//SFP
//may also need to do SFP check at zone level - if so add this to
zone.cpp later, not used now
//find HVAC guide SFP system number
//set to -1111 if HVAC has no ventilation associated with it
//will be -5555 if system has ventilation but system is not in the
HVAC guide, or there is no min eff defined in HVAC guide
//SFP-CHECK is a new keyword that is set to YES where there is HVAC
level ventilation, default is NO
  if (P.Vcod(Id,"SFP-CHECK")=="YES") {

    mSYSTEM TYPE=P.Vcodid(Id,"TYPE");
    mHEAT_SOURCE=P.Vcodid(Id,"HEAT-SOURCE");
    mFUEL_TYPE=P.Vcodid(Id,"FUEL-TYPE");

//use HVAC system type, heat source and fuel type IDs to create
unique table name:
//table name is prefixed "tHVACGUIDE SFP":
    string HVACGUIDE_SFP_name="tHVACGUIDE_SFP";
    char HVACGUIDE SFP name char[20];
    char *line = " ";

//add system type Id to table name:
    sprintf(HVACGUIDE SFP name char,"%s%d","",mSYSTEM TYPE);
    { string s(HVACGUIDE_SFP_name_char); HVACGUIDE_SFP_name+=s; }
//add underscore then heat source Id to table name:
    sprintf(HVACGUIDE SFP name char,"%s%d",line,mHEAT SOURCE);
    { string s(HVACGUIDE_SFP_name_char); HVACGUIDE_SFP_name+=s; }
//add underscore then fuel type Id to table name:
    sprintf(HVACGUIDE SFP name char,"%s%d",line,mFUEL TYPE);
    { string s(HVACGUIDE_SFP_name_char); HVACGUIDE_SFP_name+=s; }

//you now have the complete table name to look up in the HVACdefault
tables:
//the table has the following information (1 row and either 2 or 3
columns):
//-first column:
//       value=1->no extra checks (your system type, heat source and
fuel type combination gives you a unique HVAC guid system and no
further checks need to be carried out),
//       value=2->check heat recovery (the HVAC guide type depends on
whether system has heat recovery or not)
//       value=3->check chiller type (the HVAC guide type depends on
```

```
type of chiller, only air and water-cooled are considered in the HVAC
guide, remote condenser falls outside guide)
//-this is followed by either (if no extra checks) 1 value of HVAC
guide number or (if there are any extra checks) 2 values of HVAC
guide number

    if (HVG.Value(HVACGUIDE SFP name,1,1)==1)
      mHVACGUIDE_SFP=HVG.Value(HVACGUIDE_SFP_name,1,2);
    else if (HVG.Value(HVACGUIDE SFP name,1,1)==2) {
    //check heat recovery
      if (P.Vcodid(Id,"HEAT-REC-SYSTEM")==1)
        mHVACGUIDE SFP=HVG.Value(HVACGUIDE SFP name,1,2);  //no  heat
recovery
      else
        mHVACGUIDE SFP=HVG.Value(HVACGUIDE SFP name,1,3);      //heat
recovery
    }
  }
  else {
    mHVACGUIDE_SFP=-1111; //no HVAC ventilation
  }

  {
  ostringstream res; res << mHVACGUIDE SFP;
  P.ModKey(Id,"HVACGUIDE-SFP",res.str());
  }

}


void HVAC::fQ PRIM HEAT(PROJECTDefaults& D, PROJECTTables& T, BOC& P)
{
  if(func report) cout << endl << "  " << "HVAC::fQ PRIM HEAT";
//calculations now carried out at zone level
//this function now calls the zone function and adds up all the
figures from zone level

//v2.0.a, calculation moved from HVAC to zone level:
  for (int i=0;i<vZ.size();i++)
    vZ[i].fQ PRIM HEAT(D, T, P);

 double mQ_DEM_HEAT=0.0;
 double mQ DEM HEAT M[12]= { 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };

 //heat demand for zone:
 deque<double> mQ DEM HEAT M Z;

 //for each zone served by HVAC system:
 for(int i=0;i<vZ.size();i++) {
   //heat demand for zone for each month (12 values)
   mQ DEM HEAT M Z=P.Vlnum Mul(vZ[i].Id,"Q-DEM-HEAT-
M",P.Vnum(vZ[i].Id,"MULTIPLIER"));
   //for each month of year:
   for(int m=0;m<12;m++) {
     //add zone demand to HVAC demand for that month:
     mQ_DEM_HEAT_M[m]+=mQ_DEM_HEAT_M_Z[m];
     //add zone demand to annual HVAC demand:
     mQ DEM HEAT+=mQ DEM HEAT M Z[m];
    }
 }

 //store heating demand for HVAC system
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ DEM HEAT M[i] << ", ";
  res << mQ DEM HEAT M[11] << " }";
  P.ModKey(Id,"Q-DEM-HEAT-M",res.str());
```

```
 }

 {
  ostringstream res; res << mQ DEM HEAT;
  P.ModKey(Id,"Q-DEM-HEAT",res.str());
 }

 //primary energy for heating for HVAC system
 double mQ PRIM HEAT=0.0;
 double mQ PRIM HEAT M[12]={ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };
 double mQ PRIM HEAT CHP=0.0;
 double mQ PRIM HEAT CHP M[12]={ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0 };
 deque<double> mQ PRIM HEAT M Z;
 deque<double> mQ PRIM HEAT CHP M Z;

 for(int i=0;i<vZ.size();i++) {
   // 18 April 2010 JO: Shell and core
   //if(P.Vcodid(vZ[i].Id,"SHELL-ASSUMED")==2) {
   if(!(P.Vcodid(cId,"SHELL-AND-CORE")==1    &&    P.Vcodid(cId,"EPC-
TYPE")==1 && P.Vcodid(cId,"BR-STAGE")==2 && P.Vcodid(vZ[i].Id,"SHELL-
ASSUMED")==1)) {
     //zone primary energy for heating for each month (12 values)
     mQ PRIM HEAT M Z=P.Vlnum Mul(vZ[i].Id,"Q-PRIM-HEAT-
M",P.Vnum(vZ[i].Id,"MULTIPLIER"));
     mQ PRIM HEAT CHP M Z=P.Vlnum Mul(vZ[i].Id,"Q-PRIM-HEAT-CHP-
M",P.Vnum(vZ[i].Id,"MULTIPLIER"));
     //for each month of year:
     for(int m=0;m<12;m++) {
       //add zone primary energy to HVAC primary energy for that
month:
       mQ PRIM HEAT M[m]+=mQ PRIM HEAT M Z[m];
       //add zone primary energy to annual HVAC primary energy:
       mQ PRIM HEAT+=mQ PRIM HEAT M Z[m];
       mQ PRIM HEAT CHP M[m]+=mQ PRIM HEAT CHP M Z[m];
       //mQ PRIM HEAT CHP+=mQ PRIM HEAT CHP M[m];  //  JO  17Jan08  I
think this is wrong it should be as below. Azza, could you confirm
please?. // AN, confirmed
       mQ PRIM HEAT CHP+=mQ PRIM HEAT CHP M Z[m];
     }
   }
 }

 //store primary energy for heating for HVAC system, monthly and
annual figures in MJ and kWh
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ PRIM HEAT M[i] << ", ";
  res << mQ_PRIM_HEAT_M[11] << " }";
  P.ModKey(Id,"Q-PRIM-HEAT-M",res.str());
 }
 {
  ostringstream res; res << mQ PRIM HEAT;
  P.ModKey(Id,"Q-PRIM-HEAT",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ PRIM HEAT M[i]/3.6 << ", ";
  res << mQ PRIM HEAT M[11]/3.6 << " }";
  P.ModKey(Id,"KWH-HEAT-M",res.str());
 }
 {
  ostringstream res; res << mQ_PRIM_HEAT/3.6;
  P.ModKey(Id,"KWH-HEAT",res.str());
 }
  if((P.Vcodid(cId,"BR-TYPE")==5) || (P.Vcodid(cId,"EPC-TYPE")==6))
```

```
      {
       ostringstream              res;              res              <<
mQ PRIM HEAT/3.6*T.Value("tKGCO2 IR",1,P.Vcodid(Id,"FUEL-TYPE"));
       P.ModKey(Id,"KG-CO2-HEAT",res.str());
      }
   else if((P.Vcodid(cId,"BR-TYPE")==6))  //Jersey only
      {
       ostringstream              res;              res              <<
mQ PRIM HEAT/3.6*T.Value("tKGCO2 JER",1,P.Vcodid(Id,"FUEL-TYPE"));
       P.ModKey(Id,"KG-CO2-HEAT",res.str());
      }
   else // all UK
      {
       ostringstream              res;              res              <<
mQ PRIM HEAT/3.6*T.Value("tKGCO2",1,P.Vcodid(Id,"FUEL-TYPE"));
       P.ModKey(Id,"KG-CO2-HEAT",res.str());
      }
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mQ PRIM HEAT CHP M[i] << ", ";
   res << mQ PRIM HEAT CHP M[11] << " }";
   P.ModKey(Id,"Q-PRIM-HEAT-CHP-M",res.str());
  }
  {
   ostringstream res; res << mQ PRIM HEAT CHP;
   P.ModKey(Id,"Q-PRIM-HEAT-CHP",res.str());
  }

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mQ PRIM HEAT CHP M[i]/3.6 << ", ";
   res << mQ PRIM HEAT CHP M[11]/3.6 << " }";
   P.ModKey(Id,"KWH-HEAT-CHP-M",res.str());
  }
  {
   ostringstream res; res << mQ PRIM HEAT CHP/3.6;
   P.ModKey(Id,"KWH-HEAT-CHP",res.str());
  }

}

void HVAC::fQ PRIM COOL(PROJECTDefaults& D, PROJECTTables& T, BOC& P)
{
  if(func report) cout << endl << "  " << "HVAC::fQ PRIM COOL";
//calculations now carried out at zone level
//this function now calls zone function and adds up all the figures
from zone level

//v2.0.a, calculation moved from HVAC to zone level:
  for (int i=0;i<vZ.size();i++)
    vZ[i].fQ PRIM COOL(D, T, P);

 double mQ DEM COOL=0;
 double mQ DEM COOL M[12]= { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };

 //calculate cooling demand for HVAC system by summing cooling demand
for all zones served by HVAC system
 deque<double> mQ_DEM_COOL_M_Z;

 //for each zone served by HVAC system:
 for(int i=0;i<vZ.size();i++) {
   //cooling for this zone
   mQ DEM COOL M Z=P.Vlnum Mul(vZ[i].Id,"Q-DEM-COOL-
M",P.Vnum(vZ[i].Id,"MULTIPLIER"));
   //add monthly values to HVAC cooling demand, monthly and annual:
   for(int m=0;m<12;m++) {
     mQ_DEM_COOL_M[m]+=mQ_DEM_COOL_M_Z[m];
```

```
      mQ DEM COOL+=mQ DEM COOL M Z[m];
    }
  }

  //store cooling demand, monthly and annual:
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mQ DEM COOL M[i] << ", ";
   res << mQ DEM COOL M[11] << " }";
   P.ModKey(Id,"Q-DEM-COOL-M",res.str());
  }

  {
   ostringstream res; res << mQ DEM COOL;
   P.ModKey(Id,"Q-DEM-COOL",res.str());
  }

  double mQ PRIM COOL=0;
  double mQ_PRIM_COOL_M[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };

  //calculate primary cooling energy for HVAC system by summing
cooling primary energy  for all zones served by HVAC system
  deque<double> mQ PRIM COOL M Z;
  //for each zone served by HVAC system:
  for(int i=0;i<vZ.size();i++) {
    // 18 April 2010 JO: Shell and core
    //if(P.Vcodid(vZ[i].Id,"SHELL-ASSUMED")==2) {
    if(!(P.Vcodid(cId,"SHELL-AND-CORE")==1     &&     P.Vcodid(cId,"EPC-
TYPE")==1 && P.Vcodid(cId,"BR-STAGE")==2 && P.Vcodid(vZ[i].Id,"SHELL-
ASSUMED")==1)) {
      //cooling for this zone
      mQ PRIM COOL M Z=P.Vlnum Mul(vZ[i].Id,"Q-PRIM-COOL-
M",P.Vnum(vZ[i].Id,"MULTIPLIER"));
      //add monthly values to HVAC cooling demand, monthly and annual:
      for(int m=0;m<12;m++) {
        mQ PRIM COOL M[m]+=mQ PRIM COOL M Z[m];
        mQ PRIM COOL+=mQ PRIM COOL M Z[m];
      }
    }
  }
  //store primary energy for cooling for HVAC, monthly and annual
values in kWh and MJ
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mQ PRIM COOL M[i] << ", ";
   res << mQ_PRIM_COOL_M[11] << " }";
   P.ModKey(Id,"Q-PRIM-COOL-M",res.str());
  }
  {
   ostringstream res; res << mQ PRIM COOL;
   P.ModKey(Id,"Q-PRIM-COOL",res.str());
  }

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mQ PRIM COOL M[i]/3.6 << ", "; //3.6 used to convert from
MJ to kWh
   res << mQ PRIM COOL M[11]/3.6 << " }";
   P.ModKey(Id,"KWH-COOL-M",res.str());
  }
  {
   ostringstream res; res << mQ PRIM COOL/3.6; //3.6 used to convert
from MJ to kWh
   P.ModKey(Id,"KWH-COOL",res.str());
  }
   //write KG CO2 for cooling
```

```
  double mFUEL CARBON INTENSITY;

  // In the next line, the codid of the cooling fuel type is matched
to the codid of the full list of fuel types
  // (as in heating) so that the correct values of emission factors,
primary energy factors, etc, can be retrieved from
  // the tables.
  if  ((P.Vcodid(cId,"BR-TYPE")==5)  ||  (P.Vcodid(cId,"EPC-TYPE")==6))
// RoI calculations

mFUEL_CARBON_INTENSITY=T.Value("tKGCO2_IR",1,(int)T.Value("tFUEL_TYPE
 COOL",1,P.Vcodid(Id,"FUEL-TYPE-COOL")));
  else if ((P.Vcodid(cId,"BR-TYPE")==6))   // Jersey calculations

mFUEL CARBON INTENSITY=T.Value("tKGCO2 JER",1,(int)T.Value("tFUEL TYP
E COOL",1,P.Vcodid(Id,"FUEL-TYPE-COOL")));
  else  // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,(int)T.Value("tFUEL TYPE CO
OL",1,P.Vcodid(Id,"FUEL-TYPE-COOL")));

 {
  ostringstream res; res << mQ PRIM COOL/3.6*mFUEL CARBON INTENSITY;
  P.ModKey(Id,"KG-CO2-COOL",res.str());
 }

}


void HVAC::fQ AUX(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "HVAC::fQ AUX";
//adds auxiliary energy for zones served by HVAC system - auxiliary
energy for zones calculated at zone level

  double mQ AUX=0.0;
  deque<double> mQ AUX ZONE M;
  double mQ AUX M[12] ={0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };

  //for each zone served by HVAC system:
  for (int i=0;i<vZ.size();i++) {
    // 18 April 2010 JO: Shell and core
    //if(P.Vcodid(vZ[i].Id,"SHELL-ASSUMED")==2) {
    if(!(P.Vcodid(cId,"SHELL-AND-CORE")==1    &&    P.Vcodid(cId,"EPC-
TYPE")==1 && P.Vcodid(cId,"BR-STAGE")==2 && P.Vcodid(vZ[i].Id,"SHELL-
ASSUMED")==1)) {
      //monthly auxiliary energy for this zone
      mQ AUX ZONE M=P.Vlnum Mul(vZ[i].Id,"Q-AUX-
M",P.Vnum(vZ[i].Id,"MULTIPLIER"));  // MJ
      //for each month of year
      for(int j=0;j<12;j++) {
        //add zone auxiliary energy to HVAC total, monthly values and
annual
        mQ AUX M[j]+=mQ AUX ZONE M[j];    //monthly values
 //       mQ_AUX+=mQ_AUX_M[j];
 //  The line below has been amended from the commented line above -
AN - 15/01/2008
        mQ_AUX+=mQ_AUX_ZONE_M[j];
      }
    }
  }
 {
  ostringstream res; res << mQ AUX;
  P.ModKey(Id,"Q-AUX",res.str());  //MJ
 }

 {
  ostringstream res; res << mQ AUX/3.6;
  P.ModKey(Id,"KWH-AUX",res.str());   //KWh
 }
```

```
  //write KG CO2 for auxiliary
  if((P.Vcodid(cId,"BR-TYPE")==5) || (P.Vcodid(cId,"EPC-TYPE")==6))
     {
      ostringstream             res;             res             <<
mQ_AUX/3.6*T.Value("tKGCO2_IR",1,(int)D.Value("dELEC_CODID"));
      P.ModKey(Id,"KG-CO2-AUX",res.str());
     }
  else if((P.Vcodid(cId,"BR-TYPE")==6))
     {
      ostringstream             res;             res             <<
mQ_AUX/3.6*T.Value("tKGCO2_JER",1,(int)D.Value("dELEC_CODID"));
      P.ModKey(Id,"KG-CO2-AUX",res.str());
     }
  else // all UK
     {
      ostringstream             res;             res             <<
mQ_AUX/3.6*T.Value("tKGCO2",1,(int)D.Value("dELEC_CODID"));
      P.ModKey(Id,"KG-CO2-AUX",res.str());
     }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ AUX M[i]/3.6 << ", ";
  res << mQ AUX M[11]/3.6 << " }";
  P.ModKey(Id,"KWH-AUX-M",res.str());   //KWh
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ AUX M[i] << ", ";
  res << mQ AUX M[11] << " }";
  P.ModKey(Id,"Q-AUX-M",res.str());     //MJ
 }
}

void HVAC::fQ PRIM EQUP(PROJECTDefaults& D, PROJECTTables& T, BOC& P)
{
  if(func report) cout << endl << "  " << "HVAC::fQ PRIM EQUP";
//calculations now carried out at zone level
//this function now calls zone function and adds up all the figures
from zone level

//v2.0.a, calculation moved from HVAC to zone level:
  for (int i=0;i<vZ.size();i++)
    vZ[i].fQ_PRIM_EQUP(D, T, P);

  deque<double> mQ PRIM EQUP M Z;
  double mQ_PRIM_EQUP_M[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  double mQ PRIM EQUP=0;

  //for each zone served by HVAC system:
  for (int i=0;i<vZ.size();i++) {
    // 18 April 2010 JO: Shell and core
    //if(P.Vcodid(vZ[i].Id,"SHELL-ASSUMED")==2) {
    if(!(P.Vcodid(cId,"SHELL-AND-CORE")==1    &&    P.Vcodid(cId,"EPC-
TYPE")==1 && P.Vcodid(cId,"BR-STAGE")==2 && P.Vcodid(vZ[i].Id,"SHELL-
ASSUMED")==1)) {
      //primary energy required for equipment, sensible and latent
      mQ PRIM EQUP M Z=P.Vlnum Mul(vZ[i].Id,"Q-PRIM-EQUP-
M",P.Vnum(vZ[i].Id,"MULTIPLIER"));
      //for each month:
      for (int m=0;m<12;m++) {
        mQ_PRIM_EQUP_M[m]+=mQ_PRIM_EQUP_M_Z[m];
        mQ PRIM EQUP+=mQ PRIM EQUP M Z[m];
      }
    }
  }

 {
```

```
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ PRIM EQUP M[i] << ", ";
  res << mQ PRIM EQUP M[11] << " }";
  P.ModKey(Id,"Q-PRIM-EQUP-M",res.str());
 }
 {
  ostringstream res; res << mQ_PRIM_EQUP;
  P.ModKey(Id,"Q-PRIM-EQUP",res.str());
 }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ PRIM EQUP M[i]/3.6 << ", ";
  res << mQ PRIM EQUP M[11]/3.6 << " }";
  P.ModKey(Id,"KWH-EQUP-M",res.str());
 }
 {
  ostringstream res; res << mQ_PRIM_EQUP/3.6;
  P.ModKey(Id,"KWH-EQUP",res.str());
 }

}


void HVAC::fQ PRIM LI(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "HVAC::fQ PRIM LI";
//calculations now carried out at zone level
//this function now just adds up all the figures from zone level

//v2.0.a, function moved from HVAC to zone level:
  for (int i=0;i<vZ.size();i++)
    vZ[i].fQ PRIM LI(D, T, P);

  double mQ PRIM LI=0;
  double mQ PRIM LI M[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  deque<double> mQ PRIM LI M Z;

  //for each zone served by HVAC system:
  for (int i=0;i<vZ.size();i++) {
    // 18 April 2010 JO: Shell and core
    //if(P.Vcodid(vZ[i].Id,"SHELL-ASSUMED")==2) {
    if(!(P.Vcodid(cId,"SHELL-AND-CORE")==1    &&    P.Vcodid(cId,"EPC-
TYPE")==1 && P.Vcodid(cId,"BR-STAGE")==2 && P.Vcodid(vZ[i].Id,"SHELL-
ASSUMED")==1)) {
      //primary lighting energy for zone
      mQ_PRIM_LI_M_Z=P.Vlnum_Mul(vZ[i].Id,"Q-PRIM-LI-
M",P.Vnum(vZ[i].Id,"MULTIPLIER"));
      //for each month of year:
      for (int m=0;m<12;m++) {
        mQ PRIM LI M[m]+=mQ PRIM LI M Z[m];      //3.6 is to convert
KWh/m2 to MJ
        mQ_PRIM_LI+=mQ_PRIM_LI_M_Z[m];
      }
    }
  }

 //store lighting energy, monthly and annual values in kWh and MJ
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ_PRIM_LI_M[i] << ", ";
  res << mQ PRIM LI M[11] << " }";
  P.ModKey(Id,"Q-PRIM-LI-M",res.str());
 }

 {
  ostringstream res; res << mQ_PRIM_LI;
```

```
  P.ModKey(Id,"Q-PRIM-LI",res.str());
 }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ PRIM LI M[i]/3.6 << ", ";
  res << mQ_PRIM_LI_M[11]/3.6 << " }";
  P.ModKey(Id,"KWH-LI-M",res.str());
 }
 {
  ostringstream res; res << mQ PRIM LI/3.6;
  P.ModKey(Id,"KWH-LI",res.str());
 }
  //write KG CO2 for lighting
  if((P.Vcodid(cId,"BR-TYPE")==5)   ||   (P.Vcodid(cId,"EPC-TYPE")==6))
//Republic of Ireland only
     {
      ostringstream              res;              res              <<
mQ_PRIM_LI/3.6*T.Value("tKGCO2_IR",1,(int)D.Value("dELEC_CODID"));
      P.ModKey(Id,"KG-CO2-LI",res.str());
     }
  else if((P.Vcodid(cId,"BR-TYPE")==6))  //Jersey only
     {
      ostringstream              res;              res              <<
mQ PRIM LI/3.6*T.Value("tKGCO2 JER",1,(int)D.Value("dELEC CODID"));
      P.ModKey(Id,"KG-CO2-LI",res.str());
     }
  else  // all UK
     {
      ostringstream              res;              res              <<
mQ PRIM LI/3.6*T.Value("tKGCO2",1,(int)D.Value("dELEC CODID"));
      P.ModKey(Id,"KG-CO2-LI",res.str());
     }
}


void HVAC::fFUEL TYPE(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "  " << "HVAC::fFUEL_TYPE";
//primary energy needed for each fuel type for zones served by HVAC
system
//calculations now carried out at zone level
//this function now calls zone function and adds up all the figures
from zone level

//v2.0.a, calculation moved from HVAC to zone level:
//zone level function also calculates CO2 produced by zone
  for (int i=0;i<vZ.size();i++)
    vZ[i].fFUEL TYPE CO2(D, T, P);

 double mKWH_NATGAS_M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH NATGAS=0.0;
 double mKWH LPG M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH_LPG=0.0;
 double mKWH BIOGAS M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH BIOGAS=0.0;
 double mKWH_OIL_M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH OIL=0.0;
 double mKWH COAL M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH_COAL=0.0;
 double mKWH ANTHRACITE M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH ANTHRACITE=0.0;
 double mKWH_SMOKELESS_M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH SMOKELESS=0.0;
 double mKWH DUELFUEL M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH_DUELFUEL=0.0;
 double mKWH BIOMASS M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH BIOMASS=0.0;
 double mKWH_SUPELEC_M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
```

```
 double mKWH SUPELEC=0.0;
 double mKWH WASTEHEAT M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH WASTEHEAT=0.0;
 double mKWH DISTRICT HEATING M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH_DISTRICT_HEATING=0.0;

// New parameters to save the annual heating fuel ONLY (space heating
and DHW only).
 double mKWH NATGAS heat=0.0;
 double mKWH LPG heat=0.0;
 double mKWH_BIOGAS_heat=0.0;
 double mKWH OIL heat=0.0;
 double mKWH COAL heat=0.0;
 double mKWH_ANTHRACITE_heat=0.0;
 double mKWH SMOKELESS heat=0.0;
 double mKWH DUELFUEL heat=0.0;
 double mKWH_BIOMASS_heat=0.0;
 double mKWH SUPELEC heat=0.0;
 double mKWH WASTEHEAT heat=0.0;
 double mKWH_DISTRICT_HEATING_heat=0.0;
 double  mKWH HEATPUMP heat=0.0;            // annual heating (space
heating and DHW only) attributed to heat pumps

 deque<double> mKWH NATGAS M Z;
 deque<double> mKWH LPG M Z;
 deque<double> mKWH BIOGAS M Z;
 deque<double> mKWH OIL M Z;
 deque<double> mKWH COAL M Z;
 deque<double> mKWH ANTHRACITE M Z;
 deque<double> mKWH SMOKELESS M Z;
 deque<double> mKWH DUELFUEL M Z;
 deque<double> mKWH BIOMASS M Z;
 deque<double> mKWH SUPELEC M Z;
 deque<double> mKWH WASTEHEAT M Z;
 deque<double> mKWH DISTRICT HEATING M Z;

  double iMultiplier;
 //for each zone served by HVAC system:
  for (int i=0;i<vZ.size();i++) {
      iMultiplier=P.Vnum(vZ[i].Id,"MULTIPLIER");
    mKWH NATGAS M Z=P.Vlnum Mul(vZ[i].Id,"KWH-NATGAS-M",iMultiplier);
    mKWH_LPG_M_Z=P.Vlnum_Mul(vZ[i].Id,"KWH-LPG-M",iMultiplier);
    mKWH BIOGAS M Z=P.Vlnum Mul(vZ[i].Id,"KWH-BIOGAS-M",iMultiplier);
    mKWH OIL M Z=P.Vlnum Mul(vZ[i].Id,"KWH-OIL-M",iMultiplier);
    mKWH_COAL_M_Z=P.Vlnum_Mul(vZ[i].Id,"KWH-COAL-M",iMultiplier);
    mKWH ANTHRACITE M Z=P.Vlnum Mul(vZ[i].Id,"KWH-ANTHRACITE-
M",iMultiplier);
    mKWH_SMOKELESS_M_Z=P.Vlnum_Mul(vZ[i].Id,"KWH-SMOKELESS-
M",iMultiplier);
    mKWH DUELFUEL M Z=P.Vlnum Mul(vZ[i].Id,"KWH-DUELFUEL-
M",iMultiplier);
    mKWH BIOMASS M Z=P.Vlnum Mul(vZ[i].Id,"KWH-BIOMASS-
M",iMultiplier);
    mKWH_SUPELEC_M_Z=P.Vlnum_Mul(vZ[i].Id,"KWH-SUPELEC-
M",iMultiplier);
    mKWH WASTEHEAT M Z=P.Vlnum Mul(vZ[i].Id,"KWH-WASTEHEAT-
M",iMultiplier);
    mKWH DISTRICT HEATING M Z=P.Vlnum Mul(vZ[i].Id,"KWH-DISTRICT-
HEATING-M",iMultiplier);

// New parameters to save the annual heating fuel ONLY (space heating
and DHW only).
    mKWH_NATGAS_heat+=P.Vnum(vZ[i].Id,"KWH-NATGAS-HEAT")*iMultiplier;
    mKWH LPG heat+=P.Vnum(vZ[i].Id,"KWH-LPG-HEAT")*iMultiplier;
    mKWH BIOGAS heat+=P.Vnum(vZ[i].Id,"KWH-BIOGAS-HEAT")*iMultiplier;
    mKWH_OIL_heat+=P.Vnum(vZ[i].Id,"KWH-OIL-HEAT")*iMultiplier;
    mKWH COAL heat+=P.Vnum(vZ[i].Id,"KWH-COAL-HEAT")*iMultiplier;
    mKWH ANTHRACITE heat+=P.Vnum(vZ[i].Id,"KWH-ANTHRACITE-
HEAT")*iMultiplier;
```

```
    mKWH SMOKELESS heat+=P.Vnum(vZ[i].Id,"KWH-SMOKELESS-
HEAT")*iMultiplier;
    mKWH DUELFUEL heat+=P.Vnum(vZ[i].Id,"KWH-DUELFUEL-
HEAT")*iMultiplier;
    mKWH_BIOMASS_heat+=P.Vnum(vZ[i].Id,"KWH-BIOMASS-
HEAT")*iMultiplier;
    mKWH SUPELEC heat+=P.Vnum(vZ[i].Id,"KWH-SUPELEC-
HEAT")*iMultiplier;
    mKWH WASTEHEAT heat+=P.Vnum(vZ[i].Id,"KWH-WASTEHEAT-
HEAT")*iMultiplier;
    mKWH_DISTRICT_HEATING_heat+=P.Vnum(vZ[i].Id,"KWH-DISTRICT-
HEATING-HEAT")*iMultiplier;
    mKWH HEATPUMP heat+=P.Vnum(vZ[i].Id,"KWH-HEATPUMP-
HEAT")*iMultiplier;

    //for each month of year:
    for (int m=0;m<12;m++) {
      mKWH NATGAS M[m]+=mKWH NATGAS M Z[m];
      mKWH LPG M[m]+=mKWH LPG M Z[m];
      mKWH_BIOGAS_M[m]+=mKWH_BIOGAS_M_Z[m];
      mKWH OIL M[m]+=mKWH OIL M Z[m];
      mKWH COAL M[m]+=mKWH COAL M Z[m];
      mKWH ANTHRACITE M[m]+=mKWH ANTHRACITE M Z[m];
      mKWH SMOKELESS M[m]+=mKWH SMOKELESS M Z[m];
      mKWH DUELFUEL M[m]+=mKWH DUELFUEL M Z[m];
      mKWH BIOMASS M[m]+=mKWH BIOMASS M Z[m];
      mKWH SUPELEC M[m]+=mKWH SUPELEC M Z[m];
      mKWH WASTEHEAT M[m]+=mKWH WASTEHEAT M Z[m];
      mKWH DISTRICT HEATING M[m]+=mKWH DISTRICT HEATING M Z[m];
      mKWH NATGAS+=mKWH NATGAS M Z[m];
      mKWH LPG+=mKWH LPG M Z[m];
      mKWH BIOGAS+=mKWH BIOGAS M Z[m];
      mKWH OIL+=mKWH OIL M Z[m];
      mKWH COAL+=mKWH COAL M Z[m];
      mKWH ANTHRACITE+=mKWH ANTHRACITE M Z[m];
      mKWH SMOKELESS+=mKWH SMOKELESS M Z[m];
      mKWH DUELFUEL+=mKWH DUELFUEL M Z[m];
      mKWH BIOMASS+=mKWH BIOMASS M Z[m];
      mKWH_SUPELEC+=mKWH_SUPELEC_M_Z[m];
      mKWH WASTEHEAT+=mKWH WASTEHEAT M Z[m];
      mKWH DISTRICT HEATING+=mKWH DISTRICT HEATING M Z[m];
    }
   }

 //store energy used by fuel type in both kWh and MJ, and both annual
and monthly:
 {
  ostringstream res; res << mKWH_NATGAS_heat;
  P.ModKey(Id,"KWH-NATGAS-HEAT",res.str());  // for heating (and DHW)
only
 }
 {
  ostringstream res; res << mKWH NATGAS;
  P.ModKey(Id,"KWH-NATGAS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH NATGAS M[i] << ", ";
  res << mKWH_NATGAS_M[11] << " }";
  P.ModKey(Id,"KWH-NATGAS-M",res.str());
 }
 {
  ostringstream res; res << mKWH NATGAS*3.6;
  P.ModKey(Id,"Q-NATGAS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
```

```
    res << mKWH_NATGAS_M[i]*3.6 << ", ";
  res << mKWH_NATGAS_M[11]*3.6 << " }";
  P.ModKey(Id,"Q-NATGAS-M",res.str());
 }
 {
  ostringstream res; res << mKWH_LPG_heat;  // for heating (and DHW)
only
  P.ModKey(Id,"KWH-LPG-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH_LPG;
  P.ModKey(Id,"KWH-LPG",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_LPG_M[i] << ", ";
  res << mKWH_LPG_M[11] << " }";
  P.ModKey(Id,"KWH-LPG-M",res.str());
 }
 {
  ostringstream res; res << mKWH_LPG*3.6;
  P.ModKey(Id,"Q-LPG",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_LPG_M[i]*3.6 << ", ";
  res << mKWH_LPG_M[11]*3.6 << " }";
  P.ModKey(Id,"Q-LPG-M",res.str());
 }
 {
  ostringstream res; res << mKWH_BIOGAS_heat;  // for heating (and
DHW) only
  P.ModKey(Id,"KWH-BIOGAS-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH_BIOGAS;
  P.ModKey(Id,"KWH-BIOGAS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_BIOGAS_M[i] << ", ";
  res << mKWH_BIOGAS_M[11] << " }";
  P.ModKey(Id,"KWH-BIOGAS-M",res.str());
 }
 {
  ostringstream res; res << mKWH_BIOGAS*3.6;
  P.ModKey(Id,"Q-BIOGAS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_BIOGAS_M[i]*3.6 << ", ";
  res << mKWH_BIOGAS_M[11]*3.6 << " }";
  P.ModKey(Id,"Q-BIOGAS-M",res.str());
 }
 {
  ostringstream res; res << mKWH_OIL_heat;  // for heating (and DHW)
only
  P.ModKey(Id,"KWH-OIL-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH_OIL;
  P.ModKey(Id,"KWH-OIL",res.str());
 }
 {
  ostringstream res; res << "{ ";
```

```
  for (int i=0;i<11;i++)
    res << mKWH OIL M[i] << ", ";
  res << mKWH OIL M[11] << " }";
  P.ModKey(Id,"KWH-OIL-M",res.str());
 }
 {
  ostringstream res; res << mKWH OIL*3.6;
  P.ModKey(Id,"Q-OIL",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH OIL M[i]*3.6 << ", ";
  res << mKWH_OIL_M[11]*3.6 << " }";
  P.ModKey(Id,"Q-OIL-M",res.str());
 }
 {
  ostringstream res; res << mKWH COAL heat;     // for heating (and
DHW) only
  P.ModKey(Id,"KWH-COAL-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH COAL;
  P.ModKey(Id,"KWH-COAL",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH COAL M[i] << ", ";
  res << mKWH COAL M[11] << " }";
  P.ModKey(Id,"KWH-COAL-M",res.str());
 }
 {
  ostringstream res; res << mKWH COAL*3.6;
  P.ModKey(Id,"Q-COAL",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH COAL M[i]*3.6 << ", ";
  res << mKWH COAL M[11]*3.6 << " }";
  P.ModKey(Id,"Q-COAL-M",res.str());
 }
 {
  ostringstream res; res << mKWH ANTHRACITE heat;     // for heating
(and DHW) only
  P.ModKey(Id,"KWH-ANTHRACITE-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH ANTHRACITE;
  P.ModKey(Id,"KWH-ANTHRACITE",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH ANTHRACITE M[i] << ", ";
  res << mKWH_ANTHRACITE_M[11] << " }";
  P.ModKey(Id,"KWH-ANTHRACITE-M",res.str());
 }
 {
  ostringstream res; res << mKWH ANTHRACITE*3.6;
  P.ModKey(Id,"Q-ANTHRACITE",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH ANTHRACITE M[i]*3.6 << ", ";
  res << mKWH ANTHRACITE M[11]*3.6 << " }";
  P.ModKey(Id,"Q-ANTHRACITE-M",res.str());
```

```
 }
 {
  ostringstream res; res << mKWH SMOKELESS heat;       // for heating
(and DHW) only
  P.ModKey(Id,"KWH-SMOKELESS-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH_SMOKELESS;
  P.ModKey(Id,"KWH-SMOKELESS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_SMOKELESS_M[i] << ", ";
  res << mKWH SMOKELESS M[11] << " }";
  P.ModKey(Id,"KWH-SMOKELESS-M",res.str());
 }
 {
  ostringstream res; res << mKWH SMOKELESS*3.6;
  P.ModKey(Id,"Q-SMOKELESS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH SMOKELESS M[i]*3.6 << ", ";
  res << mKWH SMOKELESS M[11]*3.6 << " }";
  P.ModKey(Id,"Q-SMOKELESS-M",res.str());
 }
 {
  ostringstream res; res << mKWH DUELFUEL heat;        // for heating
(and DHW) only
  P.ModKey(Id,"KWH-DUELFUEL-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH DUELFUEL;
  P.ModKey(Id,"KWH-DUELFUEL",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH DUELFUEL M[i] << ", ";
  res << mKWH_DUELFUEL_M[11] << " }";
  P.ModKey(Id,"KWH-DUELFUEL-M",res.str());
 }
 {
  ostringstream res; res << mKWH DUELFUEL*3.6;
  P.ModKey(Id,"Q-DUELFUEL",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH DUELFUEL M[i]*3.6 << ", ";
  res << mKWH DUELFUEL M[11]*3.6 << " }";
  P.ModKey(Id,"Q-DUELFUEL-M",res.str());
 }
 {
  ostringstream res; res << mKWH_BIOMASS heat;         // for heating
(and DHW) only
  P.ModKey(Id,"KWH-BIOMASS-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH BIOMASS;
  P.ModKey(Id,"KWH-BIOMASS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH BIOMASS M[i] << ", ";
  res << mKWH_BIOMASS_M[11] << " }";
```

```
  P.ModKey(Id,"KWH-BIOMASS-M",res.str());
 }
 {
  ostringstream res; res << mKWH BIOMASS*3.6;
  P.ModKey(Id,"Q-BIOMASS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH BIOMASS M[i]*3.6 << ", ";
  res << mKWH_BIOMASS_M[11]*3.6 << " }";
  P.ModKey(Id,"Q-BIOMASS-M",res.str());
 }
 {
  ostringstream res; res << mKWH SUPELEC heat;         // for heating
(and DHW) only
  P.ModKey(Id,"KWH-SUPELEC-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH_SUPELEC;
  P.ModKey(Id,"KWH-SUPELEC",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH SUPELEC M[i] << ", ";
  res << mKWH SUPELEC M[11] << " }";
  P.ModKey(Id,"KWH-SUPELEC-M",res.str());
 }
 {
  ostringstream res; res << mKWH SUPELEC*3.6;
  P.ModKey(Id,"Q-SUPELEC",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH SUPELEC M[i]*3.6 << ", ";
  res << mKWH SUPELEC M[11]*3.6 << " }";
  P.ModKey(Id,"Q-SUPELEC-M",res.str());
 }
 {
  ostringstream res; res << mKWH WASTEHEAT heat;       // for heating
(and DHW) only
  P.ModKey(Id,"KWH-WASTEHEAT-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH WASTEHEAT;
  P.ModKey(Id,"KWH-WASTEHEAT",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH WASTEHEAT M[i] << ", ";
  res << mKWH_WASTEHEAT_M[11] << " }";
  P.ModKey(Id,"KWH-WASTEHEAT-M",res.str());
 }
 {
  ostringstream res; res << mKWH WASTEHEAT*3.6;
  P.ModKey(Id,"Q-WASTEHEAT",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH WASTEHEAT M[i]*3.6 << ", ";
  res << mKWH WASTEHEAT M[11]*3.6 << " }";
  P.ModKey(Id,"Q-WASTEHEAT-M",res.str());
 }
 {
  ostringstream res; res << mKWH DISTRICT HEATING heat;       // for
```

```
heating (and DHW) only
  P.ModKey(Id,"KWH-DISTRICT-HEATING-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH_DISTRICT_HEATING;
  P.ModKey(Id,"KWH-DISTRICT-HEATING",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_DISTRICT_HEATING_M[i] << ", ";
  res << mKWH DISTRICT HEATING M[11] << " }";
  P.ModKey(Id,"KWH-DISTRICT-HEATING-M",res.str());
 }
 {
  ostringstream res; res << mKWH DISTRICT HEATING*3.6;
  P.ModKey(Id,"Q-DISTRICT-HEATING",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH DISTRICT HEATING M[i]*3.6 << ", ";
  res << mKWH DISTRICT HEATING M[11]*3.6 << " }";
  P.ModKey(Id,"Q-DISTRICT-HEATING-M",res.str());
 }

 {
  ostringstream res; res << mKWH HEATPUMP heat;       // for heating
(and DHW) only
  P.ModKey(Id,"KWH-HEATPUMP-HEAT",res.str());
 }


}




void HVAC::Recommendation(PROJECTDefaults& D, PROJECTTables& T, BOC&
P, DATA REC& TNR) {
  if(func report) cout << endl << "  " << "HVAC::Recommendation";
/*
  EPC recommendations function for HVAC system

  This function:
     - assesses overall energy and CO2 performance of HVAC-SYSTEM for
both heating and cooling
     - assesses the impact of recommendations for heating and cooling

    Recommendations for HVAC-SYSTEM are:

      1) heating

         EPC-H1 - install high efficiency boiler
           -to assess impact, Calculate modified energy/ CO2 as ratio
between actual efficiency and 0.81.
         EPC-H3 - install condensing boiler
           -to assess impact, calculate modified energy/ CO2 as ratio
between actual efficiency and 0.89.
         EPC-R1   consider GSHP
           -to  assess  impact,  calculate  modified  energy  as  ratio
between actual efficiency and 3.1.
         EPC-R5   consider ASHP
           -to  assess  impact,  calculate  modified  energy  as  ratio
between actual efficiency and 2.2.
         EPC-H4   default heat generator efficiency
           -(similar to that for EPC-H1) to assess impact, Calculate
modified energy/ CO2 as ratio between actual efficiency and 0.81.
         EPC-H2   heating central time control
```

```
          -Improve heating efficiency by 1 %
        EPC-H5   local time control
          -Improve heating efficiency by 1 %
        EPC-H6   room temperature control
          -Improve heating efficiency by 2 %
        EPC-H7   Heating optimum start and stop control
          -Improve heating efficiency by 2 %
        EPC-H8   heating weather compensation controls
          -Improve heating efficiency by 1.5 %

      2) cooling

        EPC-C2   install high efficency chiller
          -to assess impact, calculate modified energy as ratio
between actual efficiency and 2.5
        EPC-C3   Inspect and seal ductwork
          -to assess impact, calculate modified energy by:
              If duct and AHU leakage is between 5% and 10% (fair):
              -  Reduce cooling energy by P% where P is: VAV,
constant volume, multizone, terminal reheat, dual duct P=5%, Fan
coil, induction P = 2%, Chilled ceiling, chilled beam P= 0.5%
              If duct and AHU leakage is less than 10% (poor):
              -  Reduce cooling energy by P% where P is: VAV,
constant volume, multizone, terminal reheat, dual duct P=10%, Fan
coil, induction P = 4%, Chilled ceiling, chilled beam P= 1%
        EPC-C1   default chiller efficiency
          -(similar to that for EPC-C2)to assess impact, calculate
modified energy as ratio between actual efficiency and 2.5
      3) fuel switching

        EPC-F1   Oil or LPG to natural gas (heating)
          -Calculate modified price weighted energy by weighting
with price weighting actor for original and new fuel
          -Calculate modified CO2 by changing fuel type
        EPC-F2   Coal to natural gas (heating)
          -Calculate modified price weighted energy by weighting
with price weighting actor for original and new fuel
          -Calculate modified CO2 by changing fuel type
        EPC-F3   Coal to biomass (heating)
          -Calculate modified price weighted energy by weighting
with price weighting actor for original and new fuel
          -Calculate modified CO2 by changing fuel type
        EPC-F4   Oil or LPG to biomass (heating)
          -Calculate modified price weighted energy by weighting
with price weighting actor for original and new fuel
          -Calculate modified CO2 by changing fuel type
        EPC-F5   gas to biomass (heating)
          -Calculate modified price weighted energy by weighting
with price weighting actor for original and new fuel
          -Calculate modified CO2 by changing fuel type
        EPC-F6   Coal to oil (heating)
          -Calculate modified price weighted energy by weighting
with price weighting actor for original and new fuel
          -Calculate modified CO2 by changing fuel type

*/


  // First Child Recommendation
  for (int i=0;i<vZ.size();i++) {
    vZ[i].Recommendation(D,T,P);  //currently  there  are  no  zone
recommendations
  }

  // Then Hvac Recommendations

//1) heating

  //depending on HVAC heating efficiency set HVAC keyword REC-CODE-
```

```
HEAT to GOOD, FAIR or POOR

  //first check if system provides heating
  if (T.Value("tSYS HEAT",1,P.Vcodid(Id,"TYPE"))==1)  {
    //basic parameters
    double mHEAT GEN SEFF=P.Vnum(Id,"HEAT-GEN-SEFF");

    //price weighting value from table tFuel Price Weighting (depends
on relative prices of fuels)
    //used in calculation of energy impact
    double
Fuel Price Weighting=T.Value("tFuel Price Weighting",1,P.Vcodid(Id,"F
UEL-TYPE"));

    //carbon intensity of fuel
    double mFUEL CARBON INTENSITY=0.0;
    if((P.Vcodid(cId,"BR-TYPE")==5) || (P.Vcodid(cId,"EPC-TYPE")==6))
//Republic of Ireland only

mFUEL_CARBON_INTENSITY=T.Value("tKGCO2_IR",1,P.Vcodid(Id,"FUEL-
TYPE"));
    else // all UK
       mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,P.Vcodid(Id,"FUEL-
TYPE"));
    deque<int> mDHW SERVED REF=P.Vlrefid(Id,"DHW-SERVED-REF");
    //heating energy and CO2 for HVAC heat generator:
    double mQ HEAT=P.Vnum(Id,"Q-PRIM-HEAT");
    double                    mQ HEAT PRICE WEIGHTED=P.Vnum(Id,"Q-PRIM-
HEAT")*Fuel Price Weighting;
    double                                mKG CO2 HEAT=(P.Vnum(Id,"KWH-
HEAT")*mFUEL CARBON INTENSITY);

    //variable for adjusted Total Building energy
    double mQ TOTAL ADJUSTED=0.0;
    double mCO2 TOTAL ADJUSTED=0.0;

    double mAREA=P.Vnum(gId,"AREA");

    double mQ_DHW=0.0;
    double mQ DHW PRICE WEIGHTED=0.0;
    double mKG CO2 DHW=0.0;
    string mREC_CODE_HEAT="Not applicable";

    //keywords for recommendations EPCR1 (GSHP) and EPCR5 (ASHP)
    //1=recommendation   is   triggered,   0=recommendation   is   not
triggered
    bool EPCR1=0;      //install ground source heat pump (GSHP)
    bool EPCR5=0;      //install air source heat pump (ASHP)

    //fuel switching recommendations:
    //1=recommendation   is   triggered,   0=recommendation   is   not
triggered
    bool EPCF2=0;      //Coal to natural gas
    bool EPCF3=0;      //Coal to biomass
    bool EPCF6=0;      //Coal to oil
    bool EPCF1=0;      //Oil or LPG to natural gas
    bool EPCF4=0;      //Oil or LPG to biomass
    bool EPCF5=0;      //gas to biomass

    //heat generator efficiency recommendations:
    //1=recommendation   is   triggered,   0=recommendation   is   not
triggered
    bool EPCH3=0;      //condensing boiler
    bool EPCH1=0;      //high efficiency boiler
    bool EPCH4=0;      //improve efficiency above default

    //REC-CODE can be either Not applicable, GOOD, FAIR or POOR
    mREC CODE HEAT="POOR";
    if (mHEAT_GEN_SEFF>D.Value("REC_DEF3")) {
```

```
      //if heat generator efficiency > x, classify heat generator
efficiency as "good":
      mREC CODE HEAT="GOOD";
    }
    else { // i.e., efficiency will be either FAIR or POOR
          // If fuel is Gas, LPG, or Oil, recommend condensing boiler
          if   ((P.Vcodid(Id,"FUEL-TYPE")==D.Value("dGAS CODID"))   ||
(P.Vcodid(Id,"FUEL-TYPE")==D.Value("dLPG CODID"))                 ||
(P.Vcodid(Id,"FUEL-TYPE")==D.Value("dOIL CODID")))
             //trigger recommendation EPCH3: condensing boiler
             EPCH3=1;
          if (mHEAT GEN SEFF>D.Value("REC DEF4"))
             //If x >= heat generator efficiency > y, classify heat
generator efficiency as "fair":
             mREC CODE HEAT="FAIR";
          else                        //i.e., heat generator efficiency
=< y
          //otherwise heat generator efficiency is "poor" (this is
set as the initial value above)
              //trigger  recommendation  EPCH1:  higher  efficiency
boiler
              EPCH1=1;
    }

    //write keyword: REC-PERFORMANCE-HEAT - heat generator efficiency
is GOOD/ FAIR/ POOR
    {
    ostringstream res; res << mREC CODE HEAT;
    P.ModKey(Id,"REC-PERFORMANCE-HEAT",res.str());
    }

    //if  using  default  heat  generator  efficiency,  trigger  EPCH4:
improve on default efficiency
    if (P.Vcodid(Id,"EFF-DEFAULT-HEAT")==1)
       EPCH4=1;


    //recommendations for this HVAC heat generator:

    //Trigger GSHP and ASHP recommendation
    //if fuel is electricity, check efficiency
    //if  less  than  D.Value("REC DEF7"),  trigger  GSHP  and  ASHP
recommendation
    if (P.Vcodid(Id,"FUEL-TYPE")==(int)D.Value("dELEC CODID")) {
      if (mHEAT_GEN_SEFF<D.Value("REC_DEF7")) {
        EPCR1=1; //install ground source heat pump (GSHP)
        EPCR5=1; //install air source heat pump (ASHP)
      }
    }

    //heating controls recommendations
    bool EPCH2=0;     //heating central time control
    bool EPCH5=0;     //local time control
    bool EPCH6=0;     //room temperature control
    bool EPCH7=0;     //Heating optimum start and stop control
    bool EPCH8=0;     //heating weather compensation controls

    //no central time control EPC-H2 (and no local time control or
optimum start/stop either)
    if (P.Vcodid(Id,"CEN-TIME-CTRL")==2) {
       if (P.Vcodid(Id,"ROOM-TIME-CTRL")==2) {
          if (P.Vcodid(Id,"OPTIMUM-START")==2) EPCH2=1;
       }
    }
    //no room by room time control
    if (P.Vcodid(Id,"ROOM-TIME-CTRL")==2) EPCH5=1;

    //no room by room temp control
    if (P.Vcodid(Id,"ROOM-TEMP-CTRL")==2) EPCH6=1;
```

```
    //no optimum start
    if (P.Vcodid(Id,"OPTIMUM-START")==2) EPCH7=1;

    //no weather compensation
    if (P.Vcodid(Id,"WEATHER-COMP")==2) EPCH8=1;

    //primary energy used for DHW:
    for (int j=0;j< mDHW SERVED REF.size();j++) {
      mQ DHW+=P.Vnum(mDHW SERVED REF[j],"Q-PRIM");
      mQ_DHW_PRICE_WEIGHTED+=P.Vnum(mDHW_SERVED_REF[j],"Q-
PRIM")*Fuel Price Weighting;
      mKG CO2 DHW+=P.Vnum(mDHW SERVED REF[j],"KWH-
DHW")*mFUEL_CARBON_INTENSITY;
    }

    //total building energy for all end uses (to be used when
assessing impact of recommendation):
    //this is price weighted using factor from table
tFuel_Price_Weighting (depends on relative prices of fuels)
    //used in calculation of energy impact for recommendations
    double mQ TOTAL PRICE WEIGHTED=P.Vnum(gId,"Q-TOTAL-P-WEIGHTED");

    //total building CO2 for all end uses (to be used when assessing
impact of recommendation)
    //and for calculating the % CO2 for this end use:
    //this keyword is the bulding CO2 if there were no renewables:
    double mKG CO2 PROJECT=P.Vnum(gId,"KG-CO2-EXCL-RENEWABLES");

    //fuel payback adjustment, weight with energy - used to calculate
carbon saved/pound spent
    double  Fuel Payback=T.Value("tFuel Payback",1,P.Vcodid(Id,"FUEL-
TYPE"));

    //price weighting factor from table tFuel Price Weighting
(depends on relative prices of fuels)
    //used in calculation of energy impact
    double
Fuel_Price_Weighting_heat=T.Value("tFuel_Price_Weighting",1,P.Vcodid(
Id,"FUEL-TYPE"));
    //price weighting factor for other fuels
    //these are used when assessing price weighted Energy impact of
fuel switching recommendations
    double
Fuel_Price_Weighting_NatGas=T.Value("tFuel_Price_Weighting",1,(int)D.
Value("dGAS CODID"));
    double
Fuel_Price_Weighting_biomass=T.Value("tFuel_Price_Weighting",1,(int)D
.Value("dBIOMASS CODID"));
    double
Fuel_Price_Weighting_oil=T.Value("tFuel_Price_Weighting",1,(int)D.Val
ue("dOIL CODID"));

    //variable to store energy for HVAC heat after recommendation is
applied
    double mHVAC HEAT ENERGY WEIGHTED=0.0;
    //variable to store CO2 for HVAC after recommendation is applied
    double mHVAC HEAT CO2 WEIGHTED=0.0;

    //stores impact of recommendation on CO2 (1=HIGH/2=MEDIUM/3=LOW)
    int IMPACT CO2=1;
    //stores impact of recommendation on energy
(1=HIGH/2=MEDIUM/3=LOW)
    int IMPACT ENERGY=1;

    //payback period
    double PAYBACK=0;
    //carbon saving per pound code  (GOOD, FAIR, POOR)
    int CARBON_SAVING_POUND=1;
```

```
    //carbon saving per pound numerical value, used to determine
carbon saving per pound code
    double CARBON SAVING NO;


    //Trigger fuel switching recommendations for heating
    switch(P.Vcodid(Id,"FUEL-TYPE")) {
      case 5: {
        //coal
        EPCF2=1; //Coal to natural gas
        EPCF3=1; //Coal to biomass
        EPCF6=1; //Coal to oil
      }
      break;
      case 4: {
        //oil
        EPCF1=1; //Oil or LPG to natural gas
        EPCF4=1; //Oil or LPG to biomass
      }
      break;
      case 2: {
        //LPG
        EPCF1=1; //Oil or LPG to natural gas
        EPCF4=1; //Oil or LPG to biomass
      }
      break;
      case 1: {
        //gas
        EPCF5=1; //gas to biomass
      }
      break;
    }


    //HVAC heating energy performance - GOOD/ FAIR/ POOR/

   //---- recommendation EPC-H3 ----

    //condensing boiler

    if (EPCH3) {
      //assess impact of recommendation EPCH3 on CO2 and Energy
      //if DHW energy is provided by HVAC generator, include this
energy in HVAC assessment (if DHW is not provided by HVAC, mQ DHW
will be zero)
        //correct energy according to recommendation (set efficiency
to D.Value("REC DEF20"))
        if ((D.Value("REC DEF20"))!=0) {                 // although
REC DEF20 should not be zero
          //weight energy => energy*(actual eff/D.Value("REC DEF20"))

mHVAC HEAT ENERGY WEIGHTED=(mQ HEAT+mQ DHW)*(mHEAT GEN SEFF/D.Value("
REC DEF20"));
          //weight CO2 => CO2*(actual eff/D.Value("REC_DEF20"))

mHVAC HEAT CO2 WEIGHTED=(mKG CO2 HEAT+mKG CO2 DHW)*(mHEAT GEN SEFF/D.
Value("REC_DEF20"));
        }

        //variable for adjusted Total Building energy
        mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
(mQ HEAT PRICE WEIGHTED+mQ DHW PRICE WEIGHTED)+(mHVAC HEAT ENERGY WEI
GHTED*Fuel_Price_Weighting_heat);
        mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
(mKG CO2 HEAT+mKG CO2 DHW)+mHVAC HEAT CO2 WEIGHTED;

        //assess energy and CO2 impact
        IMPACT ENERGY=fIMPACT ENERGY(D,       mQ TOTAL PRICE WEIGHTED,
mQ_TOTAL_ADJUSTED);
```

```
        IMPACT CO2=fIMPACT CO2(D,                        mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

        //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON_SAVING_POUND)
        //Multiply default payback by 140 and divide by TYPICAL
building heating consumption (kWh/m2.year)
        if((mAREA!=0)&&(TNR.typKWH_HEAT!=0))
          PAYBACK=D.Value("dEPC H3")*140.0/(TNR.typKWH HEAT/mAREA);
        else
          PAYBACK=0.0;

        CARBON SAVING NO=PAYBACK*Fuel Payback;

        //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)
        CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

        //write the EPCH3 recommendation keyword
        string EPC XX="EPC-H3";
        Write Rec Keyword(D, T, P, EPC XX, IMPACT ENERGY, IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
    }
   //---- END recommendation EPC-H3 ----


   //---- recommendation EPC-H1 ----

    //high efficiency boiler

    if (EPCH1) {
      //assess impact of recommendation EPC-H1 on CO2 and Energy
      //if DHW energy is provided by HVAC generator, include this
energy in HVAC assessment

      //correct energy according to recommendation (set efficiency to
D.Value("REC DEF15"))
      if ((D.Value("REC DEF15"))!=0) {
        //weight energy => energy*(actual eff/D.Value("REC DEF15"))

mHVAC HEAT ENERGY WEIGHTED=(mQ HEAT+mQ DHW)*(mHEAT GEN SEFF/D.Value("
REC DEF15"));
        //weight CO2 => CO2*(actual eff/D.Value("REC_DEF15"))

mHVAC HEAT CO2 WEIGHTED=(mKG CO2 HEAT+mKG CO2 DHW)*(mHEAT GEN SEFF/D.
Value("REC_DEF15"));
      }

      //variable for adjusted Total Building energy
      mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
(mQ HEAT PRICE WEIGHTED+mQ DHW PRICE WEIGHTED)+(mHVAC HEAT ENERGY WEI
GHTED*Fuel_Price_Weighting_heat);
      mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
(mKG CO2 HEAT+mKG CO2 DHW)+mHVAC HEAT CO2 WEIGHTED;

      //assess energy and CO2 impact
      IMPACT ENERGY=fIMPACT ENERGY(D,         mQ TOTAL PRICE WEIGHTED,
mQ_TOTAL_ADJUSTED);
      IMPACT CO2=fIMPACT CO2(D,                        mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

      //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
      //Multiply default payback by 140 and divide by TYPICAL
building heating consumption (kWh/m2.year)
      if((mAREA!=0)&&(TNR.typKWH HEAT!=0))
        PAYBACK=D.Value("dEPC_H1")*140.0/(TNR.typKWH_HEAT/mAREA);
      else
        PAYBACK=0.0;
```

```
      CARBON SAVING NO=PAYBACK*Fuel Payback;

      //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)
      CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

      //write the EPCH1 recommendation keyword
      string EPC XX="EPC-H1";
      Write Rec Keyword(D, T, P, EPC XX, IMPACT ENERGY, IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
    }
   //---- END recommendation EPC-H1 ----


   //---- recommendation EPC-R1 ----

    //GROUND SOURCE HEAT PUMP

    if (EPCR1) {
      //assess impact of recommendation EPC-R1 on CO2 and Energy
      //if DHW energy is provided by HVAC generator, include this
energy in HVAC assessment

      //correct energy according to recommendation (set efficiency to
D.Value("REC DEF16"))
      if ((D.Value("REC DEF16"))!=0) {
        //weight energy => energy*(actual eff/D.Value("REC DEF16"))

mHVAC HEAT ENERGY WEIGHTED=(mQ HEAT+mQ DHW)*(mHEAT GEN SEFF/D.Value("
REC DEF16"));
        //weight CO2 => CO2*(actual eff/D.Value("REC DEF16"))

mHVAC HEAT CO2 WEIGHTED=(mKG CO2 HEAT+mKG CO2 DHW)*(mHEAT GEN SEFF/D.
Value("REC DEF16"));
      }

      //variable for adjusted Total Building energy
      mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
(mQ HEAT PRICE WEIGHTED+mQ DHW PRICE WEIGHTED)+(mHVAC HEAT ENERGY WEI
GHTED*Fuel_Price_Weighting_heat);
      mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
(mKG CO2 HEAT+mKG CO2 DHW)+mHVAC HEAT CO2 WEIGHTED;

      //assess energy and CO2 impact
      IMPACT ENERGY=fIMPACT ENERGY(D,         mQ TOTAL PRICE WEIGHTED,
mQ_TOTAL_ADJUSTED);
      IMPACT CO2=fIMPACT CO2(D,                       mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

      //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
      //Multiply default payback by 140 and divide by TYPICAL
building heating consumption (kWh/m2.year)
      if((mAREA!=0)&&(TNR.typKWH HEAT!=0))
        PAYBACK=D.Value("dEPC_R1")*140.0/(TNR.typKWH_HEAT/mAREA);
      else
        PAYBACK=0.0;

      CARBON SAVING NO=PAYBACK*Fuel Payback;

      //determine CARBON_SAVING_POUND value (GOOD, FAIR, POOR)
      CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

      //write the EPCR1 recommendation keyword
      string EPC XX="EPC-R1";
      Write Rec Keyword(D, T, P, EPC XX, IMPACT ENERGY, IMPACT CO2,
PAYBACK, CARBON_SAVING_POUND);
    }
   //---- END recommendation EPC-R1 ----
```

```
   //---- recommendation EPC-R5 ----

    //AIR SOURCE HEAT PUMP

    if (EPCR5) {
      //assess impact of recommendation EPCR5 on CO2 and Energy
      //if DHW energy is provided by HVAC generator, include this
energy in HVAC assessment

      //correct energy according to recommendation (set efficiency to
D.Value("REC DEF17"))
      if ((D.Value("REC DEF17"))!=0) {
        //weight energy => energy*(actual eff/D.Value("REC_DEF17"))

mHVAC HEAT ENERGY WEIGHTED=(mQ HEAT+mQ DHW)*(mHEAT GEN SEFF/D.Value("
REC_DEF17"));
        //weight CO2 => CO2*(actual eff/D.Value("REC DEF17"))

mHVAC_HEAT_CO2_WEIGHTED=(mKG_CO2_HEAT+mKG_CO2_DHW)*(mHEAT_GEN_SEFF/D.
Value("REC DEF17"));
      }

      //variable for adjusted Total Building energy
      mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
(mQ HEAT PRICE WEIGHTED+mQ DHW PRICE WEIGHTED)+(mHVAC HEAT ENERGY WEI
GHTED*Fuel Price Weighting heat);
      mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
(mKG CO2 HEAT+mKG CO2 DHW)+mHVAC HEAT CO2 WEIGHTED;

      //assess energy and CO2 impact
      IMPACT ENERGY=fIMPACT ENERGY(D,         mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
      IMPACT CO2=fIMPACT CO2(D,                       mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

      //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
      //Multiply default payback by 140 and divide by TYPICAL
building heating consumption (kWh/m2.year)
      if((mAREA!=0)&&(TNR.typKWH HEAT!=0))
        PAYBACK=D.Value("dEPC_R5")*140.0/(TNR.typKWH_HEAT/mAREA);
      else
        PAYBACK=0.0;

      CARBON SAVING NO=PAYBACK*Fuel Payback;

      //determine CARBON_SAVING_POUND value (GOOD, FAIR, POOR)
      CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

      //write the EPCR5 recommendation keyword
      string EPC XX="EPC-R5";
      Write Rec Keyword(D, T, P, EPC XX, IMPACT ENERGY, IMPACT CO2,
PAYBACK, CARBON_SAVING_POUND);
    }
   //---- END recommendation EPC-R5 ----


   //---- recommendation EPC-H4 ----

    //default heat generator efficency
    if (EPCH4) {
      //assess impact of recommendation EPC-H4 on CO2 and Energy -
This uses the same value REC DEF15 used for EPC-H1
      //if DHW energy is provided by HVAC generator, include this
energy in HVAC assessment

      //correct energy according to recommendation (set efficiency to
D.Value("REC_DEF15"))
```

```
      if ((D.Value("REC DEF15"))!=0) {
        //weight energy => energy*(actual eff/D.Value("REC DEF15"))

mHVAC HEAT ENERGY WEIGHTED=(mQ HEAT+mQ DHW)*(mHEAT GEN SEFF/D.Value("
REC_DEF15"));
        //weight CO2 => CO2*(actual eff/D.Value("REC DEF15"))

mHVAC_HEAT_CO2_WEIGHTED=(mKG_CO2_HEAT+mKG_CO2_DHW)*(mHEAT_GEN_SEFF/D.
Value("REC DEF15"));
      }

      //variable for adjusted Total Building energy
      mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
(mQ_HEAT_PRICE_WEIGHTED+mQ_DHW_PRICE_WEIGHTED)+(mHVAC_HEAT_ENERGY_WEI
GHTED*Fuel Price Weighting heat);
      mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
(mKG_CO2_HEAT+mKG_CO2_DHW)+mHVAC_HEAT_CO2_WEIGHTED;

      //assess energy and CO2 impact
      IMPACT ENERGY=fIMPACT ENERGY(D,         mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
      IMPACT CO2=fIMPACT CO2(D,                       mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

      //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
      //Multiply default payback by 140 and divide by TYPICAL
building heating consumption (kWh/m2.year)
      if((mAREA!=0)&&(TNR.typKWH HEAT!=0))
        PAYBACK=D.Value("dEPC H4")*140.0/(TNR.typKWH HEAT/mAREA);
      else
        PAYBACK=0.0;

      CARBON SAVING NO=PAYBACK*Fuel Payback;

      //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)
      CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

      //write the EPCH4 recommendation keyword
      string EPC XX="EPC-H4";
      Write Rec Keyword(D, T, P, EPC XX, IMPACT ENERGY, IMPACT CO2,
PAYBACK, CARBON_SAVING_POUND);
    }
   //---- END recommendation EPC-H4 ----


   //---- recommendation EPC-H2 ----

    //centralised time control
    if (EPCH2) {
      //assess impact of recommendation EPC-H2 on CO2 and Energy
      //if DHW energy is provided by HVAC generator, include this
energy in HVAC assessment

      //correct energy according to recommendation (set efficiency to
(D.Value("REC DEF21")*mHEAT GEN SEFF))
      if ((D.Value("REC_DEF21"))!=0) {
        //weight                          energy                        =>
energy*(mHEAT GEN SEFF/(D.Value("REC DEF21")*mHEAT GEN SEFF))

mHVAC HEAT ENERGY WEIGHTED=(mQ HEAT+mQ DHW)/D.Value("REC DEF21");
        //weight                           CO2                          =>
CO2*(mHEAT_GEN_SEFF/(D.Value("REC_DEF21")*mHEAT_GEN_SEFF))

mHVAC HEAT CO2 WEIGHTED=(mKG CO2 HEAT+mKG CO2 DHW)/D.Value("REC DEF21
");
      }

      //variable for adjusted Total Building energy
```

```
      mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
(mQ HEAT PRICE WEIGHTED+mQ DHW PRICE WEIGHTED)+(mHVAC HEAT ENERGY WEI
GHTED*Fuel Price Weighting heat);
      mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
(mKG_CO2_HEAT+mKG_CO2_DHW)+mHVAC_HEAT_CO2_WEIGHTED;

      //assess energy and CO2 impact
      IMPACT ENERGY=fIMPACT ENERGY(D,         mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
      IMPACT CO2=fIMPACT CO2(D,                       mKG CO2 PROJECT,
mCO2_TOTAL_ADJUSTED);

      //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON_SAVING_POUND)
      //Multiply default payback by 140 and divide by TYPICAL
building heating consumption (kWh/m2.year)
      if((mAREA!=0)&&(TNR.typKWH_HEAT!=0))
        PAYBACK=D.Value("dEPC H2")*140.0/(TNR.typKWH HEAT/mAREA);
      else
        PAYBACK=0.0;

      CARBON SAVING NO=PAYBACK*Fuel Payback;

      //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)
      CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

      //write the EPCH2 recommendation keyword
      string EPC XX="EPC-H2";
      Write Rec Keyword(D,  T,  P,  EPC XX,  IMPACT ENERGY,  IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
    }
   //---- END recommendation EPC-H2 ----


   //---- recommendation EPC-H5 ----

    //room by room time control

    if (EPCH5) {
      //assess impact of recommendation EPC-H5 on CO2 and Energy
      //if  DHW  energy  is  provided  by  HVAC  generator,  include  this
energy in HVAC assessment

      //correct energy according to recommendation (set efficiency to
(D.Value("REC_DEF21")*mHEAT_GEN_SEFF))
      if ((D.Value("REC DEF21"))!=0) {
        //weight                        energy                          =>
energy*(mHEAT_GEN_SEFF/(D.Value("REC_DEF21")*mHEAT_GEN_SEFF))

mHVAC HEAT ENERGY WEIGHTED=(mQ HEAT+mQ DHW)/D.Value("REC DEF21");
        //weight                         CO2                            =>
CO2*(mHEAT GEN SEFF/(D.Value("REC DEF21")*mHEAT GEN SEFF))

mHVAC_HEAT_CO2_WEIGHTED=(mKG_CO2_HEAT+mKG_CO2_DHW)/D.Value("REC_DEF21
");
      }

      //variable for adjusted Total Building energy
      mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
(mQ_HEAT_PRICE_WEIGHTED+mQ_DHW_PRICE_WEIGHTED)+(mHVAC_HEAT_ENERGY_WEI
GHTED*Fuel Price Weighting heat);
      mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
(mKG_CO2_HEAT+mKG_CO2_DHW)+mHVAC_HEAT_CO2_WEIGHTED;

      //assess energy and CO2 impact
      IMPACT ENERGY=fIMPACT ENERGY(D,         mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
      IMPACT CO2=fIMPACT CO2(D,                       mKG CO2 PROJECT,
mCO2_TOTAL_ADJUSTED);
```

```
      //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
      //Multiply default payback by 140 and divide by TYPICAL
building heating consumption (kWh/m2.year)
      if((mAREA!=0)&&(TNR.typKWH HEAT!=0))
        PAYBACK=D.Value("dEPC H5")*140.0/(TNR.typKWH HEAT/mAREA);
      else
        PAYBACK=0.0;

      CARBON_SAVING_NO=PAYBACK*Fuel_Payback;

      //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)
      CARBON_SAVING_POUND=fCARBON_SAVING_POUND(D,CARBON_SAVING_NO);

      //write the EPCH5 recommendation keyword
      string EPC_XX="EPC-H5";
      Write Rec Keyword(D, T, P, EPC XX, IMPACT ENERGY, IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
    }
   //---- END recommendation EPC-H5 ----


   //---- recommendation EPC-H6 ----

    //room by room temperature control
    if (EPCH6) {
      //assess impact of recommendation EPCH6 on CO2 and Energy
      //if DHW energy is provided by HVAC generator, include this
energy in HVAC assessment

      //correct energy according to recommendation (set efficiency to
(D.Value("REC DEF22")*mHEAT GEN SEFF))
      if ((D.Value("REC DEF22"))!=0) {
        //weight                          energy                          =>
energy*(mHEAT GEN SEFF/(D.Value("REC DEF22")*mHEAT GEN SEFF))

mHVAC HEAT ENERGY WEIGHTED=(mQ HEAT+mQ DHW)/D.Value("REC DEF22");
        //weight                           CO2                            =>
CO2*(mHEAT GEN SEFF/(D.Value("REC DEF22")*mHEAT GEN SEFF))

mHVAC_HEAT_CO2_WEIGHTED=(mKG_CO2_HEAT+mKG_CO2_DHW)/D.Value("REC_DEF22
");
      }

      //variable for adjusted Total Building energy
      mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
(mQ_HEAT_PRICE_WEIGHTED+mQ_DHW_PRICE_WEIGHTED)+(mHVAC_HEAT_ENERGY_WEI
GHTED*Fuel Price Weighting heat);
      mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
(mKG_CO2_HEAT+mKG_CO2_DHW)+mHVAC_HEAT_CO2_WEIGHTED;

      //assess energy and CO2 impact
      IMPACT ENERGY=fIMPACT ENERGY(D,         mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
      IMPACT CO2=fIMPACT CO2(D,                       mKG CO2 PROJECT,
mCO2_TOTAL_ADJUSTED);

      //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON_SAVING_POUND)
      //Multiply default payback by 140 and divide by TYPICAL
building heating consumption (kWh/m2.year)
      if((mAREA!=0)&&(TNR.typKWH_HEAT!=0))
        PAYBACK=D.Value("dEPC H6")*140.0/(TNR.typKWH HEAT/mAREA);
      else
        PAYBACK=0.0;

      CARBON SAVING NO=PAYBACK*Fuel Payback;
```

```
      //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)
      CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

      //write the EPCH6 recommendation keyword
      string EPC_XX="EPC-H6";
      Write Rec Keyword(D, T, P, EPC XX, IMPACT ENERGY, IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
    }
   //---- END recommendation EPC-H6 ----


   //---- recommendation EPC-H7 ----

    //optimum start and stop control

    if (EPCH7) {
      //assess impact of recommendation EPCH7 on CO2 and Energy
      //if DHW energy is provided by HVAC generator, include this
energy in HVAC assessment

      //correct energy according to recommendation (set efficiency to
(D.Value("REC DEF22")*mHEAT GEN SEFF))
      if ((D.Value("REC DEF22"))!=0) {
        //weight                          energy                          =>
energy*(mHEAT GEN SEFF/(D.Value("REC DEF22")*mHEAT GEN SEFF))

mHVAC HEAT ENERGY WEIGHTED=(mQ HEAT+mQ DHW)/D.Value("REC DEF22");
        //weight                           CO2                            =>
CO2*(mHEAT GEN SEFF/(D.Value("REC DEF22")*mHEAT GEN SEFF))

mHVAC HEAT CO2 WEIGHTED=(mKG CO2 HEAT+mKG CO2 DHW)/D.Value("REC DEF22
");
      }

      //variable for adjusted Total Building energy
      mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
(mQ HEAT PRICE WEIGHTED+mQ DHW PRICE WEIGHTED)+(mHVAC HEAT ENERGY WEI
GHTED*Fuel Price Weighting heat);
      mCO2_TOTAL_ADJUSTED=mKG_CO2_PROJECT-
(mKG CO2 HEAT+mKG CO2 DHW)+mHVAC HEAT CO2 WEIGHTED;

      //assess energy and CO2 impact
      IMPACT ENERGY=fIMPACT ENERGY(D,         mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
      IMPACT CO2=fIMPACT CO2(D,                       mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

      //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
      //Multiply default payback by 140 and divide by TYPICAL
building heating consumption (kWh/m2.year)
      if((mAREA!=0)&&(TNR.typKWH HEAT!=0))
        PAYBACK=D.Value("dEPC H7")*140.0/(TNR.typKWH HEAT/mAREA);
      else
        PAYBACK=0.0;

      CARBON_SAVING_NO=PAYBACK*Fuel_Payback;

      //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)
      CARBON_SAVING_POUND=fCARBON_SAVING_POUND(D,CARBON_SAVING_NO);

      //write the EPCH7 recommendation keyword
      string EPC_XX="EPC-H7";
      Write Rec Keyword(D, T, P, EPC XX, IMPACT ENERGY, IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
    }
   //---- END recommendation EPC-H7 ----
```

```
   //---- recommendation EPC-H8 ----

    //weather compensation controls

    if (EPCH8) {
      //assess impact of recommendation EPCH8 on CO2 and Energy
      //if DHW energy is provided by HVAC generator, include this
energy in HVAC assessment

      //correct energy according to recommendation (set efficiency to
(D.Value("REC_DEF23")*mHEAT_GEN_SEFF))
      if ((D.Value("REC DEF23"))!=0) {
        //weight                          energy                          =>
energy*(mHEAT_GEN_SEFF/(D.Value("REC_DEF23")*mHEAT_GEN_SEFF))

mHVAC HEAT ENERGY WEIGHTED=(mQ HEAT+mQ DHW)/D.Value("REC DEF23");
        //weight                           CO2                            =>
CO2*(mHEAT GEN SEFF/(D.Value("REC DEF23")*mHEAT GEN SEFF))

mHVAC_HEAT_CO2_WEIGHTED=(mKG_CO2_HEAT+mKG_CO2_DHW)/D.Value("REC_DEF23
");
      }

      //variable for adjusted Total Building energy
      mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
(mQ HEAT PRICE WEIGHTED+mQ DHW PRICE WEIGHTED)+(mHVAC HEAT ENERGY WEI
GHTED*Fuel Price Weighting heat);
      mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
(mKG CO2 HEAT+mKG CO2 DHW)+mHVAC HEAT CO2 WEIGHTED;

      //assess energy and CO2 impact
      IMPACT ENERGY=fIMPACT ENERGY(D,         mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
      IMPACT CO2=fIMPACT CO2(D,                       mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

      //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
      //Multiply default payback by 140 and divide by TYPICAL
building heating consumption (kWh/m2.year)
      if((mAREA!=0)&&(TNR.typKWH HEAT!=0))
        PAYBACK=D.Value("dEPC_H8")*140.0/(TNR.typKWH_HEAT/mAREA);
      else
        PAYBACK=0.0;

      CARBON SAVING NO=PAYBACK*Fuel Payback;

      //determine CARBON_SAVING_POUND value (GOOD, FAIR, POOR)
      CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

      //write the EPCH8 recommendation keyword
      string EPC XX="EPC-H8";
      Write Rec Keyword(D, T, P, EPC XX, IMPACT ENERGY, IMPACT CO2,
PAYBACK, CARBON_SAVING_POUND);
    }
   //---- END recommendation EPC-H8 ----


/*    Fuel switching recommendations for heating
        coal:
             EPCF2 - Coal to natural gas
             EPCF3 - Coal to biomass
             EPCF6 - Coal to oil
        oil:
            EPCF1 - Oil or LPG to natural gas
            EPCF4 - Oil or LPG to biomass
        LPG:
            EPCF1 - Oil or LPG to natural gas
            EPCF4 - Oil or LPG to biomass
```

```
        gas:
            EPCF5 - gas to biomass
*/


   //---- recommendation EPC-F1 ----

    //Oil or LPG to natural gas

    if (EPCF1) {
      //recommendation has been triggered if HVAC heating fuel is oil
or LPG
      //assess impact of recommendation EPCF1 on CO2 and Energy
      //if DHW energy is provided by HVAC generator, include this
energy in HVAC assessment

      //variable for adjusted Total Building price weighted energy
      //the price weighting factor will be different for oil (or LPG)
and natural gas
      //adjusted figure uses price weighting factor for natural gas
      mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
(mQ HEAT PRICE WEIGHTED+mQ DHW PRICE WEIGHTED)+((mQ HEAT+mQ DHW)*Fuel
 Price Weighting NatGas);

      //assess change in CO2 output
      //weight  CO2  =>  CO2*(fuel  intensity  for  natural  gas/fuel
intensity for current fuel)
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mHVAC HEAT CO2 WEIGHTED=(mKG CO2 HEAT+mKG CO2 DHW)*(T.Value("tKGCO2 I
R",1,(int)D.Value("dGAS CODID"))/mFUEL CARBON INTENSITY);
      else // all UK

mHVAC HEAT CO2 WEIGHTED=(mKG CO2 HEAT+mKG CO2 DHW)*(T.Value("tKGCO2",
1,(int)D.Value("dGAS CODID"))/mFUEL CARBON INTENSITY);

      //variable for adjusted Total Building CO2
      mCO2_TOTAL_ADJUSTED=mKG_CO2_PROJECT-
(mKG CO2 HEAT+mKG CO2 DHW)+mHVAC HEAT CO2 WEIGHTED;

      //assess energy and CO2 impact
      IMPACT ENERGY=fIMPACT ENERGY(D,         mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
      IMPACT CO2=fIMPACT CO2(D,                       mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

      //calculate  payback  (PAYBACK)  and  Carbon  saving  per  pound
(CARBON SAVING POUND)
      //Multiply  default  payback  by  140  and  divide  by  TYPICAL
building heating consumption (kWh/m2.year)
      if((mAREA!=0)&&(TNR.typKWH HEAT!=0))
        PAYBACK=D.Value("dEPC F1")*140.0/(TNR.typKWH HEAT/mAREA);
      else
        PAYBACK=0.0;

      CARBON_SAVING_NO=PAYBACK*Fuel_Payback;

      //for  fuel  switching  recommendations  only,  when  assessing  the
CARBON SAVING POUND value, adjust for the carbon content of different
fuels by
      //multiplying  CARBON SAVING POUND  value  calculated  above  by
relevant value from table tRel_Carbon:
      //    the table value used depends on both the original and the
new fuel
      //     in table tRel Carbon: original fuel in columns, new fuel
in rows

CARBON_SAVING_NO*=T.Value("tRel_Carbon",(int)D.Value("dGAS_CODID"),P.
```

```
Vcodid(Id,"FUEL-TYPE"));

      //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)
      CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

      //write the EPCF1 recommendation keyword
      string EPC XX="EPC-F1";
      Write Rec Keyword(D, T, P, EPC XX, IMPACT ENERGY, IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
    }
   //---- END recommendation EPC-F1 ----


   //---- recommendation EPC-F2 ----

    //Coal to natural gas

    if (EPCF2) {
      //recommendation has been triggered if HVAC heating fuel is
coal
      //assess impact of recommendation EPCF2 on CO2 and Energy
      //if DHW energy is provided by HVAC generator, include this
energy in HVAC assessment

      //variable for adjusted Total Building price weighted energy
      //the price weighting factor will be different for coal and
natural gas
      //adjusted figure uses price weighting factor for natural gas
      mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
(mQ HEAT PRICE WEIGHTED+mQ DHW PRICE WEIGHTED)+((mQ HEAT+mQ DHW)*Fuel
 Price Weighting NatGas);

      //assess change in CO2 output
      //weight CO2 => CO2*(fuel intensity for natural gas/fuel
intensity for current fuel)
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mHVAC_HEAT_CO2_WEIGHTED=(mKG_CO2_HEAT+mKG_CO2_DHW)*(T.Value("tKGCO2_I
R",1,(int)D.Value("dGAS CODID"))/mFUEL CARBON INTENSITY);
      else // all UK

mHVAC HEAT CO2 WEIGHTED=(mKG CO2 HEAT+mKG CO2 DHW)*(T.Value("tKGCO2",
1,(int)D.Value("dGAS CODID"))/mFUEL CARBON INTENSITY);
      //variable for adjusted Total Building CO2
      mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
(mKG CO2 HEAT+mKG CO2 DHW)+mHVAC HEAT CO2 WEIGHTED;

      //assess energy and CO2 impact
      IMPACT ENERGY=fIMPACT ENERGY(D,         mQ TOTAL PRICE WEIGHTED,
mQ_TOTAL_ADJUSTED);
      IMPACT CO2=fIMPACT CO2(D,                       mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

      //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
      //Multiply default payback by 140 and divide by TYPICAL
building heating consumption (kWh/m2.year)
      if((mAREA!=0)&&(TNR.typKWH HEAT!=0))
        PAYBACK=D.Value("dEPC_F2")*140.0/(TNR.typKWH_HEAT/mAREA);
      else
        PAYBACK=0.0;

      CARBON SAVING NO=PAYBACK*Fuel Payback;

      //for fuel switching recommendations only, when assessing the
CARBON SAVING POUND value, adjust for the carbon content of different
fuels by
      //multiplying  CARBON SAVING POUND  value  calculated  above  by
```

```
relevant value from table tRel Carbon:
      //    the table value used depends on both the original and the
new fuel
      //    in table tRel Carbon: original fuel in columns, new fuel
in rows

CARBON SAVING NO*=T.Value("tRel Carbon",(int)D.Value("dGAS CODID"),P.
Vcodid(Id,"FUEL-TYPE"));

      //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)
      CARBON_SAVING_POUND=fCARBON_SAVING_POUND(D,CARBON_SAVING_NO);

      //write the EPCF2 recommendation keyword
      string EPC_XX="EPC-F2";
      Write Rec Keyword(D, T, P, EPC XX, IMPACT ENERGY, IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
    }
   //---- END recommendation EPC-F2 ----


   //---- recommendation EPC-F3 ----

    //Coal to biomass

    if (EPCF3) {
      //recommendation has been triggered if HVAC heating fuel is
coal
      //assess impact of recommendation EPCF3 on CO2 and Energy
      //if DHW energy is provided by HVAC generator, include this
energy in HVAC assessment

      //variable for adjusted Total Building price weighted energy
      //the price weighting factor will be different for coal and
biomass
      //adjusted figure uses price weighting factor for biomass
      mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
(mQ HEAT PRICE WEIGHTED+mQ DHW PRICE WEIGHTED)+((mQ HEAT+mQ DHW)*Fuel
 Price Weighting biomass);

      //assess change in CO2 output
      //weight CO2 => CO2*(fuel intensity for biomass/fuel intensity
for current fuel)
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mHVAC HEAT CO2 WEIGHTED=(mKG CO2 HEAT+mKG CO2 DHW)*(T.Value("tKGCO2 I
R",1,(int)D.Value("dBIOMASS CODID"))/mFUEL CARBON INTENSITY);
      else // all UK

mHVAC HEAT CO2 WEIGHTED=(mKG CO2 HEAT+mKG CO2 DHW)*(T.Value("tKGCO2",
1,(int)D.Value("dBIOMASS_CODID"))/mFUEL_CARBON_INTENSITY);
      //variable for adjusted Total Building CO2
      mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
(mKG_CO2_HEAT+mKG_CO2_DHW)+mHVAC_HEAT_CO2_WEIGHTED;

      //assess energy and CO2 impact
      IMPACT ENERGY=fIMPACT ENERGY(D,         mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
      IMPACT CO2=fIMPACT CO2(D,                       mKG CO2 PROJECT,
mCO2_TOTAL_ADJUSTED);

      //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON_SAVING_POUND)
      //Multiply default payback by 140 and divide by TYPICAL
building heating consumption (kWh/m2.year)
      if((mAREA!=0)&&(TNR.typKWH_HEAT!=0))
        PAYBACK=D.Value("dEPC F3")*140.0/(TNR.typKWH HEAT/mAREA);
      else
        PAYBACK=0.0;
```

```
      CARBON SAVING NO=PAYBACK*Fuel Payback;

      //for fuel switching recommendations only, when assessing the
CARBON SAVING POUND value, adjust for the carbon content of different
fuels by
      //multiplying  CARBON SAVING POUND  value  calculated  above  by
relevant value from table tRel_Carbon:
      //    the table value used depends on both the original and the
new fuel
      //     in table tRel Carbon: original fuel in columns, new fuel
in rows

CARBON_SAVING_NO*=T.Value("tRel_Carbon",(int)D.Value("dBIOMASS_CODID"
),P.Vcodid(Id,"FUEL-TYPE"));

      //determine CARBON_SAVING_POUND value (GOOD, FAIR, POOR)
      CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

      //write the EPCF3 recommendation keyword
      string EPC XX="EPC-F3";
      Write Rec Keyword(D,  T,  P,  EPC XX,  IMPACT ENERGY,  IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
    }
   //---- END recommendation EPC-F3 ----


   //---- recommendation EPC-F4 ----

    //Oil or LPG to biomass (heating)

    if (EPCF4) {
      //recommendation has been triggered if HVAC heating fuel is Oil
or LPG
      //assess impact of recommendation EPCF4 on CO2 and Energy
      //if  DHW  energy  is  provided  by  HVAC  generator,  include  this
energy in HVAC assessment

      //variable for adjusted Total Building price weighted energy
      //the price weighting factor will be different for Oil or LPG
and biomass
      //adjusted figure uses price weighting factor for biomass
      mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
(mQ HEAT PRICE WEIGHTED+mQ DHW PRICE WEIGHTED)+((mQ HEAT+mQ DHW)*Fuel
_Price_Weighting_biomass);

      //assess change in CO2 output
      //weight  CO2  =>  CO2*(fuel  intensity  for  biomass/fuel  intensity
for current fuel)
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mHVAC HEAT CO2 WEIGHTED=(mKG CO2 HEAT+mKG CO2 DHW)*(T.Value("tKGCO2 I
R",1,(int)D.Value("dBIOMASS_CODID"))/mFUEL_CARBON_INTENSITY);
      else // all UK

mHVAC_HEAT_CO2_WEIGHTED=(mKG_CO2_HEAT+mKG_CO2_DHW)*(T.Value("tKGCO2",
1,(int)D.Value("dBIOMASS CODID"))/mFUEL CARBON INTENSITY);
      //variable for adjusted Total Building CO2
      mCO2_TOTAL_ADJUSTED=mKG_CO2_PROJECT-
(mKG CO2 HEAT+mKG CO2 DHW)+mHVAC HEAT CO2 WEIGHTED;

      //assess energy and CO2 impact
      IMPACT ENERGY=fIMPACT ENERGY(D,         mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
      IMPACT CO2=fIMPACT CO2(D,                       mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

      //calculate  payback  (PAYBACK)  and  Carbon  saving  per  pound
```

```
(CARBON SAVING POUND)
      //Multiply default payback by 140 and divide by TYPICAL
building heating consumption (kWh/m2.year)
      if((mAREA!=0)&&(TNR.typKWH HEAT!=0))
        PAYBACK=D.Value("dEPC_F4")*140.0/(TNR.typKWH_HEAT/mAREA);
      else
        PAYBACK=0.0;

      CARBON SAVING NO=PAYBACK*Fuel Payback;

      //for fuel switching recommendations only, when assessing the
CARBON SAVING POUND value, adjust for the carbon content of different
fuels by
      //multiplying  CARBON SAVING POUND  value  calculated  above  by
relevant value from table tRel Carbon:
      //    the table value used depends on both the original and the
new fuel
      //     in table tRel Carbon: original fuel in columns, new fuel
in rows

CARBON SAVING NO*=T.Value("tRel Carbon",(int)D.Value("dBIOMASS CODID"
),P.Vcodid(Id,"FUEL-TYPE"));

      //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)
      CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

      //write the EPCF4 recommendation keyword
      string EPC XX="EPC-F4";
      Write Rec Keyword(D,  T,  P,  EPC XX,  IMPACT ENERGY,  IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
    }
   //---- END recommendation EPC-F4 ----


   //---- recommendation EPC-F5 ----

    //gas to biomass (heating)

    if (EPCF5) {
      //recommendation  has  been  triggered  if  HVAC  heating  fuel  is
natural gas
      //assess impact of recommendation EPCF5 on CO2 and Energy
      //if  DHW  energy  is  provided  by  HVAC  generator,  include  this
energy in HVAC assessment

      //variable for adjusted Total Building price weighted energy
      //the price weighting factor will be different for natural gas
and biomass
      //adjusted figure uses price weighting factor for biomass
      mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
(mQ_HEAT_PRICE_WEIGHTED+mQ_DHW_PRICE_WEIGHTED)+((mQ_HEAT+mQ_DHW)*Fuel
 Price Weighting biomass);

      //assess change in CO2 output
      //weight  CO2  =>  CO2*(fuel  intensity  for  biomass/fuel  intensity
for current fuel)
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mHVAC_HEAT_CO2_WEIGHTED=(mKG_CO2_HEAT+mKG_CO2_DHW)/mFUEL_CARBON_INTEN
SITY*T.Value("tKGCO2 IR",1,(int)D.Value("dBIOMASS CODID"));
      else // all UK

mHVAC HEAT CO2 WEIGHTED=(mKG CO2 HEAT+mKG CO2 DHW)/mFUEL CARBON INTEN
SITY*T.Value("tKGCO2",1,(int)D.Value("dBIOMASS CODID"));
      //variable for adjusted Total Building CO2
      mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
(mKG CO2 HEAT+mKG CO2 DHW)+mHVAC HEAT CO2 WEIGHTED;
```

```
      //assess energy and CO2 impact
      IMPACT ENERGY=fIMPACT ENERGY(D,         mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
      IMPACT CO2=fIMPACT CO2(D,                       mKG CO2 PROJECT,
mCO2_TOTAL_ADJUSTED);

      //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON_SAVING_POUND)
      //Multiply default payback by 140 and divide by TYPICAL
building heating consumption (kWh/m2.year)
      if((mAREA!=0)&&(TNR.typKWH_HEAT!=0))
        PAYBACK=D.Value("dEPC F5")*140.0/(TNR.typKWH HEAT/mAREA);
      else
        PAYBACK=0.0;

      CARBON SAVING NO=PAYBACK*Fuel Payback;

      //for fuel switching recommendations only, when assessing the
CARBON SAVING POUND value, adjust for the carbon content of different
fuels by
      //multiplying  CARBON SAVING POUND  value  calculated  above  by
relevant value from table tRel Carbon:
      //    the table value used depends on both the original and the
new fuel
      //     in table tRel Carbon: original fuel in columns, new fuel
in rows

CARBON SAVING NO*=T.Value("tRel Carbon",(int)D.Value("dBIOMASS CODID"
),P.Vcodid(Id,"FUEL-TYPE"));

      //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)
      CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

      //write the EPCF5 recommendation keyword
      string EPC XX="EPC-F5";
      Write Rec Keyword(D,  T,  P,  EPC XX,  IMPACT ENERGY,  IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
    }
   //---- END recommendation EPC-F5 ----


   //---- recommendation EPC-F6 ----

    //Coal to oil (heating)

    if (EPCF6) {
      //recommendation  has  been  triggered  if  HVAC  heating  fuel  is
Coal
      //assess impact of recommendation EPCF6 on CO2 and Energy
      //if  DHW  energy  is  provided  by  HVAC  generator,  include  this
energy in HVAC assessment

      //variable for adjusted Total Building price weighted energy
      //the price weighting factor will be different for Coal and oil
      //adjusted figure uses price weighting factor for oil
      mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
(mQ_HEAT_PRICE_WEIGHTED+mQ_DHW_PRICE_WEIGHTED)+(mQ_HEAT+mQ_DHW)*Fuel_
Price Weighting oil;

      //assess change in CO2 output
      //weight CO2 => CO2*(fuel intensity for oil/fuel intensity for
current fuel)
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mHVAC_HEAT_CO2_WEIGHTED=(mKG_CO2_HEAT+mKG_CO2_DHW)*(T.Value("tKGCO2_I
R",1,(int)D.Value("dOIL CODID"))/mFUEL CARBON INTENSITY);
      else // all UK
```

```
mHVAC HEAT CO2 WEIGHTED=(mKG CO2 HEAT+mKG CO2 DHW)*(T.Value("tKGCO2",
1,(int)D.Value("dOIL CODID"))/mFUEL CARBON INTENSITY);
      //variable for adjusted Total Building CO2
      mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
(mKG_CO2_HEAT+mKG_CO2_DHW)+mHVAC_HEAT_CO2_WEIGHTED;

      //assess energy and CO2 impact
      IMPACT_ENERGY=fIMPACT ENERGY(D,         mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
      IMPACT CO2=fIMPACT CO2(D,                       mKG CO2 PROJECT,
mCO2_TOTAL_ADJUSTED);

      //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON_SAVING_POUND)
      //Multiply default payback by 140 and divide by TYPICAL
building heating consumption (kWh/m2.year)
      if((mAREA!=0)&&(TNR.typKWH_HEAT!=0))
        PAYBACK=D.Value("dEPC F6")*140.0/(TNR.typKWH HEAT/mAREA);
      else
        PAYBACK=0.0;

      CARBON SAVING NO=PAYBACK*Fuel Payback;

      //for fuel switching recommendations only, when assessing the
CARBON SAVING POUND value, adjust for the carbon content of different
fuels by
      //multiplying  CARBON SAVING POUND  value  calculated  above  by
relevant value from table tRel Carbon:
      //    the table value used depends on both the original and the
new fuel
      //     in table tRel Carbon: original fuel in columns, new fuel
in rows

CARBON SAVING NO*=T.Value("tRel Carbon",(int)D.Value("dOIL CODID"),P.
Vcodid(Id,"FUEL-TYPE"));

      //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)
      CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

      //write the EPCF6 recommendation keyword
      string EPC XX="EPC-F6";
      Write_Rec_Keyword(D,  T,  P,  EPC_XX,  IMPACT ENERGY,  IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
    }
   //---- END recommendation EPC-F6 ----


  }   //end of... check if system provides heating


//2) cooling

  //here using keyword COOL-GEN-EER for cooling generator efficiency
  //first check if system provides cooling
  if (T.Value("tSYS COOL",1,P.Vcodid(Id,"TYPE"))==1) {
    string mREC CODE COOL="Not applicable";
    double mCOOL_GEN_EER=P.Vnum(Id,"COOL-GEN-EER");

    //keywords for recommendations
    //1->  recommendation  is  triggered,  0->recommendation  is  not
triggered
    bool EPCC2=0; // higher efficiency cold generator

    //default cold generator efficiency
    bool EPCC1=0;

    //REC-CODE can be either GOOD, FAIR or POOR
    mREC CODE COOL="POOR";
    //if  cool  generator  efficiency  >  x,  classify  cool  generator
```

```
efficiency as "good":
    // AN (03/01/2008): a new keyword REC DEF5a was created which is
slightly lower than REC DEF5
    // so that the recommendation of upgrading to REC DEF5 has an
impact on energy and CO2
    if (mCOOL GEN EER>D.Value("REC DEF5a"))
       mREC CODE COOL="GOOD";
    //If x >= cool generator efficiency > y, classify cool generator
efficiency as "fair":
    else {    // i.e., efficiency will be either FAIR or POOR
         EPCC2=1;        // Trigger recommendation EPCC2: upgrade to
higher efficiency chiller
         if (mCOOL GEN EER>D.Value("REC DEF6"))
            mREC_CODE_COOL="FAIR";
    }
    //otherwise cool generator efficiency is "poor" (this is set as
the initial value above)
   {
    ostringstream res; res << mREC CODE COOL;
    P.ModKey(Id,"REC-PERFORMANCE-COOL",res.str());
   }

    //if using default cool generator efficiency, trigger EPC-C1
    if (P.Vcodid(Id,"EFF-DEFAULT-COOL")==1)
       EPCC1=1;

    double mFUEL CARBON INTENSITY=0.0;
    double Fuel Payback=0.0;

    //stores impact of recommendation on CO2 (1=HIGH/2=MEDIUM/3=LOW)
    int IMPACT CO2=1;
    //stores     impact     of     recommendation     on     energy
(1=HIGH/2=MEDIUM/3=LOW)
    int IMPACT ENERGY=1;

    //payback period
    double PAYBACK=0;
    //carbon saving per pound
    int CARBON_SAVING_POUND=1;
    double CARBON SAVING NO;

    //total project area
    double mAREA=P.Vnum(gId,"AREA");
    double Fuel Price Weighting cool=1.0;

    //determine cooling fuel carbon intensity:
/*
    // If Heat Pump then Fuel Type for the Heat Pump defined in the
cooling Tab
    //  is  also  used  for  the  cooling  system.  Otherwise,  use
electricity as fuel type.
     //fuel  payback  adjustment,  weight  with  energy  -  used  to
calculate carbon saved/pound spent
    if    (P.Vcodid(Id,"HEAT-SOURCE")==12    ||    P.Vcodid(Id,"HEAT-
SOURCE")==13) {
       if((P.Vcodid(cId,"BR-TYPE")==5)     ||     (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,P.Vcodid(Id,"FUEL-
TYPE"));
       else // all UK

mFUEL_CARBON_INTENSITY=T.Value("tKGCO2",1,P.Vcodid(Id,"FUEL-TYPE"));
      Fuel Payback=T.Value("tFuel Payback",1,P.Vcodid(Id,"FUEL-
TYPE"));
      //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
      //used in calculation of energy impact
```

```
Fuel Price Weighting cool=T.Value("tFuel Price Weighting",1,P.Vcodid(
Id,"FUEL-TYPE"));
    }
    else  {
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL_CARBON_INTENSITY=T.Value("tKGCO2_IR",1,(int)D.Value("dELEC_CODI
D"));
      else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,(int)D.Value("dELEC CODID")
);

Fuel Payback=T.Value("tFuel Payback",1,(int)D.Value("dELEC CODID"));
      //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
      //used in calculation of energy impact

Fuel_Price_Weighting_cool=T.Value("tFuel_Price_Weighting",1,(int)D.Va
lue("dELEC CODID"));
    }
*/
    // !!! 15/12/2008: Cooling now has its own fuel type keyword
    if((P.Vcodid(cId,"BR-TYPE")==5) || (P.Vcodid(cId,"EPC-TYPE")==6))
//Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,(int)T.Value("tFUEL TYPE
 COOL",1,P.Vcodid(Id,"FUEL-TYPE-COOL")));
    else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,(int)T.Value("tFUEL TYPE CO
OL",1,P.Vcodid(Id,"FUEL-TYPE-COOL")));

Fuel Payback=T.Value("tFuel Payback",1,(int)T.Value("tFUEL TYPE COOL"
,1,P.Vcodid(Id,"FUEL-TYPE-COOL")));
    //price weighting value from table tFuel Price Weighting (depends
on relative prices of fuels)
    //used in calculation of energy impact

Fuel Price Weighting cool=T.Value("tFuel Price Weighting",1,(int)T.Va
lue("tFUEL_TYPE_COOL",1,P.Vcodid(Id,"FUEL-TYPE-COOL")));

    //cooling energy and CO2:
    double mQ_COOL=P.Vnum(Id,"Q-PRIM-COOL");
    double                    mQ COOL PRICE WEIGHTED=P.Vnum(Id,"Q-PRIM-
COOL")*Fuel Price Weighting cool;
    double                                mKG_CO2_COOL=(P.Vnum(Id,"KWH-
COOL")*mFUEL CARBON INTENSITY);
    //used to store adjusted energy and CO2 after recommendations are
applied:
    double mQ ADJUSTED=0.0;
    double mCO2 ADJUSTED=0.0;

    //total  building  energy  for  all  end  uses  (to  be  used  when
assessing impact of recommendation):
    //this   is   price   weighted   using   factor   from   table
tFuel Price Weighting (depends on relative prices of fuels)
    //used in calculation of energy impact for recommendations
    double mQ_TOTAL_PRICE_WEIGHTED=P.Vnum(gId,"Q-TOTAL-P-WEIGHTED");

    //total building CO2 for all end uses (to be used when assessing
impact of recommendation)
    //and for calculating the % CO2 for this end use:
    //this keyword is the bulding CO2 if there were no renewables:
    double mKG_CO2_PROJECT=P.Vnum(gId,"KG-CO2-EXCL-RENEWABLES");

    //variable to store energy for HVAC cooling after recommendation
is applied
```

```
    double mHVAC COOL ENERGY WEIGHTED=0.0;
    //variable to store CO2 for HVAC after recommendation is applied
    double mHVAC COOL CO2 WEIGHTED=0.0;


   //---- recommendation EPC-C2 ----

    //high efficiency chiller

    //if recommendation EPC-C2  was triggered
    if (EPCC2) {
      //assess impact of recommendation EPCC2 on CO2 and Energy

      //correct energy according to recommendation (set efficiency to
D.Value("REC DEF5"))
      if ((D.Value("REC DEF5"))!=0) {
        //weight energy => energy*(actual eff/D.Value("REC_DEF5"))

mHVAC COOL ENERGY WEIGHTED=mQ COOL*(mCOOL GEN EER/D.Value("REC DEF5")
);
        //weight CO2 => CO2*(actual eff/D.Value("REC DEF5"))

mHVAC COOL CO2 WEIGHTED=mKG CO2 COOL*(mCOOL GEN EER/D.Value("REC DEF5
"));
      }

      //used to store adjusted energy and CO2 after recommendations
are applied:
      mQ ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mQ COOL PRICE WEIGHTED+(mHVAC COOL ENERGY WEIGHTED*Fuel Price Weighti
ng cool);
      mCO2 ADJUSTED=mKG CO2 PROJECT-
mKG CO2 COOL+mHVAC COOL CO2 WEIGHTED;

      //assess energy and CO2 impact
      IMPACT ENERGY=fIMPACT ENERGY(D,         mQ TOTAL PRICE WEIGHTED,
mQ ADJUSTED);
      IMPACT CO2=fIMPACT CO2(D, mKG CO2 PROJECT, mCO2 ADJUSTED);

      //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
      //Multiply default payback by 30 and divide by 1.2*Notional
building cooling consumption (kWh/m2.year)
      if((mAREA!=0)&&(TNR.notKWH COOL!=0))

PAYBACK=D.Value("dEPC C2")*30.0/((TNR.notKWH COOL*1.2)/mAREA);
      else
        PAYBACK=0.0;

      CARBON SAVING NO=PAYBACK*Fuel Payback;

      //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)
      CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

      //write the EPCC2 recommendation keyword
      string EPC XX="EPC-C2";
      Write Rec Keyword(D,  T,  P,  EPC XX,  IMPACT ENERGY,  IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
    }
   //---- END recommendation EPC-C2 ----


   //---- recommendation EPC-C1 ----

    //default chiller efficency

    //if recommendation EPC-C1  was triggered
    if (EPCC1) {
      //assess impact of recommendation EPCC2 on CO2 and Energy
```

```
      //correct energy according to recommendation (set efficiency to
D.Value("REC DEF5"))
      if ((D.Value("REC DEF5"))!=0) {
        //weight energy => energy*(actual eff/D.Value("REC_DEF5"))

mHVAC COOL ENERGY WEIGHTED=mQ COOL*(mCOOL GEN EER/D.Value("REC DEF5")
);
        //weight CO2 => CO2*(actual eff/D.Value("REC DEF5"))

mHVAC_COOL_CO2_WEIGHTED=mKG_CO2_COOL*(mCOOL_GEN_EER/D.Value("REC_DEF5
"));
      }

      //used to store adjusted energy and CO2 after recommendations
are applied:
      mQ_ADJUSTED=mQ_TOTAL_PRICE_WEIGHTED-
mQ COOL PRICE WEIGHTED+(mHVAC COOL ENERGY WEIGHTED*Fuel Price Weighti
ng cool);
      mCO2_ADJUSTED=mKG_CO2_PROJECT-
mKG CO2 COOL+mHVAC COOL CO2 WEIGHTED;

      //assess energy and CO2 impact
      IMPACT ENERGY=fIMPACT ENERGY(D,         mQ TOTAL PRICE WEIGHTED,
mQ ADJUSTED);
      IMPACT CO2=fIMPACT CO2(D, mKG CO2 PROJECT, mCO2 ADJUSTED);

      //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
      //Multiply default payback by 30 and divide by 1.2*Notional
building cooling consumption (kWh/m2.year)
      if((mAREA!=0)&&(TNR.notKWH COOL!=0))

PAYBACK=D.Value("dEPC C1")*30.0/((TNR.notKWH COOL*1.2)/mAREA);
      else
        PAYBACK=0.0;

      CARBON SAVING NO=PAYBACK*Fuel Payback;

      //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)
      CARBON SAVING POUND=fCARBON SAVING POUND(D,CARBON SAVING NO);

      //write the EPCC2 recommendation keyword
      string EPC XX="EPC-C1";
      Write_Rec_Keyword(D,  T,  P,  EPC_XX,  IMPACT_ENERGY,  IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
    }
   //---- END recommendation EPC-C1 ----


   //---- recommendation EPC-C3 ----

    //If the HVAC system is one of:
    //VAV (including packaged cabinet), fan coil, induction, constant
volume, multizone,
    //terminal reheat, dual duct, chilled ceiling or chilled beam
(with displacement ventilation),
    //or active chilled beams, then check duct and AHU leakage

    int mTYPE=P.Vcodid(Id,"TYPE");
    if ((mTYPE>11)&&(mTYPE<24)) {
    //If duct leakage>=5%, trigger recommendation EPCC3
      double mDuct_Leakage=P.Vnum(Id,"DUCT-LEAKAGE-PC");
      if (mDuct Leakage>=5) {
        //assess impact on cooling
        //Reduce cooling energy by P% where P is taken from table
tDUCT LEAK CORR
        //use the right row to read the correction factor from the
table, this depends on the % leakage
```

```
        int row;
        if (mDuct Leakage<10) row=1;
        else row=2;

        double mP_Factor=T.Value("tDUCT_LEAK_CORR",row,mTYPE);

        //correct energy according to recommendation (reduce HVAC
cooling energy by P%)
        mHVAC COOL ENERGY WEIGHTED=mQ COOL*(100.0-mP Factor)/100.0;
        //weight CO2 (reduce HVAC cooling energy by P%)
        mHVAC_COOL_CO2_WEIGHTED=mKG_CO2_COOL*(100.0-mP_Factor)/100.0;

        //used to store adjusted energy and CO2 after recommendations
are applied:
        mQ ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mQ COOL PRICE WEIGHTED+(mHVAC COOL ENERGY WEIGHTED*Fuel Price Weighti
ng_cool);
        mCO2 ADJUSTED=mKG CO2 PROJECT-
mKG CO2 COOL+mHVAC COOL CO2 WEIGHTED;

        //assess energy and CO2 impact
        IMPACT ENERGY=fIMPACT ENERGY(D,       mQ TOTAL PRICE WEIGHTED,
mQ ADJUSTED);
        IMPACT CO2=fIMPACT CO2(D, mKG CO2 PROJECT, mCO2 ADJUSTED);

        //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
        //Multiply payback by 60 and divide by NOTIONAL building
auxiliary energy consumption (kWh/m2.year)
        if((mAREA!=0)&&(TNR.notKWH AUX!=0))
          PAYBACK=D.Value("dEPC C3")*60.0/((TNR.notKWH AUX)/mAREA);
        else
          PAYBACK=0.0;

        CARBON SAVING NO=PAYBACK*Fuel Payback;
        CARBON SAVING POUND=1;

        //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)
        CARBON_SAVING_POUND=fCARBON_SAVING_POUND(D,CARBON_SAVING_NO);

        //write the EPCC3 recommendation keyword
        string EPC_XX="EPC-C3";
        Write Rec Keyword(D, T, P, EPC XX, IMPACT ENERGY, IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
      }
    }
   //---- END recommendation EPC-C3 ----

  }    //end of... check if system provides cooling
}


void  HVAC::Write Rec Keyword(PROJECTDefaults&  D,  PROJECTTables&  T,
BOC&  P,  string  EPC XX,  int  IMPACT ENERGY,  int  IMPACT CO2,  double
PAYBACK, int CARBON SAVING POUND) {
  if(func report) cout << endl << "  " << "HVAC::Write Rec Keyword  "
<< EPC XX << "  " << IMPACT ENERGY << "  " << IMPACT CO2 << "  " <<
PAYBACK << "  " << CARBON SAVING POUND;

/*

  Function to write keywords for EPC recommendations
  String EPC_XX is name of recommendation keyword e.g. EPC-H1

*/

  //first see what the value of the keyword is. User may have set
some or all
  //of the values or set that this recommmendation should not be used
```

```
etc...
  deque<double> mEPC XX=P.Vlnum(Id,EPC XX);

  if (mEPC XX.size()==1) {
    //user has not modified this recommendation
    for (int i=0;i<4;i++) mEPC XX.push back(0.0);
  }
  if  (mEPC XX[0]!=1)  {         //  1  means  User  has  taken  the
recommendation out
    //user modified this recommendation
    //write the parameters which were not set by the user
    //ENERGY impact (1=HIGH/2=MEDIUM/3=LOW)
    if ((mEPC XX[1]==-1)||(mEPC XX[1]==0)) mEPC XX[1]=IMPACT ENERGY;
    //CO2 impact (1=HIGH/2=MEDIUM/3=LOW) always low
    if ((mEPC XX[2]==-1)||(mEPC XX[2]==0)) mEPC XX[2]=IMPACT CO2;
    //payback period (number)
    if ((mEPC_XX[3]==-1)||(mEPC_XX[3]==0)) mEPC_XX[3]=PAYBACK;
    //Carbon saving per pound (1=GOOD/2=FAIR/3=POOR)
    if                              ((mEPC XX[4]==-1)||(mEPC XX[4]==0))
mEPC_XX[4]=CARBON_SAVING_POUND;
    //Id is included (user has modified values)
    if (mEPC XX.size()==6) {
      int REC USER Id=(int)mEPC XX[5];
      {
       ostringstream res; res << "{ ";
       for (int i=0;i<5;i++)
         res << mEPC XX[i] << ", ";
       res << REC USER Id << " }";
       P.ModKey(Id,EPC XX,res.str());
      }
    }
    else {
      //calc  in  -  sbem  decided  that  the  recommendation  should  be
included and set the values
      mEPC XX[0]=4;
      {
       ostringstream res; res << "{ ";
       for (int i=0;i<4;i++)
         res << mEPC_XX[i] << ", ";
       res << mEPC XX[4] << " }";
       P.ModKey(Id,EPC XX,res.str());
      }
    }
  }
} //end of HVAC::Write_Rec_Keyword(...)
```

PVS

The PVS command defines the performance parameters of a photovoltaic (PV) energy system. A U-name is required.

Example input:

```
"Solar PV 1" = PVS
  AREA            = 3
  WATT-PEAK-TYPE  = CRISTAL
  SES-ORIENTATION = South
  SES-INCLINATION = 60
  ..
```

AREA

Specifies the area of the PV system, without the bearing construction, in m^2. This keyword is required.

WATT-PEAK-TYPE
The watt-peak capacity per m^2 for an opaque PV panel depends on its composition which can be one of the values listed below. The default is 'Other thin films'.

Monocrystalline silicon
Policrystalline silicon
Amorphous silicon
Other thin films

SES-ORIENTATION
Specifies the orientation of the PV panel, as one of the values listed below. The default is 'South'.

North
North-East
East
South-East
South
South-West
West
North-West

SES-INCLINATION
Specifies the inclination of the PV panel in degrees from the horizontal, as one of the values listed below. The default is 60.

90
75
60
45
30
15
0

MULTIPLIER
This keyword specifies how many PV systems with this exact description exist in the project. E.g., if the value of the multiplier is 3, there are 3 PV systems each with identical keyword values in the project. This enables repeating PV systems to be entered quickly and easily. The default value is 1.

c++ code

```
//: pvs.cpp

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void PVS::TEST() {
// This function is designed to test different types of objects
// Go on developing it
```

```
//  cout << "Inicio de Test PVS" << endl;
//  cout << "Fin de Test PVS" << endl;
  BEEMObject::TEST();
}

// v4.3.0 Addition of
// PVS::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P, BOC&N,
int i)

void PVS::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "  " << "PVS::Calc";
  if(calc report) cout << endl << P.vINP[Id].Name;
// its own calcs below...
  SES::fQ_RAD(D, T, P);
  fQ DISP(D, T, P);
}

void PVS::fQ DISP(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "PVS::fQ DISP";
//calculate energy displaced by PVS (MJ and kWh)
  double mCO2 factor;
  if((P.Vcodid(cId,"BR-TYPE")==5)  ||  (P.Vcodid(cId,"EPC-TYPE")==6))
mCO2 factor=T.Value("tKGCO2 IR",1,11);
  else if((P.Vcodid(cId,"BR-TYPE")==6))
     mCO2 factor=T.Value("tKGCO2 JER",1,11);
  else //all UK
     mCO2 factor=T.Value("tKGCO2",1,11);
  double mAREA=P.Vnum(Id,"AREA");
  deque<double>                    mQ DISP M=P.Vlnum Mul(Id,"Q-RAD-
M",T.Value("T102 PVS",1,P.Vcodid(Id,"WATT-PEAK-TYPE"))*(1-
D.Value("dPVS LOSS"))*mAREA);
  double mQ DISP=0; for (int m=0;m<12;m++) mQ DISP+=mQ DISP M[m];

  // Adjust for proportion applicable to core
  double mSC ACTUAL RATIO=P.Vnum(cId,"SC-ACTUAL-RATIO");
  for (int m=0;m<12;m++)
    mQ DISP M[m]*=mSC ACTUAL RATIO;

  mQ_DISP*=mSC_ACTUAL_RATIO;


 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ_DISP_M[i] << ", ";
  res << mQ DISP M[11] << " }";
  P.ModKey(Id,"Q-DISP-M",res.str());
 }
 {
  ostringstream res; res << mQ DISP;
  P.ModKey(Id,"Q-DISP",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ DISP M[i]/3.6 << ", ";
  res << mQ_DISP_M[11]/3.6 << " }";
  P.ModKey(Id,"KWH-DISP-M",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ_DISP_M[i]/3.6*mCO2_factor << ", ";
  res << mQ DISP M[11]/3.6*mCO2 factor << " }";
  P.ModKey(Id,"KG-CO2-DISP-M",res.str());
 }
 {
  ostringstream res; res << mQ DISP/3.6;
  P.ModKey(Id,"KWH-DISP",res.str());
```

```
  }
  {
   ostringstream res; res << mQ DISP/3.6*mCO2 factor;
   P.ModKey(Id,"KG-CO2-DISP",res.str());
  }
}

void PVS::Recommendation(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func report) cout << endl << "  " << "PVS::Recommendation";

}



// End Object: PVS
```

REC-USER

The REC-USER command defines the parameters related to the recommendations that are defined by the user, in relation to the Energy Performance Certificate, whether they are modifications of the NCM-generated recommendations or additional recommendations created by the user. A U-name is required.
Example input:

```
"R1" = REC-USER
   CATEGORY             = HEATING
   CODE                 = EPC-W3
   REC-TEXT             = "REC-TEXT"
   REC-COMMENT          = "NO COMMENT"
   ENERGY-IMPACT        = HIGH
   CO2-IMPACT           = HIGH
   PAY-BACK             = LONG
   CO2-SAVE-POUND       = GOOD
   REF-OBJECT-TYPE      = DHW-GENERATOR
   REF-OBJECT           = "Default HWS"
   ..
```

CATEGORY (*RENAMED FROM REC-CATEGORY*)
Specifies the category of the recommendation, as one of the values listed below. The default is 'OTHER'.

HEATING
COOLING
HOT-WATER
LIGHTING
RENEWABLES
OVERHEATING
ENVELOPE
FUEL-SWITCHING
AUXILIARY
OTHER

CODE
Specifies the code of the recommendation, as one of the values listed below. This is a required keyword.

Code	Recommendation Text
USER	*The recommendation is input by the user.*
EPC-C1	The default chiller efficiency is chosen. It is recommended that the chiller system be investigated to gain an understanding of its efficiency and possible improvements.
EPC-C2	Chiller efficiency is low. Consider upgrading chiller plant.
EPC-C3	Ductwork leakage is high. Inspect and seal ductwork
EPC-W1	Install more efficient water heater
EPC-W3	Improve insulation on HWS storage
EPC-W4	Add time clock to HWS secondary circulation
EPC-W2	Consider installing replacing HWS with point of use system.
EPC-E1	Some floors are poorly insulated – introduce/improve insulation. Add insulation to the exposed surfaces of floors adjacent to underground, unheated spaces, or exterior.
EPC-E2	Roof is poorly insulated. Install/improve insulation of roof.
EPC-E3	Some solid walls are poorly insulated - introduce/improve internal wall insulation.
EPC-E4	Some walls have uninsulated cavities - introduce cavity wall insulation.
EPC-E5	Some windows have high U-values - consider installing secondary glazing.
EPC-E6	Some loft spaces are poorly insulated - install/improve insulation.
EPC-E7	Carry out a pressure test, identify and treat identified air leakage. Enter result in EPC calculation.
EPC-E8	Some glazing is poorly insulated. Replace/improve glazing and/or frames.
EPC-F1	Consider switching from oil or LPG to natural gas.
EPC-F2	Consider converting the existing boiler from coal to natural gas.
EPC-F3	Consider switching from coal to biomass.
EPC-F4	Consider switching from oil or LPG to biomass.
EPC-F5	Consider switching from gas to biomass.
EPC-F6	Consider switching from coal to oil.
EPC-H2	Add time control to heating system.
EPC-H5	Add local time control to heating system.
EPC-H6	Add local temperature control to the heating system.
EPC-H7	Add optimum start/stop to the heating system.
EPC-H8	Add weather compensation controls to heating system.
EPC-H1	Consider replacing heating boiler plant with high efficiency type.
EPC-H3	Consider replacing heating boiler plant with a condensing type.
EPC-H4	The default heat generator efficiency is chosen. It is recommended that the heat generator system be investigated to gain an understanding of its efficiency and possible improvements.
EPC-L1	Replace 38mm diameter (T12) fluorescent tubes on failure with 26mm (T8) tubes.
EPC-L2	Replace tungsten GLS lamps with CFLs: Payback period dependent on hours of use.
EPC-L3	Replace high-pressure mercury discharge lamps with plug-in SON replacements.
EPC-L4	Replace tungsten GLS spotlights with low-voltage tungsten halogen: Payback period dependent on hours of use.

EPC-L5 Consider replacing T8 lamps with retrofit T5 conversion kit.
EPC-L6 Replace high-pressure mercury discharge lamps with complete new lamp/gear SON (DL).
EPC-L7 Introduce HF (high frequency) ballasts for fluorescent tubes: Reduced number of fittings required.
EPC-V1 In some spaces, the solar gain limit in criterion 3 of ADL2A 2010 is exceeded, which might cause overheating. Consider solar control measures such as the application of reflective coating or shading devices to windows.
EPC-R1 Consider installing a ground source heat pump.
EPC-R2 Consider installing building mounted wind turbine(s).
EPC-R3 Consider installing solar water heating.
EPC-R4 Consider installing PV.
EPC-R5 Consider installing an air source heat pump.

REC-TEXT
Specifies the text for a user-defined recommendation when the value of the keyword CODE is 'USER'.

USER-IN-OUT
Specifies how the user modified an NCM recommendation, as one of the values listed below. The default is 'IN'.

Value	Description
IN	Recommendation was included by the user (rather than the calculation)
OUT	Recommendation was excluded by the user (rather than the calculation)

REC-TEXT
Specifies the text for the recommendation.

REC-COMMENT
Specifies comments input by the user in relation to the recommendation.

ENERGY-IMPACT
Specifies the potential impact of the recommendation on the energy performance of the building, as one of the values listed below. The default is 'UNKNOWN'.

HIGH
MEDIUM
LOW
UNKNOWN

CO2-IMPACT
Specifies the potential impact of the recommendation on the CO_2 emissions of the building, as one of the values listed below. The default is 'UNKNOWN'.

HIGH
MEDIUM
LOW
UNKNOWN

PAY-BACK

Specifies the typical payback time for implementing the recommendation in the building, as one of the values listed below. The default is 'UNKNOWN'.

LONG
MEDIUM
SHORT
UNKNOWN

PAY-BACK-NUM
Specifies the numerical value of the typical payback time for implementing the recommendation in the building, in years.

PAY-BACK-SC
Specifies the source of the typical payback time for implementing the recommendation in the building, as one of the values listed below. The default is 'CALC'.

Value	Description
USER	Payback time was input by the user
CALC	Payback time was determined by the calculation

CO2-SAVE-POUND
Specifies the level of the potential CO_2 emissions saved from the building per pound spent on implementing the recommendation, as one of the values listed below. The default is 'UNKNOWN'.

GOOD
FAIR
POOR
UNKNOWN

REF-OBJECT-TYPE
Specifies the type of object to which the recommendation refers, as one of the values listed below. The default is 'WHOLE-BUILDING'.

WHOLE-BUILDING
HVAC-SYSTEM
DHW-GENERATOR

REF-OBJECT
Specifies the U-name previously defined for the object to which the recommendation refers if the REC-USER object's REF-OBJECT-TYPE keyword is set to 'DHW-GENERATOR' or 'HVAC-SYSTEM'.

REF-OBJECT-ISBEM-ID
This keyword specifies the interface's ID number for the object referred to by the keyword REF-OBJECT, and it is used when calculated values relating to that object need to be reported within the interface.

c++ code
```
//: recommendation.cpp
```

```
#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void REC USER::TEST() {
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test REC" << endl;
//  cout << "Fin de Test REC" << endl;
  BEEMObject::TEST();
}

void REC USER::Check(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "REC USER::Check";


}

void REC USER::CheckNCM(PROJECTDefaults& D, PROJECTTables& T, BOC& P,
int rpId) {
  if(func report) cout << endl << "   " << "REC USER::CheckNCM   " <<
rpId;
// Add logic of REC-USER objects modifying NCM recommendations
  int oId;

  if (P.Vcod(Id,"CODE")!="USER") {
    if (P.Vcod(Id,"REF-OBJECT-TYPE")=="WHOLE-BUILDING")
      oId=rpId;
    else
      oId=P.Vrefid(Id,"REF-OBJECT");

    {
      ostringstream res; res << "{ ";
      // First value 3:USER-IN, 1:USER-OUT
      if(P.Vcod(Id,"USER-IN-OUT")=="OUT")
        res << 1 << ", -1, -1, -1, -1, " << Id << " }";
      else {
        res << 3 << ", ";
        // Check Energy Impact: 1:HIGH, 2:MEDIUM, 3:LOW
        int mE_IMPACT=P.Vcodid(Id,"ENERGY-IMPACT");
        if(mE_IMPACT<4)
          res << mE_IMPACT << ", ";
        else
          res << -1 << ", ";
        // Check CO2 Impact: 1:HIGH, 2:MEDIUM, 3:LOW
        int mCO2_IMPACT=P.Vcodid(Id,"CO2-IMPACT");
        if(mCO2_IMPACT<4)
          res << mCO2_IMPACT << ", ";
        else
          res << -1 << ", ";
        // Check PAY-BACK: 1:LONG, 2:MEDIUM, 3:SHORT
        double mPAY_BACK_NUM=P.Vnum(Id,"PAY-BACK-NUM");
        if(mPAY_BACK_NUM!=-5555) {
          res << mPAY_BACK_NUM << ", ";
          {
           ostringstream res; res << "USER";
           P.ModKey(Id,"PAY-BACK-SC",res.str());
          }
        }
        else {
          int mPAY_BACK=P.Vcodid(Id,"PAY-BACK");
          if(mPAY_BACK<4) {
            mPAY_BACK_NUM=T.Value("tPayBacks",1,mPAY_BACK);
            {
             ostringstream res; res << mPAY_BACK_NUM;
             P.ModKey(Id,"PAY-BACK-NUM",res.str());
            }
```

```
            res << mPAY BACK NUM << ", ";
            {
             ostringstream res; res << "USER";
             P.ModKey(Id,"PAY-BACK-SC",res.str());
            }
          }
          else
            res << -1 << ", ";
        }

        // Check CO2-SAVE-POUND: 1:GOOD, 2:FAIR, 3:POOR
        int mCO2 SAVE POUND=P.Vcodid(Id,"CO2-SAVE-POUND");
        if(mCO2 SAVE POUND<4)
          res << mCO2_SAVE_POUND << ", ";
        else
          res << -1 << ", ";
        // Finally add Object Id
        res << Id << " }";
      }
      P.ModKey(oId,P.Vcod(Id,"CODE"),res.str());       //UPdate   LNUM
recommendation in REC-PROJECT or REF-OBJECT
    }
  }
}

void REC PROJECT::TEST() {
  if(func report) cout << endl << "  " << "REC PROJECT::TEST";
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test REC P" << endl;
//  cout << "Fin de Test REC P" << endl;
  BEEMObject::TEST();
}

void REC PROJECT::Check(PROJECTDefaults& D, PROJECTTables& T, BOC& P)
{
  if(func report) cout << endl << "  " << "REC PROJECT::Check";


}


void  PROJECT::fLNUM CHECK(PROJECTDefaults& D,  PROJECTTables& T,  BOC
&P) {
  if(func_report) cout << endl << "  " << "PROJECT::fLNUM_CHECK";
/*

This function is for development use only and is used to check the
values of the
recommendation  LNUMs  which  are  produced  in  the  recommendations
functions for project, DHW and HVAC.
The LNUMS are used to write the EPC input file in epcgen inp.cpp

*/

  ostringstream os;
  string kName, kCat;
  string mLNUM CONF;

  //Check the values of the REC-PROJECT LNUMs
  cout << endl << endl << "REC-PROJECT LNUMs:" << endl;
  for (int i=1;i<=T.nRows("tREC Key PROJECT");i++) {
    kName=T.TValue("tREC_Key_PROJECT",i,1);
    kCat=T.TValue("tREC Key PROJECT",i,2);
    mLNUM CONF=P.Vtext(rpId,kName);
    cout << mLNUM_CONF << endl;
  }

  //Check the check the values of the HVAC LNUMs
```

```
  cout << endl << endl << "HVAC LNUMs:" << endl;
  for (int h=0;h<vHVAC.size();h++)
      for (int i=1;i<=T.nRows("tREC Key HVAC");i++) {
        kName=T.TValue("tREC Key HVAC",i,1);
        kCat=T.TValue("tREC_Key_HVAC",i,2);
        mLNUM CONF=P.Vtext(vHVAC[h].Id,kName);
        cout << mLNUM CONF << endl;
      }
  //Check the check the values of the DHW LNUMs
  cout << endl << endl << "DHW LNUMs:" << endl;
  for (int h=0;h<vDHG.size();h++)
      for (int i=1;i<=T.nRows("tREC Key DHW");i++) {
        kName=T.TValue("tREC Key DHW",i,1);
        kCat=T.TValue("tREC_Key_DHW",i,2);
        mLNUM CONF=P.Vtext(vDHG[h].Id,kName);
        cout << mLNUM CONF << endl;
      }

}



void PROJECT::Recommendation(PROJECTDefaults& D, PROJECTTables& T,
PROJECT& Notional) {
  if(func report) cout << endl << "  " << "PROJECT::Recommendation";
/*

This function calls the functions necessary to carry out the EPC
recommendations filtering
It carries out the following:

 - checking of REC-USER vector to populate REC-PROJECT in case the
user has modified any NCM recommendation.
 - call the object recommendations functions for each object type.
      In the object recommendations functions:
        - individual objects are assessed to be either Good, Fair or
Poor performers.
        - recommendations relating to that individual object only are
triggered and assessed.
 - call the project level recommendation functions:

      these are:

        - DHG_Recommendation
           - assesses overall energy and CO2 performance for DHW
(e.g. GOOD, FAIR, POOR, N/A)
           - assesses project level DHW recommendations
        - HVAC Recommendation
           - assesses overall energy and CO2 performance for heating
and cooling (e.g. GOOD, FAIR, POOR, N/A)
           - assesses project level recommendations for heating and
cooling
        - ZONE_Recommendation
           - assesses overall energy and CO2 performance for lighting
(e.g. GOOD, FAIR, POOR, N/A)
           - assesses project level recommendations for lighting
        - ENVELOPE Recommendation
           -  assesses project level envelope recommendations
including overheating risk check
        - SES Recommendation
           - project level SES recommendations
        - PVS_Recommendation
           - project level PVS recommendations
        - WIND Recommendation
           - project level wind generator recommendations
        - CHP Recommendation
           - currently only for reporting the CO2 emissions produced
(rather than displaced) by these systems
```

```
*/

  // First we check the REC-USER vector to populate REC-PROJECT in
case the user has modified any NCM recommendation.
  for (int i=0;i<vRU.size();i++)
    vRU[i].CheckNCM(D, T, P, rpId);

  //store necessary notional, typical and reference parameters in
DATA REC object TNR:
  TNR.notKWH_COOL=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-COOL");
//  TNR.typKWH HEAT=Typical.P.Vnum(Typical.dGEN[0].Id,"KWH-HEAT");
  TNR.typKWH HEAT=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
HEAT")*(2.0/T.Value("t2010_TYP",1,P.Vcodid(cId,"EPC-TYPE")));
  TNR.notKWH DHW=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-DHW");
  TNR.notKWH AUX=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-AUX");

  //The following code goes through all objects in turn.
  //In the object recommendations functions:
  //  1) individual objects are assessed to be either Good, Fair or
Poor performers.
  //  2) Recommendations relating to that object only are triggered
and assessed.

  //some object functions are currently blank as there are currently
no recommendations for that object
  //see individual functions in each object type for more detail

  for (int i=0;i<vC.size();i++)
    vC[i].Recommendation(D, T, P);    //not used - fabric is assessed
at project level
  for (int i=0;i<vG.size();i++)
    vG[i].Recommendation(D, T, P);    //not used - fabric is assessed
at project level
  for (int i=0;i<vCG.size();i++)
    vCG[i].Recommendation(D, T, P);   //not used
  for (int i=0;i<vHVAC.size();i++)
    vHVAC[i].Recommendation(D, T, P, TNR); //in this function the
HVAC heating (and cooling if used) generator is set as GOOD, FAIR or
POOR and individual heat (and cool) generator recommendations are
assessed
  for (int i=0;i<vDHG.size();i++)
    vDHG[i].Recommendation(D, T, P, TNR);  //in this function the DHW
generator is set as GOOD, FAIR or POOR and individual DHW generator
recommendations are assessed
  for (int i=0;i<vPV.size();i++)
    vPV[i].Recommendation(D, T, P);      //not used - renewables
currently only checked for existence at project level
  for (int i=0;i<vCHP.size();i++)
    vCHP[i].Recommendation(D, T, P);  //not used - currently no CHP
recommendations
  for (int i=0;i<vWG.size();i++)
    vWG[i].Recommendation(D, T, P);      //not used - renewables
currently only checked for existence at project level

  //In the project level recommendation functions:
  //  the overall project performance (for heating, cooling, lighting
and DHW) is computed (e.g. GOOD, FAIR, POOR, N/A)
  //  Specific recommendations are assessed for each recommendations
category (e.g. DHW, heating, cooling, envelope, fuel switching,
aux...)

  //DHW recommendation function
  DHG Recommendation(D,T,Notional);
  //HVAC recommendation function
  HVAC_Recommendation(D,T,Notional);
  //ZONE recommendation function
  ZONE Recommendation(D,T,Notional);
  //ENVELOPE recommendation function, included recommendations for
```

```
roof, walls and glazing
  //call after HVAC Recommendation as the % energy and CO2 for
heating is set in HVAC Recommendation
  ENVELOPE Recommendation(D,T,Notional);
  //solar energy system recommendation function (checks for presence
of these systems in project)
  SES Recommendation(D,T,Notional);
  //photovoltaic system recommendation function (checks for presence
of these systems in project)
  PVS Recommendation(D,T,Notional);
  //wind energy system recommendation function (checks for presence
of these systems in project)
  WIND Recommendation(D,T,Notional);
  //CHP recommendation function - calculates the percentage of total
CO2 emissions produced (rather than displaced) by
  //these systems in project
  CHP_Recommendation(D,T,Notional);


// Last Check Users recommendations
  for (int i=0;i<vRU.size();i++)
    vRU[i].Check(D, T, P);

//     {  ostringstream  logtext;  logtext  <<  "  OK"  <<  endl;
SendLog(logtext.str()); }
//     {  ostringstream  logtext;  logtext  <<  "Done"  <<  endl;
SendLog(logtext.str()); }

}


void PROJECT::Write Rec Keyword(PROJECTDefaults& D, PROJECTTables& T,
string EPC XX, int IMPACT ENERGY, int IMPACT CO2, double PAYBACK, int
CARBON SAVING POUND) {
  if(func report) cout << endl << "  " << "PROJECT::Write Rec Keyword
" << EPC XX << "  " << IMPACT ENERGY << "  " << IMPACT CO2 << "  " <<
PAYBACK << "  " << CARBON SAVING POUND;

/*

  Function to write keywords for EPC recommendations
  String EPC_XX is name of recommendation keyword e.g. EPC-H1

*/

  //first see what the value of the recommendations keyword is.
  //The assessor may have set some or all of the values or set that
this recommmendation should not be used etc...
  deque<double> mEPC XX=P.Vlnum(RP.Id,EPC XX);

  if (mEPC_XX.size()==1) {
    //user has not modified this recommendation
    for (int i=0;i<4;i++) mEPC XX.push back(0.0);
  }
  if (mEPC XX[0]!=1) {
    //user modified this recommendation
    //write the parameters which were not set by the user
    //ENERGY impact (1=HIGH/2=MEDIUM/3=LOW)
    if ((mEPC XX[1]==-1)||(mEPC XX[1]==0)) mEPC XX[1]=IMPACT ENERGY;
    //CO2 impact (1=HIGH/2=MEDIUM/3=LOW) always low
    if ((mEPC XX[2]==-1)||(mEPC XX[2]==0)) mEPC XX[2]=IMPACT CO2;
    //payback period (number)
    if ((mEPC_XX[3]==-1)||(mEPC_XX[3]==0)) mEPC_XX[3]=PAYBACK;
    //Carbon saving per pound (1=GOOD/2=FAIR/3=POOR)
    if                              ((mEPC XX[4]==-1)||(mEPC XX[4]==0))
mEPC_XX[4]=CARBON_SAVING_POUND;
    //Id is included (user has modified values)
    if (mEPC XX.size()==6) {
      int REC_USER_Id=(int)mEPC_XX[5];
```

```
      {
       ostringstream res; res << "{ ";
       for (int i=0;i<5;i++)
         res << mEPC_XX[i] << ", ";
       res << REC_USER_Id << " }";
       P.ModKey(RP.Id,EPC_XX,res.str());
      }
    }
    else {
      //calc in - sbem decided that the recommendation should be
included and set the values
      mEPC_XX[0]=4;
      {
       ostringstream res; res << "{ ";
       for (int i=0;i<4;i++)
         res << mEPC_XX[i] << ", ";
       res << mEPC_XX[4] << " }";
       P.ModKey(RP.Id,EPC_XX,res.str());
      }
    }
  }

} //end of PROJECT::Write Rec Keyword(...)


void  PROJECT::DHG Recommendation(PROJECTDefaults&  D,  PROJECTTables&
T, PROJECT& Notional) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"PROJECT::DHG Recommendation";
/*
  EPC recommendations function

  This function:
     - assesses overall energy and CO2 performance of DHW systems
     - calculates the percentage of CO2 and energy attributable to
DHW
     - assesses the impact of recommendations for DHW

   Recommendations for DHW:
      EPC-W1 - Install more efficient water heater
        -to assess impact, calculate modified energy and CO2 as ratio
between actual efficiency and 0.8
         (apply new efficiency to all DHW generators in project which
have an efficiency <=0.79 and >0.7)
      EPC-W2 - Consider point of use system
        -to assess impact, calculate modified energy and CO2 as ratio
between actual efficiency and 0.75
         (apply new efficiency to all DHW generators in project which
have an efficiency <=0.7)
      EPC-W3 - DHW storage insulation
        -to assess impact, calculate reduction in DHW energy and CO2
as 50% of storage losses
         (apply to all DHW generators in project which have storage)
      EPC-W4 - DHW secondary circulation time control
        -to assess impact, calculate reduction in DHW energy and CO2
as 30% of total DHW energy
         (apply to all DHW generators in project which have secondary
circulation)

*/

  //DHW heat generator efficiency:
  //total energy and CO2 for all DHW generators in project :
  double mQ_PRIM_TOTAL=0.0;
  //total energy weighted by energy price
  double mQ_PRIM_TOTAL_PRICE_WEIGHTED=0.0;
  double mKG_CO2_TOTAL=0.0;
  //energy and CO2 for 'GOOD' DHW generators:
  double mQ_PRIM_GOOD=0.0;
```

```
  double mKG CO2 GOOD=0.0;
  //energy and CO2 for 'FAIR' DHW generators:
  double mQ PRIM FAIR=0.0;
  double mKG CO2 FAIR=0.0;
  //energy and CO2 for 'POOR' DHW generators:
  double mQ PRIM POOR=0.0;
  double mKG CO2 POOR=0.0;

  //DHW storage:
  double mSTORE LOSS DEFAULT=0;
  double mSTORE_LOSS=0;
  double mSTORE LOSS ADJUSTED=0;
  double mSTORE LOSS TOTAL=0;
  //indicates whether DHW efficiency recommendation is triggered
  //1-> recommendation is triggered, 0->recommendation is not
triggered
  bool EPCW1=0;
  //indicates whether DHW recommendation on replacing system with
point of use system is triggered
  bool EPCW2=0;
  //indicates whether DHW storage recommendation is triggered
  bool EPCW3=0;
  //indicates whether DHW secondary circulation time switch
recommendation is triggered
  bool EPCW4=0;

  //used to store adjusted total building energy after
recommendations are applied:
  double mQ TOTAL ADJUSTED=0.0;
  double mCO2 TOTAL ADJUSTED=0.0;

  //total building energy for all end uses (to be used when assessing
impact of recommendation):
  //this is price weighted using factor from table
tFuel Price Weighting (depends on relative prices of fuels)
  //used in calculation of energy impact for recommendations
  double mQ TOTAL PRICE WEIGHTED=P.Vnum(gId,"Q-TOTAL-P-WEIGHTED");

  //variable to store CO2 for DHW after recommendation is applied
  double mDHW CO2 WEIGHTED=0.0;
  double mDHW CO2 ACT=0.0;

  //stores impact of recommendation on CO2 (1=HIGH/2=MEDIUM/3=LOW)
  //set to HIGH initially:
  int IMPACT_CO2=1;
  //stores impact of recommendation on energy (1=HIGH/2=MEDIUM/3=LOW)
  //set to HIGH initially:
  int IMPACT_ENERGY=1;
  //payback period
  double PAYBACK=0;
  //carbon saving per pound
  int CARBON SAVING POUND=1;

  //total building CO2 for all end uses (to be used when assessing
impact of recommendation)
  //and for calculating the % CO2 for this end use:
  //this keyword is the bulding CO2 if there were no renewables:
  double mKG CO2 PROJECT=P.Vnum(gId,"KG-CO2-EXCL-RENEWABLES");

  //variable to store energy for DHW after recommendation is applied
  double mDHW ENERGY WEIGHTED PRICE WEIGHTED=0.0;
  //variable to store energy for each DHW generator
  double mDHW_ENERGY_ACT=0.0;

  double mFUEL CARBON INTENSITY;
  double Fuel_Payback=0.0;

  //to store price weighting factor for fuel
  double Fuel_Price_Weighting=0.0;
```

```
  // to store weighted-average performance for all systems in the
building
  double Weighted Energy=0.0;
  double Weighted_CO2=0.0;

  //go through all DHW generators
  for (int i=0;i<vDHG.size();i++) {
    //only include here if heat generator is not HVAC system heat
generator
    if (P.Vcodid(vDHG[i].Id,"HEAT-GEN-TYPE")>1) {
       if((P.Vcodid(cId,"BR-TYPE")==5)     ||     (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,P.Vcodid(vDHG[i].Id,"FUE
L-TYPE"));
       else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vDHG[i].Id,"FUEL-
TYPE"));
       //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
       //used in calculation of energy impact

Fuel Price Weighting=T.Value("tFuel Price Weighting",1,P.Vcodid(vDHG[
i].Id,"FUEL-TYPE"));
       mQ PRIM TOTAL+=P.Vnum(vDHG[i].Id,"Q-PRIM");
       mQ PRIM TOTAL PRICE WEIGHTED+=P.Vnum(vDHG[i].Id,"Q-
PRIM")*Fuel Price Weighting;
       mKG CO2 TOTAL+=(P.Vnum(vDHG[i].Id,"KWH-
DHW")*mFUEL CARBON INTENSITY);
       //fuel  payback  adjustment,  weight  with  energy  -  used  to
calculate carbon saved/pound spent

Fuel Payback+=T.Value("tFuel Payback",1,P.Vcodid(vDHG[i].Id,"FUEL-
TYPE"))*P.Vnum(vDHG[i].Id,"Q-PRIM");

       Weighted Energy+=(P.Vnum(vDHG[i].Id,"Q-
PRIM"))*(P.Vnum(vDHG[i].Id,"HEAT-GEN-EFF"));
       Weighted CO2+=(P.Vnum(vDHG[i].Id,"KWH-
DHW")*mFUEL CARBON INTENSITY)*(P.Vnum(vDHG[i].Id,"HEAT-GEN-EFF"));
    }

    //-----DHW storage-----
    //Check whether there is storage
    if(P.Vcodid(vDHG[i].Id,"STORE-SYSTEM")==1) {
      //storage volume for dhw generator
      mSTORE_LOSS=P.Vnum(vDHG[i].Id,"STORE-LOSS");
      //default storage volume for dhw generator, weighted by energy
price
      //necessary to weight for calculation below for energy impact
      mSTORE LOSS DEFAULT=P.Vnum(vDHG[i].Id,"STORE-LOSS-
DEFAULT")*Fuel Price Weighting;
      //store  total  storage  losses  for  whole  building,  weighted  by
energy price
      //necessary to weight for calculation below for energy impact
      mSTORE_LOSS_TOTAL+=P.Vnum(vDHG[i].Id,"STORE-
LOSS")*Fuel Price Weighting;
      //If  heat  loss  >  default  loss  *  0.9,  trigger  recommendation
EPCW3
      if (mSTORE LOSS>D.Value("REC DEF12")*mSTORE LOSS DEFAULT) {
        //trigger  recommendation  EPCW3:  improve  insulation  on  DHW
storage
        EPCW3=1;
      }
    }

    //-----DHW secondary cirulation-----
    //If  there  is  secondary  DHW  circulation  and  there  is  no  time
```

```
control, trigger recommendation EPCW4
    //Check whether there is secondary cirulation
    if (P.Vcodid(vDHG[i].Id,"SEC-CIRCULATION")==1) {
      //check if there is time control, if not trigger recommendation
EPCW4: add time control to sec circulation
      if (P.Vcodid(vDHG[i].Id,"SEC-CIRC-TS")==2)
         EPCW4=1;
    }

  }  //end of loop through all DHW generators


  //divide by total DHW energy
  if (mQ_PRIM_TOTAL!=0) {
     // Average Fuel Payback for DHW of the whole building weighted
by energy of all DHW systems
     Fuel_Payback/=mQ_PRIM_TOTAL;
     // Average energy performance for DHW of the whole building
weighted by energy of all DHW systems
     Weighted_Energy/=mQ_PRIM_TOTAL;
  }
  else {
    Fuel Payback=0.0;
    Weighted Energy=0.0;
  }

  //  divide by total DHW CO2
  if (mKG CO2 TOTAL!=0) {
     //  Average CO2 performance for DHW of the whole building
weighted by CO2 of all DHW systems
     Weighted CO2/=mKG CO2 TOTAL;
  }
  else {
    Weighted CO2=0.0;
  }


  //------DHW energy performance - whole building GOOD/ FAIR/ POOR---
---

  string PERFORMANCE TOTAL ENERGY="NOT APPLICABLE";

  //For whole building - is DHW ENERGY USE is classified as GOOD,
FAIR or POOR
  //    this decides whether you need to trigger recommendation
associated with FAIR or POOR.

  if (Weighted_Energy>0.0) {
      if (Weighted Energy>D.Value("REC DEF1a")) {
      //if averaged energy efficiency is (>x) building end use is
GOOD
        PERFORMANCE TOTAL ENERGY="GOOD";
      }
      else { // i.e., efficiency will be either FAIR or POOR
           //trigger recommendation EPCW1: more efficient water
heater
           EPCW1=1;
           //If x >= average efficiency > y, classify energy
performance as "fair":
           if (Weighted_Energy>D.Value("REC_DEF2")) {
              PERFORMANCE TOTAL ENERGY="FAIR";
           }
           else {  // i.e., average efficiency =< y, building end use
is POOR
                PERFORMANCE TOTAL ENERGY="POOR";
                //trigger recommendation EPCW2: replace with point of
use system
                EPCW2=1;
           }
```

```
      }
  }


  //------CO2 performance - whole building GOOD/ FAIR/ POOR------

  string PERFORMANCE TOTAL CO2="NOT APPLICABLE";

  //For whole building - is DHW CO2 is classified as GOOD, FAIR or
POOR
  //    this  decides  whether  you  need  to  trigger  recommendation
associated with FAIR or POOR.

  if (Weighted_CO2>0.0) {
      if (Weighted CO2>D.Value("REC DEF1a")) {
      //if averaged CO2 efficiency is (>x) building end use is GOOD
        PERFORMANCE_TOTAL_CO2="GOOD";
      }
      else { // i.e., efficiency will be either FAIR or POOR
           //trigger  recommendation  EPCW1:  more  efficient  water
heater
           EPCW1=1;
           //If  x  >=  average  efficiency  >  y,  classify  energy
performance as "fair":
           if (Weighted CO2>D.Value("REC DEF2")) {
              PERFORMANCE TOTAL CO2="FAIR";
           }
           else {  // i.e., average efficiency =< y, building end use
is POOR
                PERFORMANCE TOTAL CO2="POOR";
                //trigger recommendation EPCW2: replace with point of
use system
                EPCW2=1;
           }
      }
  }

  //write end use average efficiency keywords
  //PERF-DHW-ENERGY and PERF-DHW-CO2 Codes: 1 - GOOD, 2 - FAIR, 3 -
POOR, 4 - NOT APPLICABLE
 {
  ostringstream res; res << PERFORMANCE_TOTAL_ENERGY;
  P.ModKey(RP.Id,"PERF-DHW-ENERGY",res.str());            //  Averaged
system energy efficiency for DHW
 }
 {
  ostringstream res; res << PERFORMANCE TOTAL CO2;
  P.ModKey(RP.Id,"PERF-DHW-CO2",res.str());        // Averaged system
CO2 efficiency for DHW
 }


   //---- recommendation EPC-W1 ----

   // more efficient water heater

  //assess       impact       of       recommendation       EPC-W1:
(1=GOOD/2=FAIR/3=POOR/4=N/A)
  if (EPCW1) {

    //variables  to  store  energy  and  CO2  for  all  DHW  system  after
recommendation is applied where applicable
    mDHW_ENERGY_WEIGHTED_PRICE_WEIGHTED=0.0;
    mDHW CO2 WEIGHTED=0.0;
    //variables to store energy and CO2 for each DHW generator
    mDHW_ENERGY_ACT=0.0;
    mDHW CO2 ACT=0.0;

    //assess impact of recommendation EPCW1 on energy and CO2
```

```
    //go through all DHW generators in project
    for (int i=0;i<vDHG.size();i++) {
      //only include here if heat generator is not HVAC system heat
generator
      if (P.Vcodid(vDHG[i].Id,"HEAT-GEN-TYPE")>1) {
        if((P.Vcodid(cId,"BR-TYPE")==5)     ||     (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,P.Vcodid(vDHG[i].Id,"FUE
L-TYPE"));
        else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vDHG[i].Id,"FUEL-
TYPE"));
        //if eff <= D.Value("REC DEF1a") store energy  recommendation
(set efficiency to D.Value("REC DEF1"))
        if   (P.Vnum(vDHG[i].Id,"HEAT-GEN-EFF")<=D.Value("REC DEF1a"))
{
          mDHW ENERGY ACT=P.Vnum(vDHG[i].Id,"Q-PRIM");
          mDHW_CO2_ACT=P.Vnum(vDHG[i].Id,"KWH-
DHW")*mFUEL CARBON INTENSITY;
          if ((D.Value("REC DEF1"))!=0) {
            //weight energy => energy*(old eff/D.Value("REC DEF1"))

mDHW ENERGY WEIGHTED PRICE WEIGHTED+=mDHW ENERGY ACT*(P.Vnum(vDHG[i].
Id,"HEAT-GEN-EFF")/D.Value("REC DEF1"))*Fuel Price Weighting;
            //weight CO2 => CO2*(old eff/D.Value("REC DEF1"))
            mDHW CO2 WEIGHTED+=mDHW CO2 ACT*(P.Vnum(vDHG[i].Id,"HEAT-
GEN-EFF")/D.Value("REC DEF1"));
          }
        }
        else {
          //if eff > D.Value("REC DEF1a") add energy directly
          mDHW ENERGY WEIGHTED PRICE WEIGHTED+=P.Vnum(vDHG[i].Id,"Q-
PRIM")*Fuel Price Weighting;
          //if eff > D.Value("REC DEF1a") add CO2 directly
          mDHW CO2 WEIGHTED+=P.Vnum(vDHG[i].Id,"KWH-
DHW")*mFUEL CARBON INTENSITY;
        }
      }
    }

    //variable for adjusted Total Building energy
    mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mQ_PRIM_TOTAL_PRICE_WEIGHTED+mDHW_ENERGY_WEIGHTED_PRICE_WEIGHTED;
    mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
mKG CO2 TOTAL+mDHW CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ_TOTAL_ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D,                   mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

    //calculate  payback  (PAYBACK)  and  Carbon  saving  per  pound
(CARBON SAVING POUND)
    //Multiply default payback by 10 and divide by NOTIONAL building
DHW consumption (kWh/m2.year)
    double mAREA=P.Vnum(gId,"AREA");
    double      mNOT_KWH_DHW=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
DHW");
    if((mAREA!=0)&&(mNOT KWH DHW!=0))
      PAYBACK=D.Value("dEPC_W1")*10.0/(mNOT_KWH_DHW/mAREA);
    else
      PAYBACK=0.0;

    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON_SAVING_POUND value (GOOD, FAIR, POOR)
```

```
CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    string EPC XX="EPC-W1";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);

  }

   //---- END recommendation EPC-W1 ----



   //---- recommendation EPC-W2 ----

   // point of use DHW system

  //assess       impact       of       recommendation       EPC-W2:
(1=GOOD/2=FAIR/3=POOR/4=N/A)
  if (EPCW2) {

    //variables to store energy and CO2 for all DHW system after
recommendation is applied where applicable
    mDHW ENERGY WEIGHTED PRICE WEIGHTED=0.0;
    mDHW CO2 WEIGHTED=0.0;
    //variables to store energy and CO2 for each DHW generator
    mDHW ENERGY ACT=0.0;
    mDHW CO2 ACT=0.0;

    //assess impact of recommendation EPCW1 on energy and CO2
    //go through all DHW generators in project
    for (int i=0;i<vDHG.size();i++) {
      //only include here if heat generator is not HVAC system heat
generator
      if (P.Vcodid(vDHG[i].Id,"HEAT-GEN-TYPE")>1) {
        if((P.Vcodid(cId,"BR-TYPE")==5)     ||     (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL_CARBON_INTENSITY=T.Value("tKGCO2_IR",1,P.Vcodid(vDHG[i].Id,"FUE
L-TYPE"));
        else //  all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vDHG[i].Id,"FUEL-
TYPE"));
        //if eff <= D.Value("REC_DEF2") store energy  recommendation
(set efficiency to D.Value("REC DEF2a"))
        if (P.Vnum(vDHG[i].Id,"HEAT-GEN-EFF")<=D.Value("REC DEF2")) {
          mDHW_ENERGY_ACT=P.Vnum(vDHG[i].Id,"Q-PRIM");
          mDHW CO2 ACT=P.Vnum(vDHG[i].Id,"KWH-
DHW")*mFUEL CARBON INTENSITY;
          if ((D.Value("REC_DEF2a"))!=0) {
            //weight energy => energy*(old eff/D.Value("REC DEF2a"))

mDHW_ENERGY_WEIGHTED_PRICE_WEIGHTED+=mDHW_ENERGY_ACT*(P.Vnum(vDHG[i].
Id,"HEAT-GEN-EFF")/D.Value("REC DEF2a"))*Fuel Price Weighting;
            //weight CO2 => CO2*(old eff/D.Value("REC DEF2a"))
            mDHW_CO2_WEIGHTED+=mDHW_CO2_ACT*(P.Vnum(vDHG[i].Id,"HEAT-
GEN-EFF")/D.Value("REC DEF2a"));
          }
        }
        else {
          //if eff > D.Value("REC DEF2") add energy directly
          mDHW_ENERGY_WEIGHTED_PRICE_WEIGHTED+=P.Vnum(vDHG[i].Id,"Q-
PRIM")*Fuel Price Weighting;
          //if eff > D.Value("REC DEF2") add CO2 directly
          mDHW_CO2_WEIGHTED+=P.Vnum(vDHG[i].Id,"KWH-
DHW")*mFUEL CARBON INTENSITY;
        }
      }
```

```
    }

    //variable for adjusted Total Building energy
    mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mQ_PRIM_TOTAL_PRICE_WEIGHTED+mDHW_ENERGY_WEIGHTED_PRICE_WEIGHTED;
    mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
mKG CO2 TOTAL+mDHW CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ_TOTAL_ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D,                   mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
    //Multiply default payback by 10 and divide by NOTIONAL building
DHW consumption (kWh/m2.year)
    double mAREA=P.Vnum(gId,"AREA");
    double      mNOT_KWH_DHW=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
DHW");
    if((mAREA!=0)&&(mNOT KWH DHW!=0))
      PAYBACK=D.Value("dEPC W2")*10.0/(mNOT KWH DHW/mAREA);
    else
      PAYBACK=0.0;

    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    string EPC XX="EPC-W2";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);

  }

   //---- END recommendation EPC-W2 ----



   //---- recommendation EPC-W3 ----

  //storage losses

  if (EPCW3) {

    //variables to store energy and CO2 for all DHW system after
recommendation is applied where applicable
    mDHW_ENERGY_WEIGHTED_PRICE_WEIGHTED=0.0;
    mDHW CO2 WEIGHTED=0.0;
    //variables to store energy and CO2 for each DHW generator
    mDHW_ENERGY_ACT=0.0;
    mDHW CO2 ACT=0.0;

    //assess impact of recommendation EPCW3 on energy and CO2
    //go through all DHW generators in project
    for (int i=0;i<vDHG.size();i++) {
        //storage losses for dhw generator
        double mSTORE LOSS=P.Vnum(vDHG[i].Id,"STORE-LOSS");
        //default storage losses for dhw generator
        double      mSTORE_LOSS_DEFAULT=P.Vnum(vDHG[i].Id,"STORE-LOSS-
DEFAULT");
        if((P.Vcodid(cId,"BR-TYPE")==5)     ||     (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,P.Vcodid(vDHG[i].Id,"FUE
L-TYPE"));
```

```
        else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vDHG[i].Id,"FUEL-
TYPE"));
        //If heat loss > default loss * 0.9, reduce DHW energy by 50%
of storage losses
        if                                ((P.Vcodid(vDHG[i].Id,"STORE-
SYSTEM")==1)&&(mSTORE_LOSS>D.Value("REC_DEF12")*mSTORE_LOSS_DEFAULT))
{
        //Note that storage losses for dhw generator are in MJ/month
so is multiplied by 12 to get annual figure
          mDHW ENERGY ACT=P.Vnum(vDHG[i].Id,"Q-PRIM");
          mDHW CO2 ACT=P.Vnum(vDHG[i].Id,"KWH-
DHW")*mFUEL_CARBON_INTENSITY;
          //weight          energy          =>          energy-(storage
losses*12.0*D.Value("REC DEF13"))
          mDHW_ENERGY_WEIGHTED_PRICE_WEIGHTED+=(mDHW_ENERGY_ACT-
(mSTORE LOSS*12.0*D.Value("REC DEF13")))*Fuel Price Weighting;
          //weight            CO2           =>           CO2-((storage
losses*12.0*D.Value("REC_DEF13"))*(mFUEL_CARBON_INTENSITY/3.6))
          //Note that the following equation includes a division by
3.6 to convert MJ to kWh before calculating the carbon
          mDHW CO2 WEIGHTED+=mDHW CO2 ACT-
((mSTORE LOSS*12.0*D.Value("REC DEF13"))*(mFUEL CARBON INTENSITY/3.6)
);
        }
        else {
          //if  there  is  no  storage  or  losses  are  better  than
defaults, add energy directly
          mDHW ENERGY WEIGHTED PRICE WEIGHTED+=P.Vnum(vDHG[i].Id,"Q-
PRIM")*Fuel Price Weighting;
          //if  there  is  no  storage  or  losses  are  better  than
defaults, add CO2 directly
          mDHW CO2 WEIGHTED+=P.Vnum(vDHG[i].Id,"KWH-
DHW")*mFUEL CARBON INTENSITY;
        }
    }

    //variable for adjusted Total Building energy
    mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mQ PRIM TOTAL PRICE WEIGHTED+mDHW ENERGY WEIGHTED PRICE WEIGHTED;
    mCO2_TOTAL_ADJUSTED=mKG_CO2_PROJECT-
mKG CO2 TOTAL+mDHW CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D,                   mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

    //calculate  payback  (PAYBACK)  and  Carbon  saving  per  pound
(CARBON SAVING POUND)
    //Multiply default payback by 10 and divide by NOTIONAL building
DHW consumption (kWh/m2.year)
    double mAREA=P.Vnum(gId,"AREA");
    double      mNOT KWH DHW=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
DHW");
    if((mAREA!=0)&&(mNOT KWH DHW!=0))
      PAYBACK=D.Value("dEPC W3")*10.0/(mNOT KWH DHW/mAREA);
    else
      PAYBACK=0.0;

    double CARBON_SAVING_NO=PAYBACK*Fuel_Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    string EPC_XX="EPC-W3";
```

```
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);

  }

   //---- END recommendation EPC-W3 ----



   //---- recommendation EPC-W4 ----
   //secondary circulation losses

  if (EPCW4) {
    //assess impact by reducing DHW energy by 30%:

    //variables to store energy and CO2 for all DHW system after
recommendation is applied where applicable
    mDHW ENERGY WEIGHTED PRICE WEIGHTED=0.0;
    mDHW CO2 WEIGHTED=0.0;
    //variables to store energy and CO2 for each DHW generator
    mDHW ENERGY ACT=0.0;
    mDHW CO2 ACT=0.0;

    //assess impact of recommendation EPCW4 on energy and CO2
    //go through all DHW generators in project
    for (int i=0;i<vDHG.size();i++) {
       if((P.Vcodid(cId,"BR-TYPE")==5)     ||     (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,P.Vcodid(vDHG[i].Id,"FUE
L-TYPE"));
       else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vDHG[i].Id,"FUEL-
TYPE"));
       //If there is no time control, assess impact by reducing DHW
energy by 30%
       if                                    ((P.Vcodid(vDHG[i].Id,"SEC-
CIRCULATION")==1)&&(P.Vcodid(vDHG[i].Id,"SEC-CIRC-TS")==2)) {
         mDHW ENERGY ACT=P.Vnum(vDHG[i].Id,"Q-PRIM");
         mDHW CO2 ACT=P.Vnum(vDHG[i].Id,"KWH-
DHW")*mFUEL_CARBON_INTENSITY;
         //weight energy => energy*(1.0-D.Value("REC DEF14"))
         mDHW ENERGY WEIGHTED PRICE WEIGHTED+=mDHW ENERGY ACT*(1.0-
D.Value("REC_DEF14"))*Fuel_Price_Weighting;
         //weight CO2 => CO2*(1.0-D.Value("REC DEF14"))
         mDHW CO2 WEIGHTED+=mDHW CO2 ACT*(1.0-D.Value("REC DEF14"));
       }
       else {
         //if there is time control on the secondary circulation,
add energy directly
         mDHW ENERGY WEIGHTED PRICE WEIGHTED+=P.Vnum(vDHG[i].Id,"Q-
PRIM")*Fuel Price Weighting;
         //if there is time control on the secondary circulation,
add CO2 directly
         mDHW CO2 WEIGHTED+=P.Vnum(vDHG[i].Id,"KWH-
DHW")*mFUEL_CARBON_INTENSITY;
       }
    }

    //variable for adjusted Total Building energy
    mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mQ_PRIM_TOTAL_PRICE_WEIGHTED+mDHW_ENERGY_WEIGHTED_PRICE_WEIGHTED;
    mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
mKG CO2 TOTAL+mDHW CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ_TOTAL_ADJUSTED);
```

```
    IMPACT CO2=BOrec.fIMPACT CO2(D,                    mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON_SAVING_POUND)
    //Multiply default payback by 10 and divide by NOTIONAL building
DHW consumption (kWh/m2.year)
    double mAREA=P.Vnum(gId,"AREA");
    double      mNOT KWH DHW=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
DHW");
    if((mAREA!=0)&&(mNOT_KWH_DHW!=0))
      PAYBACK=D.Value("dEPC W4")*10.0/(mNOT KWH DHW/mAREA);
    else
      PAYBACK=0.0;

    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON_SAVING_POUND=BOrec.fCARBON_SAVING_POUND(D,CARBON_SAVING_NO);

    string EPC XX="EPC-W4";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);

  }
   //---- END recommendation EPC-W4 ----


  //percentage of CO2 attributable to this end use:
  double DHW PERCENT CO2=0.0;
  if (mKG CO2 PROJECT!=0.0)
    DHW PERCENT CO2=(mKG CO2 TOTAL/mKG CO2 PROJECT)*100.0;

  {
  ostringstream res; res << DHW PERCENT CO2;
  P.ModKey(RP.Id,"DHW-PERCENT-CO2",res.str());
  }

 //---- Overall energy and carbon performance for DHW ----
  //categorise DHW as good/fair/poor, PERF-DHW-SYSTEM-ENERGY
  //if actual<notional, classify Overall energy performance for DHW
as good
  //if notional<=actual<typical, classify Overall energy performance
for DHW as fair
  //otherwise classify Overall energy performance for DHW as fair

  //energy
  string mPERF DHW SYSTEM ENERGY="NOT APPLICABLE";

  if (P.Vnum(gId,"KWH-DHW")>0.0) {
    if                                          (P.Vnum(gId,"KWH-
DHW")<Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-DHW")) {
      mPERF_DHW_SYSTEM_ENERGY="GOOD";
    }
    else                    if                  (P.Vnum(gId,"KWH-
DHW")<Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
DHW")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")))) {
           mPERF DHW SYSTEM ENERGY="FAIR";
         }
         else {
              mPERF DHW SYSTEM ENERGY="POOR";
         }
  }

  {
  ostringstream res; res << mPERF DHW SYSTEM ENERGY;
  P.ModKey(RP.Id,"PERF-DHW-SYSTEM-ENERGY",res.str());      // Overall
energy performance for DHW
```

```
  }

  //CO2
  string mPERF DHW SYSTEM CO2="NOT APPLICABLE";

  if (P.Vnum(gId,"KG-CO2-DHW")>0.0) {
    if                                          (P.Vnum(gId,"KG-CO2-
DHW")<Notional.P.Vnum(Notional.dGEN[0].Id,"KG-CO2-DHW")) {
      mPERF DHW SYSTEM CO2="GOOD";
    }
    else                    if                  (P.Vnum(gId,"KG-CO2-
DHW")<Notional.P.Vnum(Notional.dGEN[0].Id,"KG-CO2-
DHW")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")))) {
           mPERF_DHW_SYSTEM_CO2="FAIR";
         }
         else {
              mPERF_DHW_SYSTEM_CO2="POOR";
         }
  }

  {
  ostringstream res; res << mPERF DHW SYSTEM CO2;
  P.ModKey(RP.Id,"PERF-DHW-SYSTEM-CO2",res.str());      //Overall CO2
performance for DHW
  }

  //---- End of Overall energy and carbon performance for DHW ----

}



void PROJECT::HVAC Recommendation(PROJECTDefaults& D, PROJECTTables&
T, PROJECT& Notional) {
  if(func report)    cout    <<    endl    <<    "         "    <<
"PROJECT::HVAC Recommendation";
/*
  EPC recommendations function

  This function:
     - assesses overall energy and CO2 performance of all HVAC-
SYSTEMs in project for both heating and cooling
     - calculates the percentage of CO2 and energy attributable to
heating and cooling
     - assesses the impact of recommendations for heating and cooling

    Project level recommendations for HVAC-SYSTEM are:

      1) heating
         EPC-H1 - install high efficiency boiler
           -to assess impact, Calculate modified energy/ CO2 as ratio
between actual efficiency and 0.81.
         EPC-H3 - install condensing boiler
           -to assess impact, calculate modified energy/ CO2 as ratio
between actual efficiency and 0.89.
         EPC-R1   consider GSHP
           -to assess impact, calculate modified energy as ratio
between actual efficiency and 3.1.
         EPC-R5   consider ASHP
           -to assess impact, calculate modified energy as ratio
between actual efficiency and 2.2.

      2) cooling
         EPC-C2   install high efficency chiller
           -to assess impact, calculate modified energy as ratio
between actual efficiency and 2.5
         EPC-C3   Inspect and seal ductwork
           -to assess impact, calculate modified energy by:
```

```
                If duct and AHU leakage is between 5% and 10% (fair):
                -  Reduce  cooling  energy  by  P%  where  P  is:  VAV,
constant volume, multizone, terminal reheat, dual duct P=5%, Fan
coil, induction P = 2%, Chilled ceiling, chilled beam P= 0.5%
                If duct and AHU leakage is less than 10% (poor):
                -  Reduce  cooling  energy  by  P%  where  P  is:  VAV,
constant volume, multizone, terminal reheat, dual duct P=10%, Fan
coil, induction P = 4%, Chilled ceiling, chilled beam P= 1%

*/

//1)heating
  //total  heating  energy  and  CO2  for  all  HVAC  heat  generators  in
project:
  double mQ HEAT TOTAL=0.0;
  //total energy weighted by energy price
  double mQ_HEAT_TOTAL_PRICE_WEIGHTED=0.0;
  double mKG CO2 HEAT TOTAL=0.0;
  //energy and CO2 for 'GOOD' HVAC heat generators:
  double mQ_HEAT_GOOD=0.0;
  double mKG CO2 HEAT GOOD=0.0;
  //energy and CO2 for 'FAIR' HVAC heat generators:
  double mQ HEAT FAIR=0.0;
  double mKG CO2 HEAT FAIR=0.0;
  //energy and CO2 for 'POOR' HVAC heat generators:
  double mQ HEAT POOR=0.0;
  double mKG CO2 HEAT POOR=0.0;

  //total energy and CO2 for particular HVAC (inlcudes space heating
and DHW)
  double mQ HEAT TOTAL HVAC=0.0;
  //price weighted total energy and CO2 for particular HVAC (includes
space heating and DHW)
  double mQ HEAT TOTAL HVAC PW=0.0;
  double mKG CO2 TOTAL HVAC=0.0;

  double mFUEL CARBON INTENSITY=0.0;
  double Fuel Payback=0.0;

  //to store price weighting factor for fuel
  double Fuel Price Weighting=0.0;

  double mQ HEAT DHW=0;
  double mKWH HEAT DHW=0;

  double mQ HEAT=0.0;
  double mKG CO2 HEAT=0.0;

  //keywords for recommendations EPCR1 (GSHP) and EPCR5 (ASHP)
  //1->  recommendation  is  triggered,  0->recommendation  is  not
triggered
  bool EPCR1=0;      //install ground source heat pump (GSHP)
  bool EPCR5=0;      //install air source heat pump (ASHP)

  //fuel switching recommendations:
  bool EPCF2=0;      //Coal to natural gas
  bool EPCF3=0;      //Coal to biomass
  bool EPCF6=0;      //Coal to oil
  bool EPCF1=0;      //Oil or LPG to natural gas
  bool EPCF4=0;      //Oil or LPG to biomass
  bool EPCF5=0;      //gas to biomass

  //heating controls recommendations
  bool EPCH2=0;     //heating central time control
  bool EPCH5=0;     //local time control
  bool EPCH6=0;     //room temperature control
  bool EPCH7=0;     //Heating optimum start and stop control
  bool EPCH8=0;     //heating weather compensation controls
```

```
  //heat generator efficiency recommendations:
  bool EPCH3=0;      //condensing boiler
  bool  EPCH3 temp=0; //condensing boiler - temporary trigger until
recommendation is confirmed
  bool EPCH1=0;      //high efficiency boiler
  bool EPCH4=0;      //improve efficiency above default

  //used   to   store   adjusted   total   building   energy   after
recommendations are applied:
  double mQ TOTAL ADJUSTED=0.0;
  double mCO2_TOTAL_ADJUSTED=0.0;

  //total building energy for all end uses (to be used when assessing
impact of recommendation):
  double mQ TOTAL=P.Vnum(gId,"Q-TOTAL");
  //price weighted total building energy for all end uses
  //price  weighted  using  factor  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
  //used in calculation of energy impact for recommendations
  double mQ_TOTAL_PRICE_WEIGHTED=P.Vnum(gId,"Q-TOTAL-P-WEIGHTED");
  //variable  to  store  Price  weighted  energy  for  HVAC  heat  after
recommendation is applied
  double mHVAC HEAT ENERGY WT PW=0.0;

 //variable to store CO2 for HVAC after recommendation is applied
  double mHVAC HEAT CO2 WEIGHTED=0.0;

  //stores impact of recommendation on CO2 (1=HIGH/2=MEDIUM/3=LOW)
  int IMPACT CO2=1;
  //stores impact of recommendation on energy (1=HIGH/2=MEDIUM/3=LOW)
  int IMPACT ENERGY=1;

  //payback period
  double PAYBACK=0;
  //carbon saving per pound
  int CARBON SAVING POUND=1;

  //total building CO2 for all end uses (to be used when assessing
impact of recommendation)
  //and for calculating the % CO2 for this end use:
  //this keyword is the bulding CO2 if there were no renewables:
  double mKG_CO2_PROJECT=P.Vnum(gId,"KG-CO2-EXCL-RENEWABLES");

  //energy for one HVAC system
  double mQ_HEAT_THIS_HVAC=0.0;
  //price weighted energy for one HVAC system
  double mQ HEAT THIS HVAC PW=0.0;
  double mKG_CO2_HEAT_THIS_HVAC=0.0;
  double mQ DHW THIS HVAC=0.0;
  //price weighted energy for one HVAC system
  double mQ_DHW_THIS_HVAC_PW=0.0;
  double mKG CO2 DHW THIS HVAC=0.0;

  // to store weighted-average performance for all systems in the
building
  double Weighted Energy=0.0;
  double Weighted_CO2=0.0;

  //go through all HVAC systems
  for (int i=0;i<vHVAC.size();i++)  {
    //check if HVAC system provides heating
    if (T.Value("tSYS HEAT",1,P.Vcodid(vHVAC[i].Id,"TYPE"))==1) {
      //carbon intensity of fuel
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,P.Vcodid(vHVAC[i].Id,"FU
EL-TYPE"));
      else // all UK
```

```
mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE"));
      //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
      //used in calculation of energy impact

Fuel_Price_Weighting=T.Value("tFuel_Price_Weighting",1,P.Vcodid(vHVAC
[i].Id,"FUEL-TYPE"));
      deque<int>    mDHW SERVED REF=P.Vlrefid(vHVAC[i].Id,"DHW-SERVED-
REF");

      //primary energy used for space heating:
      mQ_HEAT_THIS_HVAC=P.Vnum(vHVAC[i].Id,"Q-PRIM-HEAT");
      mQ HEAT THIS HVAC PW=P.Vnum(vHVAC[i].Id,"Q-PRIM-
HEAT")*Fuel Price Weighting;
      mKG_CO2_HEAT_THIS_HVAC=(P.Vnum(vHVAC[i].Id,"KWH-
HEAT")*mFUEL CARBON INTENSITY);
      mQ DHW THIS HVAC=0.0;
      mQ_DHW_THIS_HVAC_PW=0.0;
      mKG CO2 DHW THIS HVAC=0.0;

      //primary energy used for DHW:
      for (int j=0;j< mDHW SERVED REF.size();j++) {
        mQ DHW THIS HVAC+=P.Vnum(mDHW SERVED REF[j],"Q-PRIM");
        mQ DHW THIS HVAC PW+=P.Vnum(mDHW SERVED REF[j],"Q-
PRIM")*Fuel Price Weighting;
        mKG CO2 DHW THIS HVAC+=P.Vnum(mDHW SERVED REF[j],"KWH-
DHW")*mFUEL CARBON INTENSITY;
      }

      //fuel  payback  adjustment,  weight  with  energy  -  used  to
calculate carbon saved/pound spent

Fuel Payback+=T.Value("tFuel Payback",1,P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE"))*(mQ HEAT THIS HVAC+mQ DHW THIS HVAC);


Weighted_Energy+=(mQ_HEAT_THIS_HVAC+mQ_DHW_THIS_HVAC)*(P.Vnum(vHVAC[i
].Id,"HEAT-GEN-SEFF"));

Weighted_CO2+=(mKG_CO2_HEAT_THIS_HVAC+mKG_CO2_DHW_THIS_HVAC)*(P.Vnum(
vHVAC[i].Id,"HEAT-GEN-SEFF"));

      mQ_HEAT_TOTAL+=mQ_HEAT_THIS_HVAC+mQ_DHW_THIS_HVAC;

mQ HEAT TOTAL PRICE WEIGHTED+=mQ HEAT THIS HVAC PW+mQ DHW THIS HVAC P
W;

mKG CO2 HEAT TOTAL+=mKG CO2 HEAT THIS HVAC+mKG CO2 DHW THIS HVAC;


      //-----GSHP and ASHP recommendation for this HVAC-----
      //if fuel is electricity, check efficiency
      //if  less  than  D.Value("REC DEF7"),  trigger  GSHP  and  ASHP
recommendation
      if                                    (P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE")==(int)D.Value("dELEC CODID")) {
        if   (P.Vnum(vHVAC[i].Id,"HEAT-GEN-SEFF")<D.Value("REC DEF7"))
{
          //trigger recommendations EPCR1 (GSHP) and EPCR5 (ASHP)
          EPCR1=1;  //install ground source heat pump (GSHP)
          EPCR5=1;  //install air source heat pump (ASHP)
        }
      }

     //--------- heating controls recommendations -------

    //no  central  time  control  EPC-H2  (and  no  local  time  control  or
```

```
optimum start/stop either)
      if (P.Vcodid(vHVAC[i].Id,"CEN-TIME-CTRL")==2) {
         if (P.Vcodid(vHVAC[i].Id,"ROOM-TIME-CTRL")==2) {
            if (P.Vcodid(vHVAC[i].Id,"OPTIMUM-START")==2)
               EPCH2=1;   //central time control
         }
      }
      if (P.Vcodid(vHVAC[i].Id,"ROOM-TIME-CTRL")==2)
         EPCH5=1;     //room by room time control
      if (P.Vcodid(vHVAC[i].Id,"ROOM-TEMP-CTRL")==2)
         EPCH6=1;     //room by room temp control
      if (P.Vcodid(vHVAC[i].Id,"OPTIMUM-START")==2)
         EPCH7=1;      //optimum start
      if (P.Vcodid(vHVAC[i].Id,"WEATHER-COMP")==2)
         EPCH8=1;       //weather compensation

      //if using default heat generator efficiency, trigger EPCH4:
improve on default efficiency
      if (P.Vcodid(vHVAC[i].Id,"EFF-DEFAULT-HEAT")==1)
         EPCH4=1;       // improve on default efficiency


      //-----fuel switching recommendations-----
        //(for heating (section 4 in RH document))
      switch(P.Vcodid(vHVAC[i].Id,"FUEL-TYPE")) {
        case 5: {
          //coal
          //trigger recommendations EPCF2, EPCF3, EPCF6
          EPCF2=1;  //Coal to natural gas
          EPCF3=1;  //Coal to biomass
          EPCF6=1;  //Coal to oil
        }
        break;
        case 4: {
          //oil
          //trigger recommendations EPCF1, EPCF4
          EPCF1=1;  //Oil or LPG to natural gas
          EPCF4=1;  //Oil or LPG to biomass
          //temporary trigger for recommendation EPCH3 - confirmed
when efficiency is fair or poor
          EPCH3 temp=1;
        }
        break;
        case 2: {
          //LPG
          //trigger recommendations EPCF1, EPCF4
          EPCF1=1;  //Oil or LPG to natural gas
          EPCF4=1;  //Oil or LPG to biomass
          //temporary trigger for recommendation EPCH3 - confirmed
when efficiency is fair or poor
          EPCH3_temp=1;
        }
        break;
        case 1: {
          //gas
          //trigger recommendations EPCF5
          EPCF5=1;  //gas to biomass
          //temporary trigger for recommendation EPCH3 - confirmed
when efficiency is fair or poor
          EPCH3_temp=1;
        }
        break;
      } //end of switch
    }  //end of check whether HVAC system provides heating
  }  //end of loop through HVAC systems for heating



  //divide by total energy
```

```
  if (mQ HEAT TOTAL!=0) {
     // Average Fuel Payback for heating of the whole building
weighted by energy of all HVAC systems
     Fuel Payback/=mQ HEAT TOTAL;
     // Average energy performance for heating of the whole building
weighted by energy of all HVAC systems
     Weighted Energy/=mQ HEAT TOTAL;
  }
  else {
    Fuel Payback=0.0;
    Weighted_Energy=0.0;
  }

  //  divide by total CO2
  if (mKG CO2 HEAT TOTAL!=0) {
     // Average CO2 performance for heating of the whole building
weighted by CO2 of all HVAC systems
     Weighted CO2/=mKG CO2 HEAT TOTAL;
  }
  else {
    Weighted CO2=0.0;
  }


 {
  ostringstream res; res << Fuel Payback;
  P.ModKey(RP.Id,"F-PAYBACK-HEAT",res.str());
 }

  //------HVAC heating energy performance - whole building GOOD/
FAIR/ POOR------

  string PERFORMANCE HEAT ENERGY="NOT APPLICABLE";

  //For whole building - is HVAC HEATING ENERGY USE is classified as
GOOD, FAIR or POOR
  //    this decides whether you need to trigger recommendation
associated with FAIR or POOR.

  if (Weighted Energy>0.0) {
      if (Weighted Energy>D.Value("REC DEF3")) {
      //if result is (>x) building end use is GOOD
        PERFORMANCE HEAT ENERGY="GOOD";
      }
      else {    // i.e., will be either FAIR or POOR
           if (EPCH3 temp)     // if it was temporarily triggered
because fuel is oil, gas, or LPG
              EPCH3=1;                    //confirm   recommendation
EPCH3(condensing boiler) as it is now applicable (Fair or Poor
performance)
           //If  x  >=  average  efficiency  >  y,  classify  energy
performance as "fair"
           if (Weighted Energy>D.Value("REC DEF4")) {
              PERFORMANCE_HEAT_ENERGY="FAIR";
           }
           else {    // i.e., average efficiency =< y, building end
use is POOR
                PERFORMANCE HEAT ENERGY="POOR";
                EPCH1=1;     // higher efficiency boiler
           }
      }
  }

  //this decides whether recommendation is triggered and if so
whether you need to
  //trigger recommendation associated with FAIR or (FAIR and) POOR.


  //------CO2 performance - whole building GOOD/ FAIR/ POOR------
```

```
  string PERFORMANCE HEAT CO2="NOT APPLICABLE";

  //For whole building - is HVAC HEATING CO2 classified as GOOD, FAIR
or POOR
  //    this  decides  whether  you  need  to  trigger  recommendation
associated with FAIR or POOR.

  if (Weighted CO2>0.0) {
      if (Weighted CO2>D.Value("REC DEF3")) {
      //if averaged efficiency is (>x) building end use is GOOD
        PERFORMANCE HEAT CO2="GOOD";
      }
      else {    // i.e., will be either FAIR or POOR
           if  (EPCH3 temp)     //  if  it  was  temporarily  triggered
because fuel is oil, gas, or LPG
              EPCH3=1;                      //confirm   recommendation
EPCH3(condensing boiler) as it is now applicable
           if (Weighted CO2>D.Value("REC DEF4")) {
           //If  x  >=  average  efficiency  >  y,  classify  energy
performance as "fair"
              PERFORMANCE HEAT CO2="FAIR";
           }
           else { // i.e., average efficiency =< y, building end use
is POOR
                PERFORMANCE HEAT CO2="POOR";
                EPCH1=1;     // higher efficiency boiler
           }
      }
  }


//write hvac heating generator efficiency keywords
//PERF-DHW-ENERGY and PERF-DHW-CO2 Codes: 1 - GOOD, 2 - FAIR, 3 -
POOR, 4 - NOT APPLICABLE
 {
  ostringstream res; res << PERFORMANCE HEAT ENERGY;
  P.ModKey(RP.Id,"PERF-HEAT-ENERGY",res.str());          //  Averaged
system energy efficiency for heating
 }
 {
  ostringstream res; res << PERFORMANCE_HEAT_CO2;
  P.ModKey(RP.Id,"PERF-HEAT-CO2",res.str());       // Averaged system
CO2 efficiency for heating
 }

  //---- recommendation EPC-H3 ----

  //condensing boiler

  if (EPCH3) {

    //assess impact of recommendation EPCH3 on CO2 and Energy

    //variable to store price weighted energy and CO2 for all HVACs
heating after recommendation is applied where applicable
    mHVAC_HEAT_ENERGY_WT_PW=0.0;
    mHVAC HEAT CO2 WEIGHTED=0.0;

    //assess impact of recommendation EPCH3 on energy and CO2
    //go through all HVAC heat generators in project
    //if  DHW  energy  is  provided  by  HVAC  generator,  include  this
energy in HVAC assessment
    for (int i=0;i<vHVAC.size();i++) {
        //carbon intensity of fuel
        if((P.Vcodid(cId,"BR-TYPE")==5)     ||     (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL_CARBON_INTENSITY=T.Value("tKGCO2_IR",1,P.Vcodid(vHVAC[i].Id,"FU
```

```
EL-TYPE"));
        else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE"));
        //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
        //used in calculation of energy impact

Fuel Price Weighting=T.Value("tFuel Price Weighting",1,P.Vcodid(vHVAC
[i].Id,"FUEL-TYPE"));
        //primary energy used for space heating:
        mQ HEAT THIS HVAC PW=P.Vnum(vHVAC[i].Id,"Q-PRIM-
HEAT")*Fuel_Price_Weighting;
        mKG CO2 HEAT THIS HVAC=(P.Vnum(vHVAC[i].Id,"KWH-
HEAT")*mFUEL CARBON INTENSITY);
        deque<int>  mDHW_SERVED_REF=P.Vlrefid(vHVAC[i].Id,"DHW-SERVED-
REF");

        //energy for DHW:
        mQ DHW THIS HVAC PW=0.0;
        mKG CO2 DHW THIS HVAC=0.0;
        for (int j=0;j< mDHW SERVED REF.size();j++) {
            mQ DHW THIS HVAC PW+=P.Vnum(mDHW SERVED REF[j],"Q-
PRIM")*Fuel Price Weighting;
            mKG CO2 DHW THIS HVAC+=P.Vnum(mDHW SERVED REF[j],"KWH-
DHW")*mFUEL CARBON INTENSITY;
        }
        //if eff <= D.Value("REC DEF3") and fuel is gas, oil, or LPG,
correct  energy  according  to  recommendation  (set  efficiency  to
D.Value("REC DEF20"))
        if                                  (((P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE")==D.Value("dGAS CODID"))     ||     (P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE")==D.Value("dLPG CODID"))     ||     (P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE")==D.Value("dOIL CODID")))&&(P.Vnum(vHVAC[i].Id,"HEAT-GEN-
SEFF")<=D.Value("REC DEF3"))) {
           if ((D.Value("REC DEF20"))!=0) {
             //weight         energy         =>         energy*(actual
eff/D.Value("REC_DEF20")) (energy is already price weighted)

mHVAC HEAT ENERGY WT PW+=(mQ HEAT THIS HVAC PW+mQ DHW THIS HVAC PW)*(
P.Vnum(vHVAC[i].Id,"HEAT-GEN-SEFF")/D.Value("REC_DEF20"));
               //weight CO2 => CO2*(actual eff/D.Value("REC DEF20"))

mHVAC_HEAT_CO2_WEIGHTED+=(mKG_CO2_HEAT_THIS_HVAC+mKG_CO2_DHW_THIS_HVA
C)*(P.Vnum(vHVAC[i].Id,"HEAT-GEN-SEFF")/D.Value("REC DEF20"));
           }
        }
        else {     // efficiency is good and fuel isn't gas, oil, or
LPG
            //if eff > D.Value("REC_DEF3") add energy directly

mHVAC HEAT ENERGY WT PW+=(mQ HEAT THIS HVAC PW+mQ DHW THIS HVAC PW);
            //if eff > D.Value("REC_DEF3") add CO2 directly

mHVAC HEAT CO2 WEIGHTED+=(mKG CO2 HEAT THIS HVAC+mKG CO2 DHW THIS HVA
C);
        }
    }

    //variable for adjusted Total Building energy
    mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mQ_HEAT_TOTAL_PRICE_WEIGHTED+mHVAC_HEAT_ENERGY_WT_PW;
    mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
mKG CO2 HEAT TOTAL+mHVAC HEAT CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ_TOTAL_ADJUSTED);
```

```
    IMPACT CO2=BOrec.fIMPACT CO2(D,                    mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON_SAVING_POUND)
    //Multiply default payback by 140 and divide by TYPICAL building
heating consumption (kWh/m2.year)
    double mAREA=P.Vnum(gId,"AREA");
    double     mTYP KWH HEAT=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
HEAT")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")));
    if((mAREA!=0)&&(mTYP_KWH_HEAT!=0))
      PAYBACK=D.Value("dEPC H3")*140.0/(mTYP KWH HEAT/mAREA);
    else
      PAYBACK=0.0;

    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON_SAVING_POUND=BOrec.fCARBON_SAVING_POUND(D,CARBON_SAVING_NO);

    string EPC XX="EPC-H3";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
  }

  //---- END recommendation EPC-H3 ----


  //---- recommendation EPC-H1 ----

  //high efficiency boiler

  //if either energy or CO2 are fair, trigger recommendation EPCH1
(1=GOOD/2=FAIR/3=POOR/4=N/A)
  if (EPCH1) {
    //assess impact of recommendation EPCH1 on CO2 and Energy

    //variable to store price weighted energy and CO2 for all HVACs
heating after recommendation is applied where applicable
    mHVAC HEAT ENERGY WT PW=0.0;
    mHVAC_HEAT_CO2_WEIGHTED=0.0;

    //assess impact of recommendation EPCH1 on energy and CO2
    //go through all HVAC heat generators in project
    //if  DHW  energy  is  provided  by  HVAC  generator,  include  this
energy in HVAC assessment
    for (int i=0;i<vHVAC.size();i++) {
      //carbon intensity of fuel
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,P.Vcodid(vHVAC[i].Id,"FU
EL-TYPE"));
      else //  all UK

mFUEL_CARBON_INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE"));
      //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
      //used in calculation of energy impact

Fuel_Price_Weighting=T.Value("tFuel_Price_Weighting",1,P.Vcodid(vHVAC
[i].Id,"FUEL-TYPE"));
      //primary energy used for space heating:
      mQ_HEAT_THIS_HVAC_PW=P.Vnum(vHVAC[i].Id,"Q-PRIM-
HEAT")*Fuel Price Weighting;
      mKG CO2 HEAT THIS HVAC=(P.Vnum(vHVAC[i].Id,"KWH-
HEAT")*mFUEL_CARBON_INTENSITY);
```

```
      deque<int>    mDHW SERVED REF=P.Vlrefid(vHVAC[i].Id,"DHW-SERVED-
REF");

      //energy for DHW:
      mQ_DHW_THIS_HVAC_PW=0.0;
      mKG CO2 DHW THIS HVAC=0.0;
      for (int j=0;j< mDHW SERVED REF.size();j++) {
        mQ_DHW_THIS_HVAC_PW+=P.Vnum(mDHW_SERVED_REF[j],"Q-
PRIM")*Fuel Price Weighting;
        mKG CO2 DHW THIS HVAC+=P.Vnum(mDHW SERVED REF[j],"KWH-
DHW")*mFUEL_CARBON_INTENSITY;
      }

      if (P.Vnum(vHVAC[i].Id,"HEAT-GEN-SEFF")<=D.Value("REC_DEF4")) {
        //if eff <= D.Value("REC DEF4") correct energy according to
recommendation (set efficiency to D.Value("REC DEF15"))
        if ((D.Value("REC_DEF15"))!=0) {
          //weight energy => energy*(actual eff/D.Value("REC DEF15"))
(energy is already price weighted)

mHVAC HEAT ENERGY WT PW+=(mQ HEAT THIS HVAC PW+mQ DHW THIS HVAC PW)*(
P.Vnum(vHVAC[i].Id,"HEAT-GEN-SEFF")/D.Value("REC DEF15"));
          //weight CO2 => CO2*(actual eff/D.Value("REC DEF15"))

mHVAC HEAT CO2 WEIGHTED+=(mKG CO2 HEAT THIS HVAC+mKG CO2 DHW THIS HVA
C)*(P.Vnum(vHVAC[i].Id,"HEAT-GEN-SEFF")/D.Value("REC DEF15"));
        }
      }
      else {
        //if eff >= D.Value("REC DEF15") add energy directly

mHVAC HEAT ENERGY WT PW+=(mQ HEAT THIS HVAC PW+mQ DHW THIS HVAC PW);
        //if eff >= D.Value("REC DEF15") add CO2 directly

mHVAC HEAT CO2 WEIGHTED+=(mKG CO2 HEAT THIS HVAC+mKG CO2 DHW THIS HVA
C);
      }
    }

    //variable for adjusted Total Building energy
    mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mQ_HEAT_TOTAL_PRICE_WEIGHTED+mHVAC_HEAT_ENERGY_WT_PW;
    mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
mKG CO2 HEAT TOTAL+mHVAC HEAT CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ_TOTAL_ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D,                   mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
    //Multiply default payback by 140 and divide by TYPICAL building
heating consumption (kWh/m2.year)
    double mAREA=P.Vnum(gId,"AREA");
    double     mTYP_KWH_HEAT=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
HEAT")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")));
    if((mAREA!=0)&&(mTYP KWH HEAT!=0))
      PAYBACK=D.Value("dEPC_H1")*140.0/(mTYP_KWH_HEAT/mAREA);
    else
      PAYBACK=0.0;

    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON_SAVING_POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);
```

```
    string EPC XX="EPC-H1";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
  }

  //---- END recommendation EPC-H1 ----



  //---- recommendation EPC-H4 ----

  //default efficiency

  if (EPCH4) {
    //assess impact of recommendation EPCH4 on CO2 and Energy

    //variable to store price weighted energy and CO2 for all HVACs
heating after recommendation is applied where applicable
    mHVAC HEAT ENERGY WT PW=0.0;
    mHVAC_HEAT_CO2_WEIGHTED=0.0;

    //assess impact of recommendation EPCH4 on energy and CO2
    //go through all HVAC heat generators in project
    //if  DHW  energy  is  provided  by  HVAC  generator,  include  this
energy in HVAC assessment
    for (int i=0;i<vHVAC.size();i++) {
      //carbon intensity of fuel
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,P.Vcodid(vHVAC[i].Id,"FU
EL-TYPE"));
      else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE"));
      //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
      //used in calculation of energy impact

Fuel Price Weighting=T.Value("tFuel Price Weighting",1,P.Vcodid(vHVAC
[i].Id,"FUEL-TYPE"));
      //primary energy used for space heating:
      mQ HEAT THIS HVAC PW=P.Vnum(vHVAC[i].Id,"Q-PRIM-
HEAT")*Fuel_Price_Weighting;
      mKG CO2 HEAT THIS HVAC=(P.Vnum(vHVAC[i].Id,"KWH-
HEAT")*mFUEL CARBON INTENSITY);
      deque<int>   mDHW_SERVED_REF=P.Vlrefid(vHVAC[i].Id,"DHW-SERVED-
REF");

      //energy for DHW:
      mQ DHW THIS HVAC PW=0.0;
      mKG CO2 DHW THIS HVAC=0.0;
      for (int j=0;j< mDHW_SERVED_REF.size();j++) {
        mQ DHW THIS HVAC PW+=P.Vnum(mDHW SERVED REF[j],"Q-
PRIM")*Fuel Price Weighting;
        mKG_CO2_DHW_THIS_HVAC+=P.Vnum(mDHW_SERVED_REF[j],"KWH-
DHW")*mFUEL CARBON INTENSITY;
      }
      // if the default efficiency is used
      if (P.Vcodid(vHVAC[i].Id,"EFF-DEFAULT-HEAT")==1) {
        if ((D.Value("REC DEF15"))!=0) {
          //weight energy => energy*(actual eff/D.Value("REC DEF15"))
(energy is already price weighted)

mHVAC_HEAT_ENERGY_WT_PW+=(mQ_HEAT_THIS_HVAC_PW+mQ_DHW_THIS_HVAC_PW)*(
P.Vnum(vHVAC[i].Id,"HEAT-GEN-SEFF")/D.Value("REC DEF15"));
          //weight CO2 => CO2*(actual eff/D.Value("REC DEF15"))
```

```
mHVAC HEAT CO2 WEIGHTED+=(mKG CO2 HEAT THIS HVAC+mKG CO2 DHW THIS HVA
C)*(P.Vnum(vHVAC[i].Id,"HEAT-GEN-SEFF")/D.Value("REC DEF15"));
        }
      }
      else {   // default efficency is NOT used
        //if eff >= D.Value("REC DEF15") add energy directly

mHVAC_HEAT_ENERGY_WT_PW+=(mQ_HEAT_THIS_HVAC_PW+mQ_DHW_THIS_HVAC_PW);
        //if eff >= D.Value("REC DEF15") add CO2 directly

mHVAC_HEAT_CO2_WEIGHTED+=(mKG_CO2_HEAT_THIS_HVAC+mKG_CO2_DHW_THIS_HVA
C);
      }
    }

    //variable for adjusted Total Building energy
    mQ_TOTAL_ADJUSTED=mQ_TOTAL_PRICE_WEIGHTED-
mQ HEAT TOTAL PRICE WEIGHTED+mHVAC HEAT ENERGY WT PW;
    mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
mKG_CO2_HEAT_TOTAL+mHVAC_HEAT_CO2_WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D,                   mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
    //Multiply default payback by 140 and divide by TYPICAL building
heating consumption (kWh/m2.year)
    double mAREA=P.Vnum(gId,"AREA");
    double     mTYP KWH HEAT=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
HEAT")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")));
    if((mAREA!=0)&&(mTYP KWH HEAT!=0))
      PAYBACK=D.Value("dEPC H4")*140.0/(mTYP KWH HEAT/mAREA);
    else
      PAYBACK=0.0;

    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON_SAVING_POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    string EPC XX="EPC-H4";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON_SAVING_POUND);
  }

  //---- END recommendation EPC-H4 ----



  //---- recommendation EPC-R5 ----

  //consider ASHP

  if (EPCR5) {
    //assess impact of recommendation EPCR5 on CO2 and Energy

    //variable to store price weighted energy and CO2 for all HVACs
heating after recommendation is applied where applicable
    mHVAC HEAT ENERGY WT PW=0.0;
    mHVAC HEAT CO2 WEIGHTED=0.0;

    //assess impact of recommendation EPCR5 on energy and CO2
    //go through all HVAC heat generators in project
    //if  DHW  energy  is  provided  by  HVAC  generator,  include  this
```

```
energy in HVAC assessment
    for (int i=0;i<vHVAC.size();i++) {
      //carbon intensity of fuel
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,P.Vcodid(vHVAC[i].Id,"FU
EL-TYPE"));
      else // all UK

mFUEL_CARBON_INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE"));
      //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
      //used in calculation of energy impact

Fuel_Price_Weighting=T.Value("tFuel_Price_Weighting",1,P.Vcodid(vHVAC
[i].Id,"FUEL-TYPE"));

      //primary energy used for space heating:
      mQ HEAT THIS HVAC PW=P.Vnum(vHVAC[i].Id,"Q-PRIM-
HEAT")*Fuel Price Weighting;
      mKG CO2 HEAT THIS HVAC=(P.Vnum(vHVAC[i].Id,"KWH-
HEAT")*mFUEL CARBON INTENSITY);
      deque<int>    mDHW SERVED REF=P.Vlrefid(vHVAC[i].Id,"DHW-SERVED-
REF");

      //energy for DHW:
      mQ DHW THIS HVAC PW=0.0;
      mKG CO2 DHW THIS HVAC=0.0;
      for (int j=0;j< mDHW SERVED REF.size();j++) {
        mQ DHW THIS HVAC+=P.Vnum(mDHW SERVED REF[j],"Q-
PRIM")*Fuel Price Weighting;
        mKG CO2 DHW THIS HVAC+=P.Vnum(mDHW SERVED REF[j],"KWH-
DHW")*mFUEL CARBON INTENSITY;
      }

      if                                ((P.Vnum(vHVAC[i].Id,"HEAT-GEN-
SEFF")<D.Value("REC_DEF7"))&&(P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE")==(int)D.Value("dELEC CODID"))) {
        //if  eff  <  D.Value("REC DEF7")  correct  energy  according  to
recommendation (set efficiency to D.Value("REC_DEF17"))
        //weight energy => energy/D.Value("REC DEF17")old eff
        if ((D.Value("REC DEF17"))!=0) {
          //weight energy => energy*(actual eff/D.Value("REC_DEF17"))

mHVAC HEAT ENERGY WT PW+=(mQ HEAT THIS HVAC PW+mQ DHW THIS HVAC PW)*(
P.Vnum(vHVAC[i].Id,"HEAT-GEN-SEFF")/D.Value("REC_DEF17"));
          //weight CO2 => CO2*(actual eff/D.Value("REC DEF17"))

mHVAC_HEAT_CO2_WEIGHTED+=(mKG_CO2_HEAT_THIS_HVAC+mKG_CO2_DHW_THIS_HVA
C)*(P.Vnum(vHVAC[i].Id,"HEAT-GEN-SEFF")/D.Value("REC DEF17"));
        }
      }
      else {
        // add energy directly

mHVAC HEAT ENERGY WT PW+=(mQ HEAT THIS HVAC PW+mQ DHW THIS HVAC PW);
        //add CO2 directly

mHVAC HEAT CO2 WEIGHTED+=(mKG CO2 HEAT THIS HVAC+mKG CO2 DHW THIS HVA
C);
      }
    }

    //variable for adjusted Total Building energy
    mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mQ HEAT TOTAL PRICE WEIGHTED+mHVAC HEAT ENERGY WT PW;
    mCO2_TOTAL_ADJUSTED=mKG_CO2_PROJECT-
```

```
mKG CO2 HEAT TOTAL+mHVAC HEAT CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ_TOTAL_ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D,                  mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
    //Multiply default payback by 140 and divide by TYPICAL building
heating consumption (kWh/m2.year)
    double mAREA=P.Vnum(gId,"AREA");
    double     mTYP_KWH_HEAT=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
HEAT")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")));
    if((mAREA!=0)&&(mTYP KWH HEAT!=0))
      PAYBACK=D.Value("dEPC_R5")*140.0/(mTYP_KWH_HEAT/mAREA);
    else
      PAYBACK=0.0;

    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    //write the EPCR5 recommendation keyword
    string EPC XX="EPC-R5";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
  }

  //---- END recommendation EPC-R5 ----


  //---- recommendation EPC-R1 ----

  //consider GSHP

  if (EPCR1) {
    //assess impact of recommendation EPCR1 on CO2 and Energy

    //variable to store price weighted energy and CO2 for all HVACs
heating after recommendation is applied where applicable
    mHVAC_HEAT_ENERGY_WT_PW=0.0;
    mHVAC HEAT CO2 WEIGHTED=0.0;

    //assess impact of recommendation EPCR1 on energy and CO2
    //go through all HVAC heat generators in project
    //if  DHW  energy  is  provided  by  HVAC  generator,  include  this
energy in HVAC assessment
    for (int i=0;i<vHVAC.size();i++) {
      //carbon intensity of fuel
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL_CARBON_INTENSITY=T.Value("tKGCO2_IR",1,P.Vcodid(vHVAC[i].Id,"FU
EL-TYPE"));
      else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE"));
      //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
      //used in calculation of energy impact

Fuel Price Weighting=T.Value("tFuel Price Weighting",1,P.Vcodid(vHVAC
[i].Id,"FUEL-TYPE"));
```

```
      //primary energy used for space heating:
      mQ HEAT THIS HVAC PW=P.Vnum(vHVAC[i].Id,"Q-PRIM-
HEAT")*Fuel Price Weighting;
      mKG CO2 HEAT THIS HVAC=(P.Vnum(vHVAC[i].Id,"KWH-
HEAT")*mFUEL_CARBON_INTENSITY);
      deque<int>    mDHW SERVED REF=P.Vlrefid(vHVAC[i].Id,"DHW-SERVED-
REF");

      //energy for DHW:
      mQ DHW THIS HVAC PW=0.0;
      mKG_CO2_DHW_THIS_HVAC=0.0;
      for (int j=0;j< mDHW SERVED REF.size();j++) {
        mQ DHW THIS HVAC PW+=P.Vnum(mDHW SERVED REF[j],"Q-
PRIM")*Fuel_Price_Weighting;
        mKG CO2 DHW THIS HVAC+=P.Vnum(mDHW SERVED REF[j],"KWH-
DHW")*mFUEL CARBON INTENSITY;
      }

      if                                ((P.Vnum(vHVAC[i].Id,"HEAT-GEN-
SEFF")<D.Value("REC_DEF7"))&&(P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE")==(int)D.Value("dELEC CODID"))) {
        //if  eff  <  D.Value("REC DEF7")  correct  energy  according  to
recommendation (set efficiency to D.Value("REC DEF16"))
        //weight energy => energy*(actual eff/D.Value("REC DEF16"))
        if ((D.Value("REC DEF16"))!=0) {
          //weight energy => energy(actual eff/D.Value("REC DEF16"))

mHVAC HEAT ENERGY WT PW+=(mQ HEAT THIS HVAC PW+mQ DHW THIS HVAC PW)*(
P.Vnum(vHVAC[i].Id,"HEAT-GEN-SEFF")/D.Value("REC DEF16"));
          //weight CO2 => CO2/D.Value("REC DEF16")*actual eff

mHVAC HEAT CO2 WEIGHTED+=(mKG CO2 HEAT THIS HVAC+mKG CO2 DHW THIS HVA
C)*(P.Vnum(vHVAC[i].Id,"HEAT-GEN-SEFF")/D.Value("REC DEF16"));
        }
      }
      else {
        // add energy directly

mHVAC_HEAT_ENERGY_WT_PW+=(mQ_HEAT_THIS_HVAC_PW+mQ_DHW_THIS_HVAC_PW);
        //add CO2 directly

mHVAC_HEAT_CO2_WEIGHTED+=(mKG_CO2_HEAT_THIS_HVAC+mKG_CO2_DHW_THIS_HVA
C);
      }
    }

    //variable for adjusted Total Building energy
    mQ_TOTAL_ADJUSTED=mQ_TOTAL_PRICE_WEIGHTED-
mQ HEAT TOTAL PRICE WEIGHTED+mHVAC HEAT ENERGY WT PW;
   //variable for adjusted Total Building CO2
    mCO2_TOTAL_ADJUSTED=mKG_CO2_PROJECT-
mKG CO2 HEAT TOTAL+mHVAC HEAT CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D,                    mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

    //calculate  payback  (PAYBACK)  and  Carbon  saving  per  pound
(CARBON SAVING POUND)
    //Multiply default payback by 140 and divide by TYPICAL building
heating consumption (kWh/m2.year)
    double mAREA=P.Vnum(gId,"AREA");
    double     mTYP KWH HEAT=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
HEAT")*(2.0/T.Value("t2010_TYP",1,P.Vcodid(cId,"EPC-TYPE")));
    if((mAREA!=0)&&(mTYP KWH HEAT!=0))
      PAYBACK=D.Value("dEPC R1")*140.0/(mTYP KWH HEAT/mAREA);
    else
```

```
      PAYBACK=0.0;

    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON_SAVING_POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    //write the EPCR1 recommendation keyword
    string EPC XX="EPC-R1";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
  }

  //---- END recommendation EPC-R1 ----



  //---- recommendation EPC-H2 ----

  //centralised time control

  if (EPCH2) {
    //assess impact of recommendation EPCH2 on CO2 and Energy

    //variable to store price weighted energy and CO2 for all HVACs
heating after recommendation is applied where applicable
    mHVAC HEAT ENERGY WT PW=0.0;
    mHVAC HEAT CO2 WEIGHTED=0.0;

    //assess impact of recommendation EPCH2 on energy and CO2
    //go through all HVAC heat generators in project
    //if  DHW  energy  is  provided  by  HVAC  generator,  include  this
energy in HVAC assessment
    for (int i=0;i<vHVAC.size();i++) {
      //carbon intensity of fuel
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,P.Vcodid(vHVAC[i].Id,"FU
EL-TYPE"));
      else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE"));
      //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
      //used in calculation of energy impact

Fuel Price Weighting=T.Value("tFuel Price Weighting",1,P.Vcodid(vHVAC
[i].Id,"FUEL-TYPE"));
      //primary energy used for space heating:
      mQ HEAT THIS HVAC PW=P.Vnum(vHVAC[i].Id,"Q-PRIM-
HEAT")*Fuel_Price_Weighting;
      mKG CO2 HEAT THIS HVAC=(P.Vnum(vHVAC[i].Id,"KWH-
HEAT")*mFUEL CARBON INTENSITY);
      deque<int>    mDHW_SERVED_REF=P.Vlrefid(vHVAC[i].Id,"DHW-SERVED-
REF");

      //energy for DHW:
      mQ DHW THIS HVAC PW=0.0;
      mKG CO2 DHW THIS HVAC=0.0;
      for (int j=0;j< mDHW_SERVED_REF.size();j++) {
        mQ DHW THIS HVAC PW+=P.Vnum(mDHW SERVED REF[j],"Q-
PRIM")*Fuel Price Weighting;
        mKG_CO2_DHW_THIS_HVAC+=P.Vnum(mDHW_SERVED_REF[j],"KWH-
DHW")*mFUEL CARBON INTENSITY;
      }
      // if there is no central time control
```

```
      if (P.Vcodid(vHVAC[i].Id,"CEN-TIME-CTRL")==2) {
        if ((D.Value("REC DEF21"))!=0) {
          //weight     energy     =>     energy*(actual     eff/(actual
eff*D.Value("REC DEF21"))) (energy is already price weighted)

mHVAC HEAT ENERGY WT PW+=(mQ HEAT THIS HVAC PW+mQ DHW THIS HVAC PW)/D
.Value("REC DEF21");
          //weight      CO2      =>      CO2*(actual      eff/(actual
eff*D.Value("REC DEF21")))

mHVAC_HEAT_CO2_WEIGHTED+=(mKG_CO2_HEAT_THIS_HVAC+mKG_CO2_DHW_THIS_HVA
C)/D.Value("REC DEF21");
        }
      }
      else {
        //if there is central time control add energy directly

mHVAC HEAT ENERGY WT PW+=(mQ HEAT THIS HVAC PW+mQ DHW THIS HVAC PW);
        //if there is central time control add CO2 directly

mHVAC HEAT CO2 WEIGHTED+=(mKG CO2 HEAT THIS HVAC+mKG CO2 DHW THIS HVA
C);
      }
    }

    //variable for adjusted Total Building energy
    mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mQ HEAT TOTAL PRICE WEIGHTED+mHVAC HEAT ENERGY WT PW;
    mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
mKG CO2 HEAT TOTAL+mHVAC HEAT CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D,                    mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON_SAVING_POUND)
    //Multiply default payback by 140 and divide by TYPICAL building
heating consumption (kWh/m2.year)
    double mAREA=P.Vnum(gId,"AREA");
    double     mTYP KWH HEAT=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
HEAT")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")));
    if((mAREA!=0)&&(mTYP_KWH_HEAT!=0))
      PAYBACK=D.Value("dEPC H2")*140.0/(mTYP KWH HEAT/mAREA);
    else
      PAYBACK=0.0;

    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON_SAVING_POUND=BOrec.fCARBON_SAVING_POUND(D,CARBON_SAVING_NO);

    string EPC XX="EPC-H2";
    Write_Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
  }

  //---- END recommendation EPC-H2 ----


  //---- recommendation EPC-H5 ----

  //local time control

  if (EPCH5) {
    //assess impact of recommendation EPCH2 on CO2 and Energy
```

```
    //variable to store price weighted energy and CO2 for all HVACs
heating after recommendation is applied where applicable
    mHVAC HEAT ENERGY WT PW=0.0;
    mHVAC_HEAT_CO2_WEIGHTED=0.0;

    //assess impact of recommendation EPCH2 on energy and CO2
    //go through all HVAC heat generators in project
    //if DHW energy is provided by HVAC generator, include this
energy in HVAC assessment
    for (int i=0;i<vHVAC.size();i++) {
      //carbon intensity of fuel
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,P.Vcodid(vHVAC[i].Id,"FU
EL-TYPE"));
      else // all UK

mFUEL_CARBON_INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE"));
      //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
      //used in calculation of energy impact

Fuel Price Weighting=T.Value("tFuel Price Weighting",1,P.Vcodid(vHVAC
[i].Id,"FUEL-TYPE"));
      //primary energy used for space heating:
      mQ HEAT THIS HVAC PW=P.Vnum(vHVAC[i].Id,"Q-PRIM-
HEAT")*Fuel Price Weighting;
      mKG CO2 HEAT THIS HVAC=(P.Vnum(vHVAC[i].Id,"KWH-
HEAT")*mFUEL CARBON INTENSITY);
      deque<int>    mDHW SERVED REF=P.Vlrefid(vHVAC[i].Id,"DHW-SERVED-
REF");

      //energy for DHW:
      mQ DHW THIS HVAC PW=0.0;
      mKG CO2 DHW THIS HVAC=0.0;
      for (int j=0;j< mDHW_SERVED_REF.size();j++) {
        mQ DHW THIS HVAC PW+=P.Vnum(mDHW SERVED REF[j],"Q-
PRIM")*Fuel Price Weighting;
        mKG_CO2_DHW_THIS_HVAC+=P.Vnum(mDHW_SERVED_REF[j],"KWH-
DHW")*mFUEL CARBON INTENSITY;
      }
      // if there is no local time control
      if (P.Vcodid(vHVAC[i].Id,"ROOM-TIME-CTRL")==2) {
        if ((D.Value("REC DEF21"))!=0) {
          //weight     energy     =>     energy*(actual     eff/(actual
eff*D.Value("REC DEF21"))) (energy is already price weighted)

mHVAC_HEAT_ENERGY_WT_PW+=(mQ_HEAT_THIS_HVAC_PW+mQ_DHW_THIS_HVAC_PW)/D
.Value("REC DEF21");
          //weight      CO2      =>      CO2*(actual      eff/(actual
eff*D.Value("REC_DEF21")))

mHVAC HEAT CO2 WEIGHTED+=(mKG CO2 HEAT THIS HVAC+mKG CO2 DHW THIS HVA
C)/D.Value("REC_DEF21");
        }
      }
      else {
        //if there is local time control add energy directly

mHVAC_HEAT_ENERGY_WT_PW+=(mQ_HEAT_THIS_HVAC_PW+mQ_DHW_THIS_HVAC_PW);
        //if there is local time control add CO2 directly

mHVAC_HEAT_CO2_WEIGHTED+=(mKG_CO2_HEAT_THIS_HVAC+mKG_CO2_DHW_THIS_HVA
C);
      }
    }
```

```
    //variable for adjusted Total Building energy
    mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mQ HEAT TOTAL PRICE WEIGHTED+mHVAC HEAT ENERGY WT PW;
    mCO2_TOTAL_ADJUSTED=mKG_CO2_PROJECT-
mKG CO2 HEAT TOTAL+mHVAC HEAT CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D,                   mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
    //Multiply default payback by 140 and divide by TYPICAL building
heating consumption (kWh/m2.year)
    double mAREA=P.Vnum(gId,"AREA");
    double     mTYP KWH HEAT=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
HEAT")*(2.0/T.Value("t2010_TYP",1,P.Vcodid(cId,"EPC-TYPE")));
    if((mAREA!=0)&&(mTYP KWH HEAT!=0))
      PAYBACK=D.Value("dEPC H5")*140.0/(mTYP KWH HEAT/mAREA);
    else
      PAYBACK=0.0;

    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    string EPC XX="EPC-H5";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
  }

  //---- END recommendation EPC-H5 ----


  //---- recommendation EPC-H6 ----

  //local temp control

  if (EPCH6) {
    //assess impact of recommendation EPCH2 on CO2 and Energy

    //variable to store price weighted energy and CO2 for all HVACs
heating after recommendation is applied where applicable
    mHVAC HEAT ENERGY WT PW=0.0;
    mHVAC HEAT CO2 WEIGHTED=0.0;

    //assess impact of recommendation EPCH2 on energy and CO2
    //go through all HVAC heat generators in project
    //if  DHW  energy  is  provided  by  HVAC  generator,  include  this
energy in HVAC assessment
    for (int i=0;i<vHVAC.size();i++) {
      //carbon intensity of fuel
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,P.Vcodid(vHVAC[i].Id,"FU
EL-TYPE"));
      else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE"));
      //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
      //used in calculation of energy impact
```

```
Fuel Price Weighting=T.Value("tFuel Price Weighting",1,P.Vcodid(vHVAC
[i].Id,"FUEL-TYPE"));
      //primary energy used for space heating:
      mQ_HEAT_THIS_HVAC_PW=P.Vnum(vHVAC[i].Id,"Q-PRIM-
HEAT")*Fuel Price Weighting;
      mKG CO2 HEAT THIS HVAC=(P.Vnum(vHVAC[i].Id,"KWH-
HEAT")*mFUEL_CARBON_INTENSITY);
      deque<int>    mDHW SERVED REF=P.Vlrefid(vHVAC[i].Id,"DHW-SERVED-
REF");

      //energy for DHW:
      mQ DHW THIS HVAC PW=0.0;
      mKG_CO2_DHW_THIS_HVAC=0.0;
      for (int j=0;j< mDHW SERVED REF.size();j++) {
        mQ DHW THIS HVAC PW+=P.Vnum(mDHW SERVED REF[j],"Q-
PRIM")*Fuel_Price_Weighting;
        mKG CO2 DHW THIS HVAC+=P.Vnum(mDHW SERVED REF[j],"KWH-
DHW")*mFUEL CARBON INTENSITY;
      }
      // if there is no local temp control
      if (P.Vcodid(vHVAC[i].Id,"ROOM-TEMP-CTRL")==2) {
        if ((D.Value("REC DEF22"))!=0) {
          //weight     energy     =>     energy*(actual     eff/(actual
eff*D.Value("REC DEF22"))) (energy is already price weighted)

mHVAC HEAT ENERGY WT PW+=(mQ HEAT THIS HVAC PW+mQ DHW THIS HVAC PW)/D
.Value("REC DEF22");
          //weight      CO2      =>      CO2*(actual      eff/(actual
eff*D.Value("REC DEF22")))

mHVAC HEAT CO2 WEIGHTED+=(mKG CO2 HEAT THIS HVAC+mKG CO2 DHW THIS HVA
C)/D.Value("REC DEF22");
        }
      }
      else {
        //if there is local temp control add energy directly

mHVAC_HEAT_ENERGY_WT_PW+=(mQ_HEAT_THIS_HVAC_PW+mQ_DHW_THIS_HVAC_PW);
        //if there is local temp control add CO2 directly

mHVAC_HEAT_CO2_WEIGHTED+=(mKG_CO2_HEAT_THIS_HVAC+mKG_CO2_DHW_THIS_HVA
C);
      }
    }

    //variable for adjusted Total Building energy
    mQ_TOTAL_ADJUSTED=mQ_TOTAL_PRICE_WEIGHTED-
mQ HEAT TOTAL PRICE WEIGHTED+mHVAC HEAT ENERGY WT PW;
    mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
mKG_CO2_HEAT_TOTAL+mHVAC_HEAT_CO2_WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D,                    mKG CO2 PROJECT,
mCO2_TOTAL_ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON_SAVING_POUND)
    //Multiply default payback by 140 and divide by TYPICAL building
heating consumption (kWh/m2.year)
    double mAREA=P.Vnum(gId,"AREA");
    double     mTYP KWH HEAT=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
HEAT")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")));
    if((mAREA!=0)&&(mTYP_KWH_HEAT!=0))
      PAYBACK=D.Value("dEPC H6")*140.0/(mTYP KWH HEAT/mAREA);
    else
      PAYBACK=0.0;
```

```
    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    string EPC_XX="EPC-H6";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
  }

  //---- END recommendation EPC-H6 ----


  //---- recommendation EPC-H7 ----

  //optimum start stop control

  if (EPCH7) {
    //assess impact of recommendation EPCH2 on CO2 and Energy

    //variable to store price weighted energy and CO2 for all HVACs
heating after recommendation is applied where applicable
    mHVAC HEAT ENERGY WT PW=0.0;
    mHVAC HEAT CO2 WEIGHTED=0.0;

    //assess impact of recommendation EPCH2 on energy and CO2
    //go through all HVAC heat generators in project
    //if  DHW  energy  is  provided  by  HVAC  generator,  include  this
energy in HVAC assessment
    for (int i=0;i<vHVAC.size();i++) {
      //carbon intensity of fuel
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,P.Vcodid(vHVAC[i].Id,"FU
EL-TYPE"));
      else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE"));
      //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
      //used in calculation of energy impact

Fuel Price Weighting=T.Value("tFuel Price Weighting",1,P.Vcodid(vHVAC
[i].Id,"FUEL-TYPE"));
      //primary energy used for space heating:
      mQ HEAT THIS HVAC PW=P.Vnum(vHVAC[i].Id,"Q-PRIM-
HEAT")*Fuel_Price_Weighting;
      mKG CO2 HEAT THIS HVAC=(P.Vnum(vHVAC[i].Id,"KWH-
HEAT")*mFUEL CARBON INTENSITY);
      deque<int>    mDHW_SERVED_REF=P.Vlrefid(vHVAC[i].Id,"DHW-SERVED-
REF");

      //energy for DHW:
      mQ DHW THIS HVAC PW=0.0;
      mKG CO2 DHW THIS HVAC=0.0;
      for (int j=0;j< mDHW_SERVED_REF.size();j++) {
        mQ DHW THIS HVAC PW+=P.Vnum(mDHW SERVED REF[j],"Q-
PRIM")*Fuel Price Weighting;
        mKG_CO2_DHW_THIS_HVAC+=P.Vnum(mDHW_SERVED_REF[j],"KWH-
DHW")*mFUEL CARBON INTENSITY;
      }
      // if there is optimum start stop control
      if (P.Vcodid(vHVAC[i].Id,"OPTIMUM-START")==2) {
        if ((D.Value("REC DEF22"))!=0) {
          //weight     energy     =>     energy*(actual     eff/(actual
```

```
eff*D.Value("REC DEF22"))) (energy is already price weighted)

mHVAC HEAT ENERGY WT PW+=(mQ HEAT THIS HVAC PW+mQ DHW THIS HVAC PW)/D
.Value("REC DEF22");
          //weight      CO2      =>      CO2*(actual      eff/(actual
eff*D.Value("REC DEF22")))

mHVAC_HEAT_CO2_WEIGHTED+=(mKG_CO2_HEAT_THIS_HVAC+mKG_CO2_DHW_THIS_HVA
C)/D.Value("REC DEF22");
        }
      }
      else {
        //if there is optimum start stop control add energy directly

mHVAC HEAT ENERGY WT PW+=(mQ HEAT THIS HVAC PW+mQ DHW THIS HVAC PW);
        //if there is optimum start stop control add CO2 directly

mHVAC HEAT CO2 WEIGHTED+=(mKG CO2 HEAT THIS HVAC+mKG CO2 DHW THIS HVA
C);
      }
    }

    //variable for adjusted Total Building energy
    mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mQ HEAT TOTAL PRICE WEIGHTED+mHVAC HEAT ENERGY WT PW;
    mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
mKG CO2 HEAT TOTAL+mHVAC HEAT CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D,                   mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

    //calculate  payback  (PAYBACK)  and  Carbon  saving  per  pound
(CARBON SAVING POUND)
    //Multiply default payback by 140 and divide by TYPICAL building
heating consumption (kWh/m2.year)
    double mAREA=P.Vnum(gId,"AREA");
    double     mTYP KWH HEAT=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
HEAT")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")));
    if((mAREA!=0)&&(mTYP_KWH_HEAT!=0))
      PAYBACK=D.Value("dEPC H7")*140.0/(mTYP KWH HEAT/mAREA);
    else
      PAYBACK=0.0;

    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON_SAVING_POUND=BOrec.fCARBON_SAVING_POUND(D,CARBON_SAVING_NO);

    string EPC XX="EPC-H7";
    Write_Rec_Keyword(D,   T,   EPC_XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
  }

  //---- END recommendation EPC-H7 ----


  //---- recommendation EPC-H8 ----

  //weather compensation control

  if (EPCH8) {
    //assess impact of recommendation EPCH2 on CO2 and Energy

    //variable to store price weighted energy and CO2 for all HVACs
heating after recommendation is applied where applicable
```

```
    mHVAC HEAT ENERGY WT PW=0.0;
    mHVAC HEAT CO2 WEIGHTED=0.0;

    //assess impact of recommendation EPCH2 on energy and CO2
    //go through all HVAC heat generators in project
    //if DHW energy is provided by HVAC generator, include this
energy in HVAC assessment
    for (int i=0;i<vHVAC.size();i++) {
      //carbon intensity of fuel
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,P.Vcodid(vHVAC[i].Id,"FU
EL-TYPE"));
      else // all UK

mFUEL_CARBON_INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE"));
      //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
      //used in calculation of energy impact

Fuel Price Weighting=T.Value("tFuel Price Weighting",1,P.Vcodid(vHVAC
[i].Id,"FUEL-TYPE"));
      //primary energy used for space heating:
      mQ HEAT THIS HVAC PW=P.Vnum(vHVAC[i].Id,"Q-PRIM-
HEAT")*Fuel Price Weighting;
      mKG CO2 HEAT THIS HVAC=(P.Vnum(vHVAC[i].Id,"KWH-
HEAT")*mFUEL CARBON INTENSITY);
      deque<int>   mDHW SERVED REF=P.Vlrefid(vHVAC[i].Id,"DHW-SERVED-
REF");

      //energy for DHW:
      mQ DHW THIS HVAC PW=0.0;
      mKG CO2 DHW THIS HVAC=0.0;
      for (int j=0;j< mDHW SERVED REF.size();j++) {
        mQ DHW THIS HVAC PW+=P.Vnum(mDHW SERVED REF[j],"Q-
PRIM")*Fuel Price Weighting;
        mKG_CO2_DHW_THIS_HVAC+=P.Vnum(mDHW_SERVED_REF[j],"KWH-
DHW")*mFUEL CARBON INTENSITY;
      }
      // if there is weather compensation control
      if (P.Vcodid(vHVAC[i].Id,"WEATHER-COMP")==2) {
        if ((D.Value("REC DEF23"))!=0) {
          //weight    energy    =>    energy*(actual    eff/(actual
eff*D.Value("REC DEF23"))) (energy is already price weighted)

mHVAC_HEAT_ENERGY_WT_PW+=(mQ_HEAT_THIS_HVAC_PW+mQ_DHW_THIS_HVAC_PW)/D
.Value("REC DEF23");
          //weight      CO2      =>      CO2*(actual      eff/(actual
eff*D.Value("REC_DEF23")))

mHVAC HEAT CO2 WEIGHTED+=(mKG CO2 HEAT THIS HVAC+mKG CO2 DHW THIS HVA
C)/D.Value("REC_DEF23");
        }
      }
      else {
        //if  there  is  weather  compensation  control  add  energy
directly

mHVAC HEAT ENERGY WT PW+=(mQ HEAT THIS HVAC PW+mQ DHW THIS HVAC PW);
        //if there is weather compensation control add CO2 directly

mHVAC HEAT CO2 WEIGHTED+=(mKG CO2 HEAT THIS HVAC+mKG CO2 DHW THIS HVA
C);
      }
    }

    //variable for adjusted Total Building energy
```

```
    mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mQ HEAT TOTAL PRICE WEIGHTED+mHVAC HEAT ENERGY WT PW;
    mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
mKG CO2 HEAT TOTAL+mHVAC HEAT CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ_TOTAL_ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D,                   mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
    //Multiply default payback by 140 and divide by TYPICAL building
heating consumption (kWh/m2.year)
    double mAREA=P.Vnum(gId,"AREA");
    double     mTYP_KWH_HEAT=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
HEAT")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")));
    if((mAREA!=0)&&(mTYP KWH HEAT!=0))
      PAYBACK=D.Value("dEPC_H8")*140.0/(mTYP_KWH_HEAT/mAREA);
    else
      PAYBACK=0.0;

    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    string EPC XX="EPC-H8";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
  }

  //---- END recommendation EPC-H8 ----




  //---- Overall energy and carbon performance for heating ----
  //categorise heating as good/fair/poor, PERF-HEAT-SYSTEM-ENERGY
  //if  actual<notional,  classify  Overall  energy  performance  for
heating as good
  //if notional<=actual<typical, classify Overall energy performance
for heating as fair
  //otherwise classify Overall energy performance for heating as fair

  //energy
  string mPERF HEAT SYSTEM ENERGY="NOT APPLICABLE";

  if (P.Vnum(gId,"KWH-HEAT")>0.0) {
    if                                          (P.Vnum(gId,"KWH-
HEAT")<Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-HEAT")) {
      mPERF_HEAT_SYSTEM_ENERGY="GOOD";
    }
    else                      if                      (P.Vnum(gId,"KWH-
HEAT")<Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
HEAT")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")))) {
           mPERF HEAT SYSTEM ENERGY="FAIR";
         }
         else {
              mPERF HEAT SYSTEM ENERGY="POOR";
         }
  }

  {
  ostringstream res; res << mPERF HEAT SYSTEM ENERGY;
  P.ModKey(RP.Id,"PERF-HEAT-SYSTEM-ENERGY",res.str());     //  Overall
energy performance for heating
```

```
  }

  //CO2
  string mPERF HEAT SYSTEM CO2="NOT APPLICABLE";

  if (P.Vnum(gId,"KG-CO2-HEAT")>0.0) {
    if                                          (P.Vnum(gId,"KG-CO2-
HEAT")<Notional.P.Vnum(Notional.dGEN[0].Id,"KG-CO2-HEAT")) {
      mPERF HEAT SYSTEM CO2="GOOD";
    }
    else                    if                  (P.Vnum(gId,"KG-CO2-
HEAT")<Notional.P.Vnum(Notional.dGEN[0].Id,"KG-CO2-
HEAT")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")))) {
           mPERF_HEAT_SYSTEM_CO2="FAIR";
         }
         else {
              mPERF_HEAT_SYSTEM_CO2="POOR";
         }
  }

  {
  ostringstream res; res << mPERF HEAT SYSTEM CO2;
  P.ModKey(RP.Id,"PERF-HEAT-SYSTEM-CO2",res.str());     //Overall  CO2
performance for heating
  }

  //percentage of CO2 attributable to heating:
  //mKG CO2 HEAT TOTAL is total heating CO2 including DHW where this
is supplied by the HVAC system
  //mKG CO2 PROJECT is total building CO2 excluding renewables

  double HEAT PERCENT CO2=0.0;
  if (mKG CO2 PROJECT!=0.0)
    HEAT PERCENT CO2=(mKG CO2 HEAT TOTAL/mKG CO2 PROJECT)*100.0;

  {
  ostringstream res; res << HEAT PERCENT CO2;
  P.ModKey(RP.Id,"HEAT-PERCENT-CO2",res.str());
  }

  //percentage of ENERGY attributable to heating:
  //mQ HEAT TOTAL is total heating load including DHW where this is
supplied by the HVAC system
  //mQ TOTAL is total building energy for all end uses
  double HEAT_PERCENT_ENERGY=0.0;
  if (mQ TOTAL!=0.0)
    HEAT PERCENT ENERGY=(mQ HEAT TOTAL/mQ TOTAL)*100.0;

  {
  ostringstream res; res << HEAT PERCENT ENERGY;
  P.ModKey(RP.Id,"HEAT-PERCENT-ENERGY",res.str());
  }

  //---- End of Overall energy and carbon performance for heating ---
-




//2)cooling

  //total cooling energy and CO2 for all HVAC COOL generators in
project:
  double mQ COOL TOTAL=0.0;
  //total energy weighted by energy price
  double mQ_COOL_TOTAL_PRICE_WEIGHTED=0.0;
  double mKG CO2 COOL TOTAL=0.0;
  //energy and CO2 for 'GOOD' HVAC COOL generators:
  double mQ_COOL_GOOD=0.0;
```

```
  double mKG CO2 COOL GOOD=0.0;
  //energy and CO2 for 'FAIR' HVAC COOL generators:
  double mQ COOL FAIR=0.0;
  double mKG CO2 COOL FAIR=0.0;
  //energy and CO2 for 'POOR' HVAC COOL generators:
  double mQ COOL POOR=0.0;
  double mKG CO2 COOL POOR=0.0;

  //primary energy used for space cooling in project
  double mQ COOL=0.0;
  //price weighted primary energy used for space cooling in project
  double mQ COOL PW=0.0;
  double mKG CO2 COOL=0.0;

  Fuel Payback=0.0;

  int mTYPE;
  double mDuct Leakage=0.0;

  //keywords for recommendations
  //1->  recommendation  is  triggered,  0->recommendation  is  not
triggered
  bool EPCC2=0;
  bool EPCC1=0;
  bool EPCC3=0;

  //variable to store price weighted energy for HVAC cooling after
recommendation is applied
  double mHVAC COOL ENERGY WT PW=0.0;

 //variable to store CO2 for HVAC after recommendation is applied
  double mHVAC COOL CO2 WEIGHTED=0.0;

  //go through all HVAC systems
  for (int i=0;i<vHVAC.size();i++)  {
    //check if HVAC system provides cooling
    if (T.Value("tSYS COOL",1,P.Vcodid(vHVAC[i].Id,"TYPE"))==1) {
      //determine cooling fuel carbon intensity:
/*
     // If Heat Pump then Fuel Type for the Heat Pump defined in the
cooling Tab
      // is also used for the cooling system
      if          (P.Vcodid(vHVAC[i].Id,"HEAT-SOURCE")==12          ||
P.Vcodid(vHVAC[i].Id,"HEAT-SOURCE")==13) {
        if((P.Vcodid(cId,"BR-TYPE")==5)     ||     (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL_CARBON_INTENSITY=T.Value("tKGCO2_IR",1,P.Vcodid(vHVAC[i].Id,"FU
EL-TYPE"));
        else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE"));
        //determine fuel price weighting factor

Fuel Price Weighting=T.Value("tFuel Price Weighting",1,P.Vcodid(vHVAC
[i].Id,"FUEL-TYPE"));
      }
      else  {
        if((P.Vcodid(cId,"BR-TYPE")==5)     ||     (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL_CARBON_INTENSITY=T.Value("tKGCO2_IR",1,(int)D.Value("dELEC_CODI
D"));
        else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,(int)D.Value("dELEC CODID")
);
        //determine fuel price weighting factor
```

```
Fuel Price Weighting=T.Value("tFuel Price Weighting",1,(int)D.Value("
dELEC CODID"));
      }
*/
      // !!! 15/12/2008: Cooling now has its own fuel type keyword
      if((P.Vcodid(cId,"BR-TYPE")==5)      ||      (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,(int)T.Value("tFUEL TYPE
_COOL",1,P.Vcodid(vHVAC[i].Id,"FUEL-TYPE-COOL")));
      else // all UK

mFUEL_CARBON_INTENSITY=T.Value("tKGCO2",1,(int)T.Value("tFUEL_TYPE_CO
OL",1,P.Vcodid(vHVAC[i].Id,"FUEL-TYPE-COOL")));
      //determine fuel price weighting factor

Fuel Price Weighting=T.Value("tFuel Price Weighting",1,(int)T.Value("
tFUEL TYPE COOL",1,P.Vcodid(vHVAC[i].Id,"FUEL-TYPE-COOL")));


      //If the HVAC system is one of:
      //   VAV (including packaged cabinet), fan coil, induction,
constant volume, multizone,
      //  terminal reheat, dual duct, chilled ceiling or chilled beam
(with displacement ventilation)
      //check duct leakage
      mTYPE=P.Vcodid(vHVAC[i].Id,"TYPE");
      if ((mTYPE>11)&&(mTYPE<24)) {
        //If duct leakage>=5%, trigger recommendation EPCC3
        mDuct Leakage=P.Vnum(vHVAC[i].Id,"DUCT-LEAKAGE-PC");
        if (mDuct Leakage>=5) EPCC3=1;
      }

      //primary energy used for space cooling:
      mQ COOL=P.Vnum(vHVAC[i].Id,"Q-PRIM-COOL");
      //price weighted primary energy used for space cooling:
      mQ COOL PW=P.Vnum(vHVAC[i].Id,"Q-PRIM-
COOL")*Fuel_Price_Weighting;
      mKG CO2 COOL=(P.Vnum(vHVAC[i].Id,"KWH-
COOL")*mFUEL CARBON INTENSITY);
      mQ_COOL_TOTAL+=mQ_COOL;
      mQ COOL TOTAL PRICE WEIGHTED+=mQ COOL PW;
      mKG CO2 COOL TOTAL+=mKG CO2 COOL;
/*
     //fuel  payback  adjustment,  weight  with  energy  -  used  to
calculate carbon saved/pound spent
      if          (P.Vcodid(vHVAC[i].Id,"HEAT-SOURCE")==12          ||
P.Vcodid(vHVAC[i].Id,"HEAT-SOURCE")==13)

Fuel_Payback+=T.Value("tFuel_Payback",1,P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE"))*mQ COOL;
      else

Fuel Payback+=T.Value("tFuel Payback",1,(int)D.Value("dELEC CODID"))*
mQ COOL;
*/

Fuel Payback+=T.Value("tFuel Payback",1,(int)T.Value("tFUEL TYPE COOL
",1,P.Vcodid(vHVAC[i].Id,"FUEL-TYPE-COOL")))*mQ_COOL;

      Weighted Energy+=mQ COOL*(P.Vnum(vHVAC[i].Id,"COOL-GEN-EER"));
      Weighted_CO2+=mKG_CO2_COOL*(P.Vnum(vHVAC[i].Id,"COOL-GEN-
EER"));

      //if  using  default  cold  generator  efficiency,  trigger  EPCC1:
improve on default efficiency
      if (P.Vcodid(vHVAC[i].Id,"EFF-DEFAULT-COOL")==1)
         EPCC1=1;       // improve on default efficiency
```

```
    } //END OF check if HVAC system provides COOLING

  } //END OF loop through all HVAC systems


  //divide by total energy
  if (mQ_COOL_TOTAL!=0) {
     // Average Fuel Payback for cooling of the whole building
weighted by energy of all HVAC systems
     Fuel_Payback/=mQ_COOL_TOTAL;
     // Average energy performance for cooling of the whole building
weighted by energy of all HVAC systems
     Weighted_Energy/=mQ_COOL_TOTAL;
  }
  else {
    Fuel_Payback=0.0;
    Weighted Energy=0.0;
  }

  //  divide by total CO2
  if (mKG CO2 COOL TOTAL!=0) {
     // Average CO2 performance for cooling of the whole building
weighted by CO2 of all HVAC systems
     Weighted CO2/=mKG CO2 COOL TOTAL;
  }
  else {
    Weighted CO2=0.0;
  }

 //------HVAC cooling energy performance - whole building GOOD/ FAIR/
POOR------

  string PERFORMANCE COOL ENERGY="NOT APPLICABLE";

  //For whole building - is HVAC COOLING ENERGY USE classified as
GOOD, FAIR or POOR
  //    this decides whether you need to trigger recommendation
associated with FAIR or POOR.

  if (Weighted Energy>0.0) {
      if (Weighted_Energy>D.Value("REC_DEF5a")) {
      //if averaged efficiency is (>x) building end use is GOOD
        PERFORMANCE COOL ENERGY="GOOD";
      }
      else {              // i.e., will be either FAIR or POOR
           EPCC2=1;       // Trigger recommendation EPCC2: upgrade to
higher efficiency chiller
           //If  x  >=  average  efficiency  >  y,  classify  energy
performance as "fair"
           if (Weighted_Energy>D.Value("REC_DEF6")) {
              PERFORMANCE COOL ENERGY="FAIR";
           }
           else {    // i.e., average efficiency =< y, building end
use is POOR
                PERFORMANCE COOL ENERGY="POOR";
           }
      }
  }


  //------CO2 performance - whole building GOOD/ FAIR/ POOR------

  string PERFORMANCE COOL CO2="NOT APPLICABLE";

  //For whole building - is HVAC COOLING CO2 classified as GOOD, FAIR
or POOR
  //    this decides whether you need to trigger recommendation
associated with FAIR or POOR.
```

```
  if (Weighted CO2>0.0) {
      if (Weighted CO2>D.Value("REC DEF5a")) {
      //if averaged efficiency is (>x) building end use is GOOD
        PERFORMANCE_COOL_CO2="GOOD";
      }
      else {             // i.e., will be either FAIR or POOR
           EPCC2=1;       // Trigger recommendation EPCC2: upgrade to
higher efficiency chiller
           if (Weighted CO2>D.Value("REC DEF6")) {
           //If  x  >=  average  efficiency  >  y,  classify  energy
performance as "fair"
              PERFORMANCE COOL CO2="FAIR";
           }
           else {    // i.e., average efficiency =< y, building end
use is POOR
                PERFORMANCE_COOL_CO2="POOR";
           }
      }
  }


//write hvac COOLing generator efficiency keywords
//PERF-COOl-ENERGY and PERF-COOL-CO2 Codes: 1 - GOOD, 2 - FAIR, 3 -
POOR, 4 - NOT APPLICABLE
 {
  ostringstream res; res << PERFORMANCE COOL ENERGY;
  P.ModKey(RP.Id,"PERF-COOL-ENERGY",res.str());          //  Averaged
system energy efficiency for cooling
 }
 {
  ostringstream res; res << PERFORMANCE COOL CO2;
  P.ModKey(RP.Id,"PERF-COOL-CO2",res.str());       // Averaged system
CO2 efficiency for cooling
 }


  //---- recommendation EPC-C2 ----

  //high efficency chiller

  if (EPCC2) {

    //assess impact of recommendation EPCC2 on CO2 and Energy

    //variable to store price-weighted energy and CO2 for all HVACs
cooling after recommendation is applied where applicable
    mHVAC_COOL_ENERGY_WT_PW=0.0;
    mHVAC COOL CO2 WEIGHTED=0.0;

    //assess impact of recommendation EPCC2 on energy and CO2

    //go through all HVAC cool generators in project
    for (int i=0;i<vHVAC.size();i++) {
      //check if HVAC system provides cooling
      if (T.Value("tSYS COOL",1,P.Vcodid(vHVAC[i].Id,"TYPE"))==1) {
        //determine cooling fuel carbon intensity:
/*
        // If Heat Pump then Fuel Type for the Heat Pump defined in
the Heating Tab
        // is also used for the cooling system
        if         (P.Vcodid(vHVAC[i].Id,"HEAT-SOURCE")==12         ||
P.Vcodid(vHVAC[i].Id,"HEAT-SOURCE")==13) {
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,P.Vcodid(vHVAC[i].Id,"FU
EL-TYPE"));
          else // all UK
```

```
mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE"));
          //price weighting value from table tFuel Price Weighting
(depends on relative prices of fuels)
          //used in calculation of energy impact

Fuel_Price_Weighting=T.Value("tFuel_Price_Weighting",1,P.Vcodid(vHVAC
[i].Id,"FUEL-TYPE"));
        }
        else {
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,(int)D.Value("dELEC CODI
D"));
          else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,(int)D.Value("dELEC CODID")
);
          //price weighting value from table tFuel Price Weighting
(depends on relative prices of fuels)
          //used in calculation of energy impact

Fuel Price Weighting=T.Value("tFuel Price Weighting",1,(int)D.Value("
dELEC CODID"));
        }
*/
        if((P.Vcodid(cId,"BR-TYPE")==5)     ||     (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,(int)T.Value("tFUEL TYPE
 COOL",1,P.Vcodid(vHVAC[i].Id,"FUEL-TYPE-COOL")));
        else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,(int)T.Value("tFUEL TYPE CO
OL",1,P.Vcodid(vHVAC[i].Id,"FUEL-TYPE-COOL")));
        //price weighting value from table tFuel Price Weighting
(depends on relative prices of fuels)
        //used in calculation of energy impact

Fuel_Price_Weighting=T.Value("tFuel_Price_Weighting",1,(int)T.Value("
tFUEL TYPE COOL",1,P.Vcodid(vHVAC[i].Id,"FUEL-TYPE-COOL")));

        //primary energy used for space cooling:
        mQ COOL PW=P.Vnum(vHVAC[i].Id,"Q-PRIM-
COOL")*Fuel Price Weighting;
        mKG_CO2_COOL=(P.Vnum(vHVAC[i].Id,"KWH-
COOL")*mFUEL CARBON INTENSITY);

        if  (P.Vnum(vHVAC[i].Id,"COOL-GEN-EER")<=D.Value("REC DEF5a"))
{
          //if eff <= D.Value("REC DEF5a") correct energy according
to recommendation (set efficiency to D.Value("REC_DEF5"))
          //weight energy => energy*(old eff/D.Value("REC DEF5"))
          if ((D.Value("REC DEF5"))!=0) {
            //weight energy => energy/D.Value("REC_DEF3")*actual eff

mHVAC COOL ENERGY WT PW+=mQ COOL PW*(P.Vnum(vHVAC[i].Id,"COOL-GEN-
EER")/D.Value("REC_DEF5"));
            //weight CO2 => CO2*(actual eff/D.Value("REC DEF3"))

mHVAC_COOL_CO2_WEIGHTED+=mKG_CO2_COOL*(P.Vnum(vHVAC[i].Id,"COOL-GEN-
EER")/D.Value("REC DEF5"));
          }
        }
        else {
          //if eff > D.Value("REC DEF5a") add energy and CO2 directly
          mHVAC_COOL_ENERGY_WT_PW+=mQ_COOL_PW;
```

```
          //if eff > D.Value("REC DEF5a") add CO2 directly
          mHVAC COOL CO2 WEIGHTED+=mKG CO2 COOL;
        }
      }
    }

    //variable for adjusted Total Building energy
    mQ_TOTAL_ADJUSTED=mQ_TOTAL_PRICE_WEIGHTED-
mQ COOL TOTAL PRICE WEIGHTED+mHVAC COOL ENERGY WT PW;
   //variable for adjusted Total Building CO2
    mCO2_TOTAL_ADJUSTED=mKG_CO2_PROJECT-
mKG CO2 COOL TOTAL+mHVAC COOL CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
    IMPACT_CO2=BOrec.fIMPACT_CO2(D,                    mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
    //Multiply default payback by 30 and divide by (1.2*NOTIONAL)
building cooling consumption (kWh/m2.year)
    double mAREA=P.Vnum(gId,"AREA");
    double     mNOT KWH COOL=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
COOL");
    if((mAREA!=0)&&(mNOT KWH COOL!=0))
      PAYBACK=D.Value("dEPC C2")*30.0/((mNOT KWH COOL*1.2)/mAREA);
    else
      PAYBACK=0.0;

    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    //write the EPCC2 recommendation keyword
    string EPC_XX="EPC-C2";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
  }

  //---- END recommendation EPC-C2 ----


  //---- recommendation EPC-C1 ----

  //default efficency chiller

  if (EPCC1) {

    //assess impact of recommendation EPCC1 on CO2 and Energy

    //variable to store price-weighted energy and CO2 for all HVACs
cooling after recommendation is applied where applicable
    mHVAC_COOL_ENERGY_WT_PW=0.0;
    mHVAC COOL CO2 WEIGHTED=0.0;

    //assess impact of recommendation EPCC1 on energy and CO2

    //go through all HVAC cool generators in project
    for (int i=0;i<vHVAC.size();i++) {
      //check if HVAC system provides cooling
      if (T.Value("tSYS COOL",1,P.Vcodid(vHVAC[i].Id,"TYPE"))==1) {
        //determine cooling fuel carbon intensity:
/*
        // If Heat Pump then Fuel Type for the Heat Pump defined in
the Heating Tab
```

```
        // is also used for the cooling system
        if         (P.Vcodid(vHVAC[i].Id,"HEAT-SOURCE")==12         ||
P.Vcodid(vHVAC[i].Id,"HEAT-SOURCE")==13) {  // 05 Sep 08, JO: Same as
HVAC and above
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL_CARBON_INTENSITY=T.Value("tKGCO2_IR",1,P.Vcodid(vHVAC[i].Id,"FU
EL-TYPE"));
          else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE"));
          //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
          //used in calculation of energy impact

Fuel Price Weighting=T.Value("tFuel Price Weighting",1,P.Vcodid(vHVAC
[i].Id,"FUEL-TYPE"));
        }
        else {
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,(int)D.Value("dELEC CODI
D"));
          else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,(int)D.Value("dELEC CODID")
);
          //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
          //used in calculation of energy impact

Fuel Price Weighting=T.Value("tFuel Price Weighting",1,(int)D.Value("
dELEC CODID"));
        }
*/
        // !!! 15/12/2008: Cooling now has its own fuel type keyword
        if((P.Vcodid(cId,"BR-TYPE")==5)     ||     (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,(int)T.Value("tFUEL TYPE
 COOL",1,P.Vcodid(vHVAC[i].Id,"FUEL-TYPE-COOL")));
        else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,(int)T.Value("tFUEL TYPE CO
OL",1,P.Vcodid(vHVAC[i].Id,"FUEL-TYPE-COOL")));
        //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
        //used in calculation of energy impact

Fuel Price Weighting=T.Value("tFuel Price Weighting",1,(int)T.Value("
tFUEL_TYPE_COOL",1,P.Vcodid(vHVAC[i].Id,"FUEL-TYPE-COOL")));

        //primary energy used for space cooling:
        mQ_COOL_PW=P.Vnum(vHVAC[i].Id,"Q-PRIM-
COOL")*Fuel Price Weighting;
        mKG CO2 COOL=(P.Vnum(vHVAC[i].Id,"KWH-
COOL")*mFUEL_CARBON_INTENSITY);

        // if default chiller efficiency is used
        if (P.Vcodid(vHVAC[i].Id,"EFF-DEFAULT-COOL")==1) {
          //if  default  efficiency  is  used,  correct  energy  according
to recommendation (set efficiency to D.Value("REC DEF5"))
          //weight energy => energy*(old eff/D.Value("REC_DEF5"))
          if ((D.Value("REC DEF5"))!=0) {
            //weight energy => energy/D.Value("REC DEF3")*actual eff
```

```
mHVAC COOL ENERGY WT PW+=mQ COOL PW*(P.Vnum(vHVAC[i].Id,"COOL-GEN-
EER")/D.Value("REC DEF5"));
            //weight CO2 => CO2*(actual eff/D.Value("REC DEF3"))

mHVAC_COOL_CO2_WEIGHTED+=mKG_CO2_COOL*(P.Vnum(vHVAC[i].Id,"COOL-GEN-
EER")/D.Value("REC DEF5"));
          }
        }
        else {
          //if default efficiency is not used, add energy and CO2
directly
          mHVAC COOL ENERGY WT PW+=mQ COOL PW;
          //if default efficiency is not used, add CO2 directly
          mHVAC_COOL_CO2_WEIGHTED+=mKG_CO2_COOL;
        }
      }
    }

    //variable for adjusted Total Building energy
    mQ_TOTAL_ADJUSTED=mQ_TOTAL_PRICE_WEIGHTED-
mQ COOL TOTAL PRICE WEIGHTED+mHVAC COOL ENERGY WT PW;
   //variable for adjusted Total Building CO2
    mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
mKG CO2 COOL TOTAL+mHVAC COOL CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D,                   mKG CO2 PROJECT,
mCO2 TOTAL ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
    //Multiply default payback by 30 and divide by (1.2*NOTIONAL)
building cooling consumption (kWh/m2.year)
    double mAREA=P.Vnum(gId,"AREA");
    double     mNOT KWH COOL=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
COOL");
    if((mAREA!=0)&&(mNOT_KWH_COOL!=0))
      PAYBACK=D.Value("dEPC C1")*30.0/((mNOT KWH COOL*1.2)/mAREA);
    else
      PAYBACK=0.0;

    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON_SAVING_POUND=BOrec.fCARBON_SAVING_POUND(D,CARBON_SAVING_NO);

    //write the EPCC1 recommendation keyword
    string EPC_XX="EPC-C1";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
  }

  //---- END recommendation EPC-C1 ----


  //---- recommendation EPC-C3 ----

  //duct leakage (cooling)

  if (EPCC3) {
    //assess impact of recommendation EPCC3 on CO2 and Energy

    //variable to store price-weighted energy and CO2 for all HVACs
cooling after recommendation is applied where applicable
    mHVAC COOL ENERGY WT PW=0.0;
    mHVAC_COOL_CO2_WEIGHTED=0.0;
```

```
    int row;
    double mDuct Leakage=0.0;
    int mTYPE;
    double mP_Factor=1.0;

    //go through all HVAC cool generators in project
    for (int i=0;i<vHVAC.size();i++) {
      //check if HVAC system provides cooling
      if (T.Value("tSYS COOL",1,P.Vcodid(vHVAC[i].Id,"TYPE"))==1) {
       //determine cooling fuel carbon intensity:
/*
        // If Heat Pump then Fuel Type for the Heat Pump defined in
the Heating Tab
        // is also used for the cooling system
        if         (P.Vcodid(vHVAC[i].Id,"HEAT-SOURCE")==12         ||
P.Vcodid(vHVAC[i].Id,"HEAT-SOURCE")==13) { // 05 Sep 08, JO: Same as
HVAC and above
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL CARBON INTENSITY=T.Value("tKGCO2 IR",1,P.Vcodid(vHVAC[i].Id,"FU
EL-TYPE"));
          else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,P.Vcodid(vHVAC[i].Id,"FUEL-
TYPE"));
          //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
          //used in calculation of energy impact

Fuel Price Weighting=T.Value("tFuel Price Weighting",1,P.Vcodid(vHVAC
[i].Id,"FUEL-TYPE"));
        }
        else {
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL_CARBON_INTENSITY=T.Value("tKGCO2_IR",1,(int)D.Value("dELEC_CODI
D"));
          else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,(int)D.Value("dELEC CODID")
);
          //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
          //used in calculation of energy impact

Fuel Price Weighting=T.Value("tFuel Price Weighting",1,(int)D.Value("
dELEC CODID"));
        }
*/
        // !!! 15/12/2008: Cooling now has its own fuel type keyword
        if((P.Vcodid(cId,"BR-TYPE")==5)     ||     (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only

mFUEL_CARBON_INTENSITY=T.Value("tKGCO2_IR",1,(int)T.Value("tFUEL_TYPE
 COOL",1,P.Vcodid(vHVAC[i].Id,"FUEL-TYPE-COOL")));
        else // all UK

mFUEL CARBON INTENSITY=T.Value("tKGCO2",1,(int)T.Value("tFUEL TYPE CO
OL",1,P.Vcodid(vHVAC[i].Id,"FUEL-TYPE-COOL")));
        //price  weighting  value  from  table  tFuel Price Weighting
(depends on relative prices of fuels)
        //used in calculation of energy impact

Fuel Price Weighting=T.Value("tFuel Price Weighting",1,(int)T.Value("
tFUEL TYPE COOL",1,P.Vcodid(vHVAC[i].Id,"FUEL-TYPE-COOL")));
```

```
        //primary energy used for space cooling:
        mQ COOL PW=P.Vnum(vHVAC[i].Id,"Q-PRIM-
COOL")*Fuel Price Weighting;
        mKG CO2 COOL=(P.Vnum(vHVAC[i].Id,"KWH-
COOL")*mFUEL_CARBON_INTENSITY);

        mTYPE=P.Vcodid(vHVAC[i].Id,"TYPE");
        if ((mTYPE>11)&&(mTYPE<24)) {
          //If duct leakage>=5%, trigger recommendation EPCC3
          mDuct Leakage=P.Vnum(vHVAC[i].Id,"DUCT-LEAKAGE-PC");
          if (mDuct_Leakage>=5) {
            //assess impact on cooling
            //Reduce cooling energy by P% where P is taken from table
tDUCT_LEAK_CORR
            //use the right row to read the correction factor from
the table, this depends on the % leakage
            if (mDuct_Leakage<10) row=1;
            else row=2;

            mP_Factor=T.Value("tDUCT_LEAK_CORR",row,mTYPE);

            //correct energy according to recommendation (reduce HVAC
cooling energy by P%)
            mHVAC COOL ENERGY WT PW=mQ COOL PW*(100.0-
mP Factor)/100.0;
            //weight CO2 (reduce HVAC cooling energy by P%)
            mHVAC COOL CO2 WEIGHTED=mKG CO2 COOL*(100.0-
mP Factor)/100.0;
          }
          else {
            //if duct leakage<5%, add energy and CO2 directly
            mHVAC COOL ENERGY WT PW+=mQ COOL PW;
            //if duct leakage<5%, add CO2 directly
            mHVAC COOL CO2 WEIGHTED+=mKG CO2 COOL;
          }
        }
      }    //end of check if system provides cooling
    }    //end of loop through HVAC systems

    //variable for adjusted Total Building energy
    mQ TOTAL ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mQ_COOL_TOTAL_PRICE_WEIGHTED+mHVAC_COOL_ENERGY_WT_PW;
   //variable for adjusted Total Building CO2
    mCO2 TOTAL ADJUSTED=mKG CO2 PROJECT-
mKG_CO2_COOL_TOTAL+mHVAC_COOL_CO2_WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ TOTAL ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D,                   mKG CO2 PROJECT,
mCO2_TOTAL_ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON_SAVING_POUND)
    //Multiply default payback by 60 and divide by NOTIONAL building
cooling consumption (kWh/m2.year)
    double     mNOT_KWH_COOL=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
COOL");
    double mAREA=P.Vnum(gId,"AREA");
    if((mAREA!=0)&&(mNOT_KWH_COOL!=0))
      PAYBACK=D.Value("dEPC C3")*60.0/(mNOT KWH COOL/mAREA);
    else
      PAYBACK=0.0;

    double CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON_SAVING_POUND=BOrec.fCARBON_SAVING_POUND(D,CARBON_SAVING_NO);
```

```
    //write the EPCC3 recommendation keyword
    string EPC XX="EPC-C3";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON_SAVING_POUND);
  }

  //---- END recommendation EPC-C3 ----

  //percentage of CO2 attributable to COOLing:
  double COOL_PERCENT_CO2=0.0;
  if (mKG CO2 PROJECT!=0.0)
    COOL PERCENT CO2=(mKG CO2 COOL TOTAL/mKG CO2 PROJECT)*100.0;

  {
  ostringstream res; res << COOL PERCENT CO2;
  P.ModKey(RP.Id,"COOL-PERCENT-CO2",res.str());
  }


  //---- Overall energy and carbon performance for COOLing ----
  //categorise COOLing as good/fair/poor, PERF-COOL-SYSTEM-ENERGY
  //if actual<0.85*notional, classify Overall energy performance for
COOLing as good
  //if  0.85*notional<=actual<1.5*notional,  classify  Overall  energy
performance for COOLing as fair
  //otherwise classify Overall energy performance for COOLing as fair

  //energy
  string mPERF COOL SYSTEM ENERGY="NOT APPLICABLE";

  if (P.Vnum(gId,"KWH-COOL")>0.0) {
    if                                          (P.Vnum(gId,"KWH-
COOL")<D.Value("REC DEF18")*Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
COOL")) {
      mPERF COOL SYSTEM ENERGY="GOOD";
    }
    else                      if                      (P.Vnum(gId,"KWH-
COOL")<D.Value("REC_DEF19")*Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
COOL")) {
           mPERF COOL SYSTEM ENERGY="FAIR";
         }
         else {
              mPERF COOL SYSTEM ENERGY="POOR";
         }
  }

  {
  ostringstream res; res << mPERF COOL SYSTEM ENERGY;
  P.ModKey(RP.Id,"PERF-COOL-SYSTEM-ENERGY",res.str());     // Overall
energy performance for cooling
  }

  //CO2
  string mPERF COOL SYSTEM CO2="NOT APPLICABLE";

  if (P.Vnum(gId,"KG-CO2-COOL")>0.0) {
    if                                        (P.Vnum(gId,"KG-CO2-
COOL")<D.Value("REC DEF18")*Notional.P.Vnum(Notional.dGEN[0].Id,"KG-
CO2-COOL")) {
      mPERF COOL SYSTEM CO2="GOOD";
    }
    else                    if                    (P.Vnum(gId,"KG-CO2-
COOL")<D.Value("REC DEF19")*Notional.P.Vnum(Notional.dGEN[0].Id,"KG-
CO2-COOL")) {
           mPERF_COOL_SYSTEM_CO2="FAIR";
         }
         else {
              mPERF_COOL_SYSTEM_CO2="POOR";
```

```
        }
  }

  {
  ostringstream res; res << mPERF_COOL_SYSTEM_CO2;
  P.ModKey(RP.Id,"PERF-COOL-SYSTEM-CO2",res.str());      //Overall CO2
performance for cooling
  }

  //---- End of Overall energy and carbon performance for cooling ---
-


  // auxiliary
  //categorise AUXILIARY as good/fair/poor, PERF-AUX-SYSTEM-ENERGY
  //if actual<notional, classify Overall energy performance for AUX
as good
  //if notional<=actual<typical, classify Overall energy performance
for AUX as fair
  //otherwise classify Overall energy performance for AUX as fair

  //energy
  string mPERF AUX SYSTEM ENERGY="NOT APPLICABLE";

  if (P.Vnum(gId,"KWH-AUX")>0.0) {
    if                                              (P.Vnum(gId,"KWH-
AUX")<Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-AUX")) {
      mPERF AUX SYSTEM ENERGY="GOOD";
    }
    else                      if                    (P.Vnum(gId,"KWH-
AUX")<Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
AUX")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")))) {
          mPERF AUX SYSTEM ENERGY="FAIR";
        }
        else {
             mPERF AUX SYSTEM ENERGY="POOR";
        }
  }

  {
  ostringstream res; res << mPERF AUX SYSTEM ENERGY;
  P.ModKey(RP.Id,"PERF-AUX-SYSTEM-ENERGY",res.str());       //Overall
energy performance for auxiliary
  }

  //CO2
  string mPERF AUX SYSTEM CO2="NOT APPLICABLE";

  if (P.Vnum(gId,"KG-CO2-AUX")>0.0) {
    if                                          (P.Vnum(gId,"KG-CO2-
AUX")<Notional.P.Vnum(Notional.dGEN[0].Id,"KG-CO2-AUX")) {
      mPERF AUX SYSTEM CO2="GOOD";
    }
    else                    if                  (P.Vnum(gId,"KG-CO2-
AUX")<Notional.P.Vnum(Notional.dGEN[0].Id,"KG-CO2-
AUX")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")))) {
          mPERF_AUX_SYSTEM_CO2="FAIR";
        }
        else {
             mPERF_AUX_SYSTEM_CO2="POOR";
        }
  }

  {
  ostringstream res; res << mPERF AUX SYSTEM CO2;
  P.ModKey(RP.Id,"PERF-AUX-SYSTEM-CO2",res.str());      //Overall CO2
performance for auxiliary
  }
```

```
  //---- End of Overall energy and carbon performance for AUXILIARY -
---

  //percentage of CO2 attributable to AUX
  double AUX_PERCENT_CO2=0.0;
  if (mKG CO2 PROJECT!=0.0)
    AUX PERCENT CO2=(P.Vnum(gId,"KG-CO2-AUX")/mKG CO2 PROJECT)*100.0;

  {
  ostringstream res; res << AUX PERCENT CO2;
  P.ModKey(RP.Id,"AUX-PERCENT-CO2",res.str());
  }


  //this keyword is the bulding CO2 if there were no renewables:
//  double mKG CO2 PROJECT=P.Vnum(gId,"KG-CO2-EXCL-RENEWABLES");

}
//end of PROJECT::HVAC Recommendation(...)




void PROJECT::ZONE Recommendation(PROJECTDefaults& D, PROJECTTables&
T, PROJECT& Notional) {
  if(func report)    cout    <<    endl    <<    "         "    <<
"PROJECT::ZONE Recommendation";
/*
  EPC recommendations function

  This function:
     - assesses overall energy and CO2 performance for lighting
     - calculates the percentage of CO2 and energy attributable to
lighting
     - assesses the impact of recommendations for lighting

    Project level recommendations for lighting are:

         EPC-L1 - T12 to T8
           -to assess impact, Calculate modified energy/ CO2 when all
instances of lighting type T12 is
            changed to typical T8.
         EPC-L2 - GLS to CFL
           -to assess impact, Calculate modified energy/ CO2 when all
instances of lighting type GLS is
            changed to typical CFL.
         EPC-L3   - HP mercury to SON replacements
           -to assess impact, Calculate modified energy/ CO2 when all
instances of lighting type HP mercury
            is changed to typical high pressure sodium.
         EPC-L5   - T8 to T5
           -to assess impact, Calculate modified energy/ CO2 when all
instances of lighting type T8 is
            changed to typical T5.
         EPC-L6   - HP mercury to SON
           -to assess impact, Calculate modified energy/ CO2 when all
instances of lighting type HP mercury
            is changed to typical high pressure sodium.
         EPC-L7   - HF ballast
           -to assess impact, Calculate modified energy/ CO2 when all
instances of lighting type T8 (mains frequency ballast) is
             changed to typical T8 (high frequency ballast).

  NOTE:  for  zone  level  recommendations  -  remember  to  include
multiplier!

*/
```

```
  //percentage of CO2 attributable to lighting:
  double mKG CO2 PROJECT=P.Vnum(gId,"KG-CO2-EXCL-RENEWABLES");
  double LIGHT PERCENT CO2=0.0;
  if (mKG CO2 PROJECT!=0.0)
    LIGHT_PERCENT_CO2=(P.Vnum(gId,"KG-CO2-
LI")/mKG CO2 PROJECT)*100.0;

  {
  ostringstream res; res << LIGHT PERCENT CO2;
  P.ModKey(RP.Id,"LIGHT-PERCENT-CO2",res.str());
  }

  //---- Overall energy and carbon performance for lighting ----
  //categorise lighting as good/fair/poor, PERF-LIGHT-SYSTEM-ENERGY
  //if  actual<notional,  classify  Overall  energy  performance  for
lighting as good
  //if notional<=actual<typical, classify Overall energy performance
for lighting as fair
  //otherwise  classify  Overall  energy  performance  for  lighting  as
fair

  //energy
  string mPERF LIGHT SYSTEM ENERGY="NOT APPLICABLE";

  if (P.Vnum(gId,"KWH-LI")>0.0) {
    if                                                (P.Vnum(gId,"KWH-
LI")<Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-LI")) {
      mPERF LIGHT SYSTEM ENERGY="GOOD";
    }
    else                      if                      (P.Vnum(gId,"KWH-
LI")<Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
LI")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")))) {
           mPERF LIGHT SYSTEM ENERGY="FAIR";
         }
         else {
              mPERF LIGHT SYSTEM ENERGY="POOR";
         }
  }

  {
  ostringstream res; res << mPERF LIGHT SYSTEM ENERGY;
  P.ModKey(RP.Id,"PERF-LIGHT-SYSTEM-ENERGY",res.str());      //Overall
energy performance for lighting
  }

  //CO2
  string mPERF LIGHT SYSTEM CO2="NOT APPLICABLE";

  if (P.Vnum(gId,"KG-CO2-LI")>0.0) {
    if                                              (P.Vnum(gId,"KG-CO2-
LI")<Notional.P.Vnum(Notional.dGEN[0].Id,"KG-CO2-LI")) {
      mPERF LIGHT SYSTEM CO2="GOOD";
    }
    else                    if                      (P.Vnum(gId,"KG-CO2-
LI")<Notional.P.Vnum(Notional.dGEN[0].Id,"KG-CO2-
LI")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")))) {
           mPERF_LIGHT_SYSTEM_CO2="FAIR";
         }
         else {
              mPERF_LIGHT_SYSTEM_CO2="POOR";
         }
  }

  {
  ostringstream res; res << mPERF LIGHT SYSTEM CO2;
  P.ModKey(RP.Id,"PERF-LIGHT-SYSTEM-CO2",res.str());     //Overall CO2
performance for lighting
  }
```

```
  //---- End of Overall energy and carbon performance for lighting --
--


  //--------  recommendations for lamp switching  ----------------

  //T12 to T8
  bool EPCL1=0;
  //GLS to CFL
  bool EPCL2=0;
  //HP mercury to SON replacements
  bool EPCL3=0;
  //GLS to LV tungsten halogen
//  bool EPCL4=0;
  //T8 to T5
  bool EPCL5=0;
  //HP mercury to SON
  bool EPCL6=0;
  //HF ballast
  bool EPCL7=0;

  //go through all ZONES and trigger relevant recommendations
  for (int i=0;i<vHVAC.size();i++) {
    for (int j=0;j<vHVAC[i].vZ.size();j++) {
      //look at lamp type:
      switch(P.Vcodid(vHVAC[i].vZ[j].Id,"LIGHT-TYPE")) {
        case 8: { //GLS lamps (tungsten or halogen)
          EPCL2=1; //GLS to CFL
          //  The next recommendation regarding switching to LV
tunsten halogen has been disabled as tungsten and halogen
          // are now the same option in the lamp types
          // EPCL4=1; //GLS to LV tungsten halogen
        }
        break;
        case 10: { //T12 lamps
          EPCL1=1; //T12 to T8
        }
        break;
        case 11: { //T8 halophosphate lamps with mains frequency
ballasts (low frequency ballast)
          EPCL7=1; //Mains to HF ballast
        }
        //no break here -> go to next case
        case 12: { //T8 halophosphate (high frequency ballast)
        }
        //no break here -> go to next case
        case 13: { //T8 triphosphor (high frequency ballast)
          EPCL5=1;  // T8 to T5
        }
        break;
        case 15: { //high-pressure mercury discharge lamps
          EPCL3=1; //HP mercury to SON replacements
          EPCL6=1; //HP mercury to SON
        }
        break;
      }
    } //end of loop through zones
  } //end of loop through HVACs


  //-------- assess impact of triggered recommendations  ------------
----

  //total building energy for all end uses (to be used when assessing
impact of recommendation):
  //this    is    price    weighted    using    factor    from    table
tFuel Price Weighting (depends on relative prices of fuels)
  //used in calculation of energy impact for recommendations
  double mQ_TOTAL_PRICE_WEIGHTED=P.Vnum(gId,"Q-TOTAL-P-WEIGHTED");
```

```
  //variable to store energy for lighting after recommendation is
applied
  double mLI ENERGY WT PW=0.0;
  //variable to store CO2 for lighting recommendation is applied
  double mLI CO2 WEIGHTED=0.0;

  //fuel payback factor for grid supplied electricity
  double
Fuel Payback=T.Value("tFuel Payback",1,(int)D.Value("dELEC CODID"));
  //price weighting value from table tFuel Price Weighting (depends
on relative prices of fuels)
  //used in calculation of energy impact
  double
Fuel Price Weighting=T.Value("tFuel Price Weighting",1,(int)D.Value("
dELEC CODID"));
  //price weighted whole project lighting energy
  double            mLI ENERGY PRICE WEIGHTED=P.Vnum(dGEN[0].Id,"KWH-
LI")*Fuel Price Weighting; //kWh
  //whole project lighting CO2
  double mLI CO2;
  if((P.Vcodid(cId,"BR-TYPE")==5)   ||   (P.Vcodid(cId,"EPC-TYPE")==6))
//Republic of Ireland only
     mLI CO2=P.Vnum(dGEN[0].Id,"KWH-
LI")*T.Value("tKGCO2 IR",1,(int)D.Value("dELEC CODID"));
  else // all UK
     mLI CO2=P.Vnum(dGEN[0].Id,"KWH-
LI")*T.Value("tKGCO2",1,(int)D.Value("dELEC CODID"));
  //stores impact of recommendation on CO2 (1=HIGH/2=MEDIUM/3=LOW)
  int IMPACT CO2=1;
  //stores impact of recommendation on energy (1=HIGH/2=MEDIUM/3=LOW)
  int IMPACT ENERGY=1;
  //payback period
  double PAYBACK=0;
  //'carbon saving per pound' value:
  double CARBON SAVING NO;
  //'carbon saving per pound' code:
  int CARBON SAVING POUND=1;
  //light type codid:
  int mLIGHT TYPE;

  double mAREA=P.Vnum(gId,"AREA");

  //---- recommendation EPC-L1 ----

  //if recommendation EPCL1 is triggered:
  if (EPCL1) {
  //T12 to T8
    //go through all ZONES
    for (int i=0;i<vHVAC.size();i++) {
      for (int j=0;j<vHVAC[i].vZ.size();j++) {
        //look at lamp type, if it is T12, switch to T8
        mLIGHT TYPE=P.Vcodid(vHVAC[i].vZ[j].Id,"LIGHT-TYPE");
        if (mLIGHT_TYPE==10) { // 10=T12 Fluorescent - halophosphate
- low frequency ballast
          double                    mLI POWER DENS=T.Value("tLUMINOUS-
EFFICACY",P.Vint(vHVAC[i].vZ[j].Id,"DRIVER2A")+1,mLIGHT_TYPE);
          // 13=T8 Fluorescent - triphosphor - high frequency ballast
          double                  mLI POWER DENS T8=T.Value("tLUMINOUS-
EFFICACY",P.Vint(vHVAC[i].vZ[j].Id,"DRIVER2A")+1,13);
          mLI ENERGY WT PW+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*(mLI POWER DENS/mLI POWER
_DENS_T8)*Fuel_Price_Weighting;
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only
             mLI_CO2_WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2 IR",1,(in
t)D.Value("dELEC CODID"))*(mLI POWER DENS/mLI POWER DENS T8);
//CO2/m2
```

```
          else // all UK
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2",1,(int)D
.Value("dELEC CODID"))*(mLI POWER DENS/mLI POWER DENS T8); //CO2/m2
          }
          else {
            mLI ENERGY WT PW+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*Fuel_Price_Weighting;
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only
             mLI_CO2_WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2 IR",1,(in
t)D.Value("dELEC CODID")); //KHW*CO2/KWH=CO2
          else // all UK
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2",1,(int)D
.Value("dELEC_CODID")); //KHW*CO2/KWH=CO2
          }
       }
    }

    //adjusted total building energy = total building MJ - whole
project lighting energy + adjusted whole project lighting energy
    double                        mQ ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mLI ENERGY PRICE WEIGHTED+mLI ENERGY WT PW;

    //adjusted total building CO2 = total building CO2 (excluding
renewables) - whole project lighting CO2 + adjusted whole project
lighting CO2
    double mCO2 ADJUSTED=mKG CO2 PROJECT-mLI CO2+mLI CO2 WEIGHTED;

    //assess energy and CO2 impact

    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D, mKG CO2 PROJECT, mCO2 ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON_SAVING_POUND)
    //Multiply default payback by 30 and divide by TYPICAL building
lighting consumption (kWh/m2.year)
    double       mTYP_KWH_LI=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
LI")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")));
    if((mAREA!=0)&&(mTYP KWH LI!=0))
      PAYBACK=D.Value("dEPC_L1")*30.0/(mTYP_KWH_LI/mAREA);
    else
      PAYBACK=0.0;

    CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON_SAVING_POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    //write the EPCL1 recommendation keyword
    string EPC XX="EPC-L1";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
  }
  //---- END recommendation EPC-L1 ----

  //---- recommendation EPC-L5 ----

  //if recommendation EPCL5 is triggered:
  if (EPCL5) {
  //T8 to T5
    //go through all ZONES
    for (int i=0;i<vHVAC.size();i++) {
      for (int j=0;j<vHVAC[i].vZ.size();j++) {
```

```
        //look at lamp type, if it is T8, switch to T5
        mLIGHT TYPE=P.Vcodid(vHVAC[i].vZ[j].Id,"LIGHT-TYPE");
        // 11=T8 Fluorescent - halophosphate - low frequency ballast
        // 12=T8 Fluorescent - halophosphate - high frequency ballast
        // 13=T8 Fluorescent - triphosphor - high frequency ballast
        if    (    (mLIGHT TYPE==11)    ||    (mLIGHT TYPE==12)    ||
(mLIGHT TYPE==13) ) {
          double                    mLI_POWER_DENS=T.Value("tLUMINOUS-
EFFICACY",P.Vint(vHVAC[i].vZ[j].Id,"DRIVER2A")+1,mLIGHT TYPE);
          // 17=T5 Fluorescent - triphosphor-coated - high frequency
ballast
          double                  mLI POWER DENS T5=T.Value("tLUMINOUS-
EFFICACY",P.Vint(vHVAC[i].vZ[j].Id,"DRIVER2A")+1,17);
          mLI_ENERGY_WT_PW+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*(mLI POWER DENS/mLI POWER
 DENS T5)*Fuel Price Weighting; //kWh
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2_IR",1,(in
t)D.Value("dELEC CODID"))*(mLI POWER DENS/mLI POWER DENS T5);
//CO2/m2
          else // all UK
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2",1,(int)D
.Value("dELEC CODID"))*(mLI POWER DENS/mLI POWER DENS T5); //CO2/m2
        }
        else {
          mLI ENERGY WT PW+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*Fuel Price Weighting;
//kWh
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2 IR",1,(in
t)D.Value("dELEC CODID")); //KHW*CO2/KWH=CO2
          else // all UK
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2",1,(int)D
.Value("dELEC CODID")); //KHW*CO2/KWH=CO2
        }
      }
    }

    //adjusted total building energy = total building MJ - whole
project lighting energy + adjusted whole project lighting energy
    double                          mQ ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mLI_ENERGY_PRICE_WEIGHTED+mLI_ENERGY_WT_PW;
    //adjusted total building CO2 = total building CO2 (excluding
renewables) - whole project lighting CO2 + adjusted whole project
lighting CO2
    double mCO2 ADJUSTED=mKG CO2 PROJECT-mLI CO2+mLI CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ ADJUSTED);
    IMPACT_CO2=BOrec.fIMPACT_CO2(D, mKG_CO2_PROJECT, mCO2_ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON_SAVING_POUND)
    //Multiply default payback by 30 and divide by TYPICAL building
lighting consumption (kWh/m2.year)
    double       mTYP_KWH_LI=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
LI")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")));
    if((mAREA!=0)&&(mTYP KWH LI!=0))
      PAYBACK=D.Value("dEPC_L5")*30.0/(mTYP_KWH_LI/mAREA);
    else
      PAYBACK=0.0;
```

```
    CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON_SAVING_POUND=BOrec.fCARBON_SAVING_POUND(D,CARBON_SAVING_NO);

    //write the EPCL5 recommendation keyword
    string EPC_XX="EPC-L5";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);

  }
  //---- END recommendation EPC-L5 ----


  //---- recommendation EPC-L2 ----

  //if recommendation EPCL2 is triggered:
  if (EPCL2) {
  //GLS to CFL
    //go through all ZONES
    for (int i=0;i<vHVAC.size();i++) {
      for (int j=0;j<vHVAC[i].vZ.size();j++) {
        //look at lamp type, if it is GLS, switch to CFL
        mLIGHT TYPE=P.Vcodid(vHVAC[i].vZ[j].Id,"LIGHT-TYPE");
        if (mLIGHT TYPE==8) { // 8=Tungsten or Halogen
          double                    mLI POWER DENS=T.Value("tLUMINOUS-
EFFICACY",P.Vint(vHVAC[i].vZ[j].Id,"DRIVER2A")+1,mLIGHT TYPE);
          // 9=Fluorescent - compact
          double               mLI POWER DENS CFL=T.Value("tLUMINOUS-
EFFICACY",P.Vint(vHVAC[i].vZ[j].Id,"DRIVER2A")+1,9);
          mLI ENERGY WT PW+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*(mLI POWER DENS/mLI POWER
 DENS CFL)*Fuel Price Weighting; //kWh
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2 IR",1,(in
t)D.Value("dELEC_CODID"))*(mLI_POWER_DENS/mLI_POWER_DENS_CFL);
//CO2/m2
          else // all UK
             mLI_CO2_WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2",1,(int)D
.Value("dELEC CODID"))*(mLI POWER DENS/mLI POWER DENS CFL); //CO2/m2
        }
        else {
          mLI ENERGY WT PW+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*Fuel_Price_Weighting;
//kWh
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2 IR",1,(in
t)D.Value("dELEC_CODID")); //KHW*CO2/KWH=CO2
          else // all UK
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2",1,(int)D
.Value("dELEC CODID")); //KHW*CO2/KWH=CO2
        }
      }
    }

    //adjusted total building energy = total building MJ - whole
project lighting energy + adjusted whole project lighting energy
    double                        mQ ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mLI_ENERGY_PRICE_WEIGHTED+mLI_ENERGY_WT_PW;
    //adjusted total building CO2 = total building CO2 (excluding
renewables) - whole project lighting CO2 + adjusted whole project
lighting CO2
```

```
    double mCO2 ADJUSTED=mKG CO2 PROJECT-mLI CO2+mLI CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ_ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D, mKG CO2 PROJECT, mCO2 ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
    //Multiply default payback by 30 and divide by TYPICAL building
lighting consumption (kWh/m2.year)
    double      mTYP KWH LI=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
LI")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")));
    if((mAREA!=0)&&(mTYP_KWH_LI!=0))
      PAYBACK=D.Value("dEPC L2")*30.0/(mTYP KWH LI/mAREA);
    else
      PAYBACK=0.0;

    CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    //write the EPCL2 recommendation keyword
    string EPC XX="EPC-L2";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);

  }
  //---- END recommendation EPC-L2 ----


  //---- recommendation EPC-L4 ----
/*
  //if recommendation EPCL4 is triggered:
  if (EPCL4) {
  //GLS to LV tungsten halogen (As per RH's advice, use power
densities of metal halides for commercial installations)
    //go through all ZONES
    for (int i=0;i<vHVAC.size();i++) {
      for (int j=0;j<vHVAC[i].vZ.size();j++) {
        //look at lamp type, if it is GLS, switch to LV tungsten
halogen
        mLIGHT_TYPE=P.Vcodid(vHVAC[i].vZ[j].Id,"LIGHT-TYPE");
        if (mLIGHT TYPE==(int)D.Value("dGLS CODID")) {
          double
mLI_POWER_DENS=T.Value("LIGHTING_FACTORS",1,P.Vcodid(vHVAC[i].vZ[j].I
d,"LIGHT-TYPE"));
          double  mLI POWER DENS LVTH=T.Value("LIGHTING FACTORS",1,6);
// codid 6 corresponds to metal halides for commercial installations
          mLI ENERGY WT PW+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*(mLI POWER DENS LVTH/mLI
POWER_DENS)*Fuel_Price_Weighting; //kWh
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only
             mLI_CO2_WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2 IR",1,(in
t)D.Value("dELEC CODID"))*(mLI POWER DENS LVTH/mLI POWER DENS);
//CO2/m2
          else // all UK
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2",1,(int)D
.Value("dELEC CODID"))*(mLI POWER DENS LVTH/mLI POWER DENS); //CO2/m2
        }
        else {
          mLI ENERGY WT PW+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*Fuel Price Weighting;
//kWh
```

```
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2 IR",1,(in
t)D.Value("dELEC_CODID")); //KHW*CO2/KWH=CO2
          else // all UK
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2",1,(int)D
.Value("dELEC CODID")); //KHW*CO2/KWH=CO2
          }
       }
    }

    //adjusted total building energy = total building MJ - whole
project lighting energy + adjusted whole project lighting energy
    double                        mQ ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mLI_ENERGY_PRICE_WEIGHTED+mLI_ENERGY_WT_PW;
    //adjusted total building CO2 = total building CO2 (excluding
renewables) - whole project lighting CO2 + adjusted whole project
lighting CO2
    double mCO2 ADJUSTED=mKG CO2 PROJECT-mLI CO2+mLI CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D, mKG CO2 PROJECT, mCO2 ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
    //Multiply default payback by 30 and divide by TYPICAL building
lighting consumption (kWh/m2.year)
    double      mTYP KWH LI=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
LI")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")));
    if((mAREA!=0)&&(mTYP KWH LI!=0))
      PAYBACK=D.Value("dEPC L4")*30.0/(mTYP KWH LI/mAREA);
    else
      PAYBACK=0.0;

    CARBON_SAVING_NO=PAYBACK*Fuel_Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    //write the EPCL2 recommendation keyword
    string EPC XX="EPC-L4";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON_SAVING_POUND);

  }
*/
  //---- END recommendation EPC-L4 ----


  //---- recommendation EPC-L7 ----

  //if recommendation EPCL7 is triggered:
  if (EPCL7) {
  //Mains to HF ballast
    //go through all ZONES
    for (int i=0;i<vHVAC.size();i++) {
      for (int j=0;j<vHVAC[i].vZ.size();j++) {
        //look at lamp type, if mains isused (low freq ballast),
switch to HF ballast
       mLIGHT TYPE=P.Vcodid(vHVAC[i].vZ[j].Id,"LIGHT-TYPE");
       if (mLIGHT_TYPE==11) { // 11=T8 Fluorescent - halophosphate -
low frequency ballast
         double                    mLI POWER DENS=T.Value("tLUMINOUS-
EFFICACY",P.Vint(vHVAC[i].vZ[j].Id,"DRIVER2A")+1,mLIGHT_TYPE);
```

```
          // 13=T8 Fluorescent - triphosphor - high frequency ballast
          double                mLI POWER DENS HF=T.Value("tLUMINOUS-
EFFICACY",P.Vint(vHVAC[i].vZ[j].Id,"DRIVER2A")+1,13);
          mLI ENERGY WT PW+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*(mLI_POWER_DENS/mLI_POWER
 DENS HF)*Fuel Price Weighting; //kWh
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2 IR",1,(in
t)D.Value("dELEC_CODID"))*(mLI_POWER_DENS/mLI_POWER_DENS_HF);
//CO2/m2
          else // all UK
             mLI_CO2_WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2",1,(int)D
.Value("dELEC CODID"))*(mLI POWER DENS/mLI POWER DENS HF); //CO2/m2
        }
        else {
          mLI ENERGY WT PW+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*Fuel_Price_Weighting;
//kWh
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2 IR",1,(in
t)D.Value("dELEC CODID")); //KHW*CO2/KWH=CO2
          else // all UK
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2",1,(int)D
.Value("dELEC CODID")); //KHW*CO2/KWH=CO2
          }
       }
    }

    //adjusted total building energy = total building MJ - whole
project lighting energy + adjusted whole project lighting energy
    double                        mQ ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mLI ENERGY PRICE WEIGHTED+mLI ENERGY WT PW;
    //adjusted total building CO2 = total building CO2 (excluding
renewables) - whole project lighting CO2 + adjusted whole project
lighting CO2
    double mCO2_ADJUSTED=mKG_CO2_PROJECT-mLI_CO2+mLI_CO2_WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ ADJUSTED);
    IMPACT CO2=BOrec.fIMPACT CO2(D, mKG CO2 PROJECT, mCO2 ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
    //Multiply default payback by 30 and divide by TYPICAL building
lighting consumption (kWh/m2.year)
    double       mTYP KWH LI=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
LI")*(2.0/T.Value("t2010_TYP",1,P.Vcodid(cId,"EPC-TYPE")));
    if((mAREA!=0)&&(mTYP KWH LI!=0))
      PAYBACK=D.Value("dEPC L7")*30.0/(mTYP KWH LI/mAREA);
    else
      PAYBACK=0.0;

    CARBON_SAVING_NO=PAYBACK*Fuel_Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    //write the EPCL7 recommendation keyword
    string EPC XX="EPC-L7";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON_SAVING_POUND);
```

```
  }
  //---- END recommendation EPC-L7 ----


  //---- recommendation EPC-L3 ----

  //if recommendation EPCL3 is triggered:
  if (EPCL3) {
  //HP mercury to SON replacements
    //go through all ZONES
    for (int i=0;i<vHVAC.size();i++) {
      for (int j=0;j<vHVAC[i].vZ.size();j++) {
        //look at lamp type, if it is HP mercury, switch to SON (HP
sodium)
        mLIGHT TYPE=P.Vcodid(vHVAC[i].vZ[j].Id,"LIGHT-TYPE");
        if (mLIGHT_TYPE==15) { // 15=High Pressure Mercury
          double                    mLI POWER DENS=T.Value("tLUMINOUS-
EFFICACY",P.Vint(vHVAC[i].vZ[j].Id,"DRIVER2A")+1,mLIGHT TYPE);
          // 16=High Pressure Sodium
          double                 mLI POWER DENS SON=T.Value("tLUMINOUS-
EFFICACY",P.Vint(vHVAC[i].vZ[j].Id,"DRIVER2A")+1,16);
          mLI ENERGY WT PW+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*(mLI POWER DENS/mLI POWER
 DENS SON)*Fuel Price Weighting; //kWh
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2 IR",1,(in
t)D.Value("dELEC CODID"))*(mLI POWER DENS/mLI POWER DENS SON);
//CO2/m2
          else // all UK
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2",1,(int)D
.Value("dELEC CODID"))*(mLI POWER DENS/mLI POWER DENS SON); //CO2/m2
        }
        else {
          mLI ENERGY WT PW+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*Fuel_Price_Weighting;
//kWh
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2 IR",1,(in
t)D.Value("dELEC_CODID")); //KHW*CO2/KWH=CO2
          else // all UK
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2",1,(int)D
.Value("dELEC CODID")); //KHW*CO2/KWH=CO2
        }
      }
    }

    //adjusted total building energy = total building MJ - whole
project lighting energy + adjusted whole project lighting energy
    double                        mQ ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mLI_ENERGY_PRICE_WEIGHTED+mLI_ENERGY_WT_PW;
    //adjusted total building CO2 = total building CO2 (excluding
renewables) - whole project lighting CO2 + adjusted whole project
lighting CO2
    double mCO2 ADJUSTED=mKG CO2 PROJECT-mLI CO2+mLI CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ ADJUSTED);
    IMPACT_CO2=BOrec.fIMPACT_CO2(D, mKG_CO2_PROJECT, mCO2_ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON_SAVING_POUND)
```

```
    //Multiply default payback by 30 and divide by TYPICAL building
lighting consumption (kWh/m2.year)
    double      mTYP KWH LI=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
LI")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")));
    if((mAREA!=0)&&(mTYP_KWH_LI!=0))
      PAYBACK=D.Value("dEPC L3")*30.0/(mTYP KWH LI/mAREA);
    else
      PAYBACK=0.0;

    CARBON SAVING NO=PAYBACK*Fuel Payback;
    CARBON_SAVING_POUND=1;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    //write the EPCL3 recommendation keyword
    string EPC XX="EPC-L3";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON_SAVING_POUND);

  }
  //---- END recommendation EPC-L3 ----


  //---- recommendation EPC-L6 ----

  //if recommendation EPCL6 is triggered:
  if (EPCL6) {
  //HP mercury to SON
    //go through all ZONES
    for (int i=0;i<vHVAC.size();i++) {
      for (int j=0;j<vHVAC[i].vZ.size();j++) {
        //look at lamp type, if it is HP mercury, switch to SON (HP
sodium)
        mLIGHT TYPE=P.Vcodid(vHVAC[i].vZ[j].Id,"LIGHT-TYPE");
        if (mLIGHT TYPE==15) { // 15=High Pressure Mercury
          double                    mLI POWER DENS=T.Value("tLUMINOUS-
EFFICACY",P.Vint(vHVAC[i].vZ[j].Id,"DRIVER2A")+1,mLIGHT_TYPE);
          // 16=High Pressure Sodium
          double                mLI POWER DENS SON=T.Value("tLUMINOUS-
EFFICACY",P.Vint(vHVAC[i].vZ[j].Id,"DRIVER2A")+1,16);
          mLI ENERGY WT PW+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*(mLI POWER DENS/mLI POWER
_DENS_SON)*Fuel_Price_Weighting; //kWh
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only
             mLI_CO2_WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2 IR",1,(in
t)D.Value("dELEC CODID"))*(mLI POWER DENS/mLI POWER DENS SON);
//CO2/m2
          else // all UK
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2",1,(int)D
.Value("dELEC CODID"))*(mLI POWER DENS/mLI POWER DENS SON); //CO2/m2
        }
        else {
          mLI ENERGY WT PW+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*Fuel Price Weighting;
//kWh
          if((P.Vcodid(cId,"BR-TYPE")==5)    ||    (P.Vcodid(cId,"EPC-
TYPE")==6))  //Republic of Ireland only
             mLI_CO2_WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2 IR",1,(in
t)D.Value("dELEC CODID")); //KHW*CO2/KWH=CO2
          else // all UK
             mLI CO2 WEIGHTED+=P.Vnum(vHVAC[i].vZ[j].Id,"KWH-
LI")*P.Vnum(vHVAC[i].vZ[j].Id,"MULTIPLIER")*T.Value("tKGCO2",1,(int)D
.Value("dELEC_CODID")); //KHW*CO2/KWH=CO2
```

```
        }
      }
    }

    //adjusted total building energy = total building MJ - whole
project lighting energy + adjusted whole project lighting energy
    double                          mQ ADJUSTED=mQ TOTAL PRICE WEIGHTED-
mLI_ENERGY_PRICE_WEIGHTED+mLI_ENERGY_WT_PW;
    //adjusted total building CO2 = total building CO2 (excluding
renewables) - whole project lighting CO2 + adjusted whole project
lighting CO2
    double mCO2 ADJUSTED=mKG CO2 PROJECT-mLI CO2+mLI CO2 WEIGHTED;

    //assess energy and CO2 impact
    IMPACT ENERGY=BOrec.fIMPACT ENERGY(D,     mQ TOTAL PRICE WEIGHTED,
mQ ADJUSTED);
    IMPACT_CO2=BOrec.fIMPACT_CO2(D, mKG_CO2_PROJECT, mCO2_ADJUSTED);

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON_SAVING_POUND)
    //Multiply default payback by 30 and divide by TYPICAL building
lighting consumption (kWh/m2.year)
    double       mTYP KWH LI=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
LI")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")));
    if((mAREA!=0)&&(mTYP KWH LI!=0))
      PAYBACK=D.Value("dEPC L6")*30.0/(mTYP KWH LI/mAREA);
    else
      PAYBACK=0.0;

    CARBON SAVING NO=PAYBACK*Fuel Payback;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    //write the EPCL6 recommendation keyword
    string EPC XX="EPC-L6";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON_SAVING_POUND);

  }
  //---- END recommendation EPC-L6 ----

}


void       PROJECT::ENVELOPE Recommendation(PROJECTDefaults&       D,
PROJECTTables& T, PROJECT& Notional) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"PROJECT::ENVELOPE Recommendation";
/*
  EPC recommendations function

  This function:
     - assesses the highest level of overheating risk for any any
zone in the building
     - assesses the impact of recommendations for envelope

    Project level recommendations for ENVELOPE are:

         EPC-E1 - insulate floor
           -to assess impact, use table tRec Pot Impact applied to
heating energy.
              rows are categories for percentage heat CO2 or energy
category (1=more than 20%, 2=5% to 20%, 3=5% or less)
              columns are overall carbon or energy efficiency for
end-use (1=GOOD, 2=FAIR, 3=POOR)
              values in table cells are (1=HIGH/2=MEDIUM/3=LOW)
         EPC-E2 - insulate flat roof
```

```
          -to assess impact, use table tRec Pot Impact applied to
heating energy.
              rows are categories for percentage heat CO2 or energy
category (1=more than 20%, 2=5% to 20%, 3=5% or less)
              columns are overall carbon or energy efficiency for
end-use (1=GOOD, 2=FAIR, 3=POOR)
              values in table cells are (1=HIGH/2=MEDIUM/3=LOW)
         EPC-E3   - insulate solid walls
           -to assess impact, use table tRec Pot Impact applied to
heating energy.
              rows are categories for percentage heat CO2 or energy
category (1=more than 20%, 2=5% to 20%, 3=5% or less)
              columns are overall carbon or energy efficiency for
end-use (1=GOOD, 2=FAIR, 3=POOR)
              values in table cells are (1=HIGH/2=MEDIUM/3=LOW)
         EPC-E4   - cavity wall insulation
            -to assess impact, use table tRec Pot Impact applied to
heating energy.
              rows are categories for percentage heat CO2 or energy
category (1=more than 20%, 2=5% to 20%, 3=5% or less)
              columns are overall carbon or energy efficiency for
end-use (1=GOOD, 2=FAIR, 3=POOR)
              values in table cells are (1=HIGH/2=MEDIUM/3=LOW)
         EPC-E5   - secondary glazing
           -to assess impact, use table tRec Pot Impact applied to
heating energy.
              rows are categories for percentage heat CO2 or energy
category (1=more than 20%, 2=5% to 20%, 3=5% or less)
              columns are overall carbon or energy efficiency for
end-use (1=GOOD, 2=FAIR, 3=POOR)
              values in table cells are (1=HIGH/2=MEDIUM/3=LOW)
         EPC-E6   - insulate loft
           -to assess impact, use table tRec Pot Impact applied to
heating energy.
              rows are categories for percentage heat CO2 or energy
category (1=more than 20%, 2=5% to 20%, 3=5% or less)
              columns are overall carbon or energy efficiency for
end-use (1=GOOD, 2=FAIR, 3=POOR)
              values in table cells are (1=HIGH/2=MEDIUM/3=LOW)
         EPC-E7   - pressure test
           -to assess impact, use table tRec Pot Impact applied to
heating energy.
              rows are categories for percentage heat CO2 or energy
category (1=more than 20%, 2=5% to 20%, 3=5% or less)
              columns are overall carbon or energy efficiency for
end-use (1=GOOD, 2=FAIR, 3=POOR)
              values in table cells are (1=HIGH/2=MEDIUM/3=LOW)
         EPC-E8   - improve glazing
           -to assess impact, use table tRec Pot Impact applied to
heating energy.
              rows are categories for percentage heat CO2 or energy
category (1=more than 20%, 2=5% to 20%, 3=5% or less)
              columns are overall carbon or energy efficiency for
end-use (1=GOOD, 2=FAIR, 3=POOR)
              values in table cells are (1=HIGH/2=MEDIUM/3=LOW)
         EPC-V1   - solar control measures
           -to assess impact, use table tRec Pot Impact applied to
heating energy.
              rows are categories for percentage heat CO2 or energy
category (1=more than 20%, 2=5% to 20%, 3=5% or less)
              columns are overall carbon or energy efficiency for
end-use (1=GOOD, 2=FAIR, 3=POOR)
              values in table cells are (1=HIGH/2=MEDIUM/3=LOW)
*/


  //variables to store if recommendation is triggered
  bool EPCE1=0;        // insulate floor
  bool EPCE2=0;        // insulate flat roof
```

```
  bool EPCE3=0;        // insulate solid walls
  bool EPCE4=0;        // insulate cavity walls
  bool EPCE5=0;        // consider secondary glazing
  bool EPCE6=0;        // insulate loft
  bool EPCE7=0;        // pressure test
  bool EPCE8=0;        // improve glazing

  bool EPCV1=0;        // improve solar control or consider cooling

  // The overheating recommendation is now triggered if the solar
gain limit is exceeded in any zone
  string overheat="NO";

  for (int i=0;i<vHVAC.size();i++) {
    for (int j=0;j<vHVAC[i].vZ.size();j++) {
      for (int k=0;k<vHVAC[i].vZ[j].vW.size();k++) {
        switch(P.Vcodid(vHVAC[i].vZ[j].vW[k].Id,"TYPE-ENV")) {
          case 1: {
          //wall
            //if    wall    is    adjacent    to    exterior/unheated
space/underground/strongly ventilated space, i.e. not adjacent to
conditioned space or same space
            if   (   (P.Vcodid(vHVAC[i].vZ[j].vW[k].Id,"TYPE")!=4)   &&
(P.Vcodid(vHVAC[i].vZ[j].vW[k].Id,"TYPE")!=6) ) {
              switch(P.Vcodid(vHVAC[i].vZ[j].vW[k].Id,"CAVITY-WALL"))
{
                case 1: {
                  //cavity wall
                  //if U value > 1.0, trigger recommendation EPC-E4
                  if   (P.Vnum(vHVAC[i].vZ[j].vW[k].Id,"U-VALUE")>1.0)
EPCE4=1;   // EPCE4: introduce cavity wall insulation
                }
                break;
                case 2: {
                  //solid wall
                  //if U value > 1.0, trigger recommendation EPC-E3
                  if   (P.Vnum(vHVAC[i].vZ[j].vW[k].Id,"U-VALUE")>1.0)
EPCE3=1;  // EPCE3: improve internal wall insulation
                  //Assess impact - Use Table 1 applied to heating
energy and carbon
                }
                break;
              }
            }
          }
          break;
          case 2: {
          //Roof
            //Identify any pitched roofs with lofts
            //check roof is not adjacent to conditioned space or same
space (it shouldn't be but checking anyway)
            if   (   (P.Vcodid(vHVAC[i].vZ[j].vW[k].Id,"TYPE")!=4)   &&
(P.Vcodid(vHVAC[i].vZ[j].vW[k].Id,"TYPE")!=6) ) {
              //next check if roof is flat or pitched
              if((P.Vcodid(vHVAC[i].vZ[j].vW[k].Id,"ROOF-PITCH"))<=2)
{ // 2 = 10 degrees, i.e., flat roof
                // If U value > 1.0, trigger recommendation EPC-E2
                if     (P.Vnum(vHVAC[i].vZ[j].vW[k].Id,"U-VALUE")>1.0)
EPCE2=1; // EPCE2: improve roof insulation
              } // end if
              else { // pitched roof
                  //check pitched roof has a loft
                  if         (P.Vcodid(vHVAC[i].vZ[j].vW[k].Id,"PITCH-
LOFT")==1)
                    //has a loft
                    //If   any   have   U   value   >   1.0,   trigger
recommendation EPC-E6
                    if             (P.Vnum(vHVAC[i].vZ[j].vW[k].Id,"U-
VALUE")>1.0) EPCE6=1;   // EPCE6: improve loft insulation
```

```
              } // end else
            } // end if
          } // end case 2
          break;
          case 3: {
          //Floor or Ceiling
            //if    floor    is    adjacent    to    exterior/unheated
space/underground/strongly ventilated space, i.e. not adjacent to
conditioned space or same space
            if   (   (P.Vcodid(vHVAC[i].vZ[j].vW[k].Id,"TYPE")!=4)   &&
(P.Vcodid(vHVAC[i].vZ[j].vW[k].Id,"TYPE")!=6) ) {
            //If U value > 1.0 trigger recommendation EPC-E1
              if       (P.Vnum(vHVAC[i].vZ[j].vW[k].Id,"U-VALUE")>1.0)
EPCE1=1;          // EPCE1:  improve floor insulation
            }
          }
          break;
        }   //end of TYPE-ENV switch
        //windows
        for (int l=0;l<vHVAC[i].vZ[j].vW[k].vW.size();l++) {
          //windows   -   exclude   display   windows   from   u-value
recommendation
          //If  any  have  U  value  >  D.Value("REC DEF24")  (assumed
single glazed), trigger recommendations EPC-E5 and EPC-E8
          if                  (P.Vnum(vHVAC[i].vZ[j].vW[k].vW[l].Id,"U-
VALUE")>D.Value("REC DEF24")) {
            EPCE5=1;    // EPCE5: consider secondary glazing
            EPCE8=1;    // EPCE8: improve glazing
          }
        }
      }     //end of loop through envelope elements

      //for zones: check airtightness:
      //If permeability > 14, trigger recommendation EPC-E7
      if  (P.Vnum(vHVAC[i].vZ[j].Id,"Q50-INF")>14)  EPCE7=1;  //  EPCE7:
treat air leakeage


      //Overheating:
      //Check whether the solar gain limit is exceeded in any zone,
in which case, trigger the recommendation on
      // overheating
      //  Because  the  modified  text  of  the  overheating  calculation
specifically refers to ADL2A, it is only generated
      // if the calculation is an EPC for England or EPC for Wales.
      if  (  (overheat=="NO")  &&  ((P.Vcodid(cId,"EPC-TYPE")==2)  ||
(P.Vcodid(cId,"EPC-TYPE")==3)) )  // EPC in England or Wales
      {
        if    (P.Vcodid(vHVAC[i].vZ[j].Id,"SOLAR-GAIN-OVER-LIMIT")==1)
// 1=YES
          overheat="YES";
      } // end if

      {
      ostringstream res; res << overheat;
      P.ModKey(RP.Id,"SOLAR-GAIN-OVER-LIMIT",res.str());
      }

    }    //end of loop through zones
  }    //end of loop through HVACs


  // Trigger overheating recommendation if the parameter overheat was
changed from the default of "No risk".
  if (overheat=="YES")   EPCV1=1;


  //assess impact of recommendations
```

```
  double mFUEL CARBON INTENSITY=0.0;
  double Fuel Payback=0.0;
  //stores impact of recommendation on CO2 (1=HIGH/2=MEDIUM/3=LOW)
  int IMPACT CO2=1;
  //stores impact of recommendation on energy (1=HIGH/2=MEDIUM/3=LOW)
  int IMPACT ENERGY=1;

  //payback period
  double PAYBACK=0;
  //carbon saving per pound, good, fair or poor
  int CARBON_SAVING_POUND=1;
  double CARBON SAVING NO;

  //% of building energy used for heating (includes DHW where this is
supplied by HVAC heat generator)
  double mPC ENERGY HEAT=P.Vnum(RP.Id,"HEAT-PERCENT-ENERGY");
  //% of CO2 attributed to heating (includes DHW where this is
supplied by HVAC heat generator)
  double mPC CO2 HEAT=P.Vnum(RP.Id,"HEAT-PERCENT-CO2");

  //determine which category your percentage heat CO2 and percentage
heat energy falls into:
  //1:  20% + energy or CO2, 2: 5% to 20% energy or CO2, 3: less than
5% - energy or CO2
  int row ENERGY;
  int row CO2;

  if (mPC ENERGY HEAT>20.0) row ENERGY=1;
  else if (mPC ENERGY HEAT>5.0) row ENERGY=2;
       else row ENERGY=3;

  if (mPC CO2 HEAT>20.0) row CO2=1;
  else if (mPC CO2 HEAT>5.0) row CO2=2;
       else row CO2=3;

  int mPERF HEAT ENERGY=P.Vcodid(RP.Id,"PERF-HEAT-ENERGY");
  int mPERF HEAT CO2=P.Vcodid(RP.Id,"PERF-HEAT-CO2");

  //if not applicable, set performance to GOOD but write warning to
log
  if (mPERF HEAT ENERGY==4) {
    mPERF_HEAT_ENERGY==1;
    { ostringstream logtext; logtext << endl << endl << "WARNING,
building heating energy performance is 0"; SendLog(logtext.str()); }
  }
  //if not applicable, set performance to GOOD but write warning to
log
  if (mPERF_HEAT_CO2==4) {
    mPERF HEAT CO2==1;
    { ostringstream logtext; logtext << endl << endl << "WARNING,
building heating energy performance is 0"; SendLog(logtext.str()); }
  }

  Fuel_Payback=P.Vnum(RP.Id,"F-PAYBACK-HEAT");

  //To assess likely impact of recommendations on energy and CO2
  //Use Table 1 applied to heating energy or CO2
  //  table tRec Pot Impact:
  //    - rows are percentage heat CO2 or energy category
  //    - columns are overall carbon or energy efficiency for end-use
(1=GOOD, 2=FAIR, 3=POOR)
  //store impact of recommendation on CO2 (1=HIGH/2=MEDIUM/3=LOW)
  IMPACT CO2=(int)T.Value("tRec Pot Impact",row CO2,mPERF HEAT CO2);
// to be used for all envelope recommendations
  //store impact of recommendation on energy (1=HIGH/2=MEDIUM/3=LOW)

IMPACT ENERGY=(int)T.Value("tRec Pot Impact",row ENERGY,mPERF HEAT EN
ERGY);  // to be used for all envelope recommendations
```

```
  double mAREA=P.Vnum(gId,"AREA");
  double       mTYP KWH HEAT=Notional.P.Vnum(Notional.dGEN[0].Id,"KWH-
HEAT")*(2.0/T.Value("t2010 TYP",1,P.Vcodid(cId,"EPC-TYPE")));


  //---- recommendation EPC-E4 ----

  if (EPCE4) {
    //cavity walls with U value > 1.0

    //calculate  payback  (PAYBACK)  and  Carbon  saving  per  pound
(CARBON SAVING POUND)
    //Multiply payback by 140 and divide by TYPICAL building heating
consumption (kWh/m2.year)

    if((mAREA!=0)&&(mTYP_KWH_HEAT!=0))
      PAYBACK=D.Value("dEPC E4")*140.0/(mTYP KWH HEAT/mAREA);
    else
      PAYBACK=0.0;

    CARBON SAVING NO=PAYBACK*Fuel Payback;
    CARBON SAVING POUND=1;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    //write the EPCE4 recommendation keyword
    string EPC XX="EPC-E4";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);

  }
  //---- END recommendation EPC-E4 ----


  //---- recommendation EPC-E1 ----

  if (EPCE1) {
    //floors with U value > 1.0

    //calculate  payback  (PAYBACK)  and  Carbon  saving  per  pound
(CARBON SAVING POUND)
    //Multiply payback by 140 and divide by TYPICAL building heating
consumption (kWh/m2.year)

    if((mAREA!=0)&&(mTYP_KWH_HEAT!=0))
      PAYBACK=D.Value("dEPC E1")*140.0/(mTYP KWH HEAT/mAREA);
    else
      PAYBACK=0.0;

    CARBON SAVING NO=PAYBACK*Fuel Payback;
    CARBON_SAVING_POUND=1;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    //write the EPCE1 recommendation keyword
    string EPC XX="EPC-E1";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON_SAVING_POUND);

  }
  //---- END recommendation EPC-E1 ----


  //---- recommendation EPC-E2 ----
```

```
  if (EPCE2) {
    //flat roofs with U value > 1.0

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
    //Multiply payback by 140 and divide by TYPICAL building heating
consumption (kWh/m2.year)

    if((mAREA!=0)&&(mTYP KWH HEAT!=0))
      PAYBACK=D.Value("dEPC_E2")*140.0/(mTYP_KWH_HEAT/mAREA);
    else
      PAYBACK=0.0;

    CARBON SAVING NO=PAYBACK*Fuel Payback;
    CARBON SAVING POUND=1;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON_SAVING_POUND=BOrec.fCARBON_SAVING_POUND(D,CARBON_SAVING_NO);

    //write the EPCE2 recommendation keyword
    string EPC XX="EPC-E2";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);

  }
  //---- END recommendation EPC-E2 ----


  //---- recommendation EPC-E3 ----
  if (EPCE3) {
    //solid walls with U value > 1.0

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
    //Multiply payback by 140 and divide by TYPICAL building heating
consumption (kWh/m2.year)

    if((mAREA!=0)&&(mTYP KWH HEAT!=0))
      PAYBACK=D.Value("dEPC E3")*140.0/(mTYP KWH HEAT/mAREA);
    else
      PAYBACK=0.0;

    CARBON_SAVING_NO=PAYBACK*Fuel_Payback;
    CARBON SAVING POUND=1;

    //determine CARBON_SAVING_POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    //write the EPCE3 recommendation keyword
    string EPC XX="EPC-E3";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);

  }
  //---- END recommendation EPC-E3 ----


  //---- recommendation EPC-E6 ----

  if (EPCE6) {
    //pitched roofs with lofts with U value > 1.0

    //calculate payback (PAYBACK) and Carbon saving per pound
(CARBON SAVING POUND)
    //Multiply payback by 140 and divide by TYPICAL building heating
consumption (kWh/m2.year)
```

```
    if((mAREA!=0)&&(mTYP KWH HEAT!=0))
      PAYBACK=D.Value("dEPC E6")*140.0/(mTYP KWH HEAT/mAREA);
    else
      PAYBACK=0.0;

    CARBON SAVING NO=PAYBACK*Fuel Payback;
    CARBON_SAVING_POUND=1;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    //write the EPCE6 recommendation keyword
    string EPC XX="EPC-E6";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON_SAVING_POUND);

  }
  //---- END recommendation EPC-E6 ----


  //---- recommendation EPC-E7 ----

  if (EPCE7) {
    //Airtightness - air permeability > 14

    //calculate  payback  (PAYBACK)  and  Carbon  saving  per  pound
(CARBON SAVING POUND)
    //Multiply payback by 140 and divide by TYPICAL building heating
consumption (kWh/m2.year)

    if((mAREA!=0)&&(mTYP KWH HEAT!=0))
      PAYBACK=D.Value("dEPC E7")*140.0/(mTYP KWH HEAT/mAREA);
    else
      PAYBACK=0.0;

    CARBON SAVING NO=PAYBACK*Fuel Payback;
    CARBON_SAVING_POUND=1;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    //write the EPCE7 recommendation keyword
    string EPC XX="EPC-E7";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON_SAVING_POUND);

  }
  //---- END recommendation EPC-E7 ----


  //---- recommendation EPC-E5 ----

  if (EPCE5) {
    //windows with U value > 3.0

    //calculate  payback  (PAYBACK)  and  Carbon  saving  per  pound
(CARBON_SAVING_POUND)
    //Multiply payback by 140 and divide by TYPICAL building heating
consumption (kWh/m2.year)

    if((mAREA!=0)&&(mTYP KWH HEAT!=0))
      PAYBACK=D.Value("dEPC E5")*140.0/(mTYP KWH HEAT/mAREA);
    else
      PAYBACK=0.0;

    CARBON_SAVING_NO=PAYBACK*Fuel_Payback;
```

```
    CARBON SAVING POUND=1;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON_SAVING_POUND=BOrec.fCARBON_SAVING_POUND(D,CARBON_SAVING_NO);

    //write the EPCE5 recommendation keyword
    string EPC_XX="EPC-E5";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);

  }
  //---- END recommendation EPC-E5 ----


  //---- recommendation EPC-E8 ----

  if (EPCE8) {
    //windows with U value > 3.5

    //calculate  payback  (PAYBACK)  and  Carbon  saving  per  pound
(CARBON SAVING POUND)
    //Multiply payback by 140 and divide by TYPICAL building heating
consumption (kWh/m2.year)

    if((mAREA!=0)&&(mTYP KWH HEAT!=0))
      PAYBACK=D.Value("dEPC E8")*140.0/(mTYP KWH HEAT/mAREA);
    else
      PAYBACK=0.0;

    CARBON SAVING NO=PAYBACK*Fuel Payback;
    CARBON SAVING POUND=1;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    //write the EPCE8 recommendation keyword
    string EPC_XX="EPC-E8";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);

  }
  //---- END recommendation EPC-E8 ----


  //---- recommendation EPC-V1 ----

  //Overheating risk
  if (EPCV1) {

    double PAYBACK=D.Value("dEPC V1");

    double CARBON_SAVING_NO=PAYBACK;
    int CARBON SAVING POUND;

    //Set carbon saving per pound as POOR (since the carbon savings
will be zero or negative)
    CARBON SAVING POUND=3;

    //impact is set to medium (based on email from RH)
    int IMPACT ENERGY=2;
    int IMPACT_CO2=2;

    //write the EPCR3 recommendation keyword
    string EPC_XX="EPC-V1";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);
```

```
  }
  //---- END recommendation EPC-V1 ----


}  //end of PROJECT::ENVELOPE_Recommendation(...)



void  PROJECT::SES Recommendation(PROJECTDefaults&  D,  PROJECTTables&
T, PROJECT& Notional) {
  if(func_report)     cout     <<     endl     <<     "          "     <<
"PROJECT::SES Recommendation";
/*
  EPC recommendations function

  This function:
     -  assesses  the  impact  of  recommendations  for  solar  energy
systems (SES)

    Project level recommendations for SES are:

         EPC-R3 - install solar thermal water heating
           -impact is always low

*/

  double Fuel Payback=0.0;
    //total energy and CO2 for all DHW generators in project :
  double mQ PRIM TOTAL=0.0;

  //determine whether there is a SES anywhere in the project
  //if EPCR3=1, recommendation is triggered,
  bool EPCR3=1;

  for (int i=0;i<vDHG.size();i++) {
    //if there is a SES, do not trigger recommendation
    if (vDHG[i].vS.size()>0)
       EPCR3=0;
    mQ_PRIM_TOTAL+=P.Vnum(vDHG[i].Id,"Q-PRIM");
    //fuel payback adjustment, weight with energy - used to calculate
carbon saved/pound spent

Fuel Payback+=T.Value("tFuel Payback",1,P.Vcodid(vDHG[i].Id,"FUEL-
TYPE"))*P.Vnum(vDHG[i].Id,"Q-PRIM");
  }

  // Fuel payback weighted by fuel types used in all DHW generators
in the building used for calculating impact of SES recommendation
  Fuel Payback/=mQ PRIM TOTAL;


  //---- recommendation EPC-R3 ----

  //no SES:
  if (EPCR3) {
    //calculate  payback  (PAYBACK)  and  Carbon  saving  per  pound
(CARBON_SAVING_POUND)

    double PAYBACK=D.Value("dEPC R3");

    double CARBON SAVING NO=PAYBACK*Fuel Payback;
    int CARBON SAVING POUND;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON_SAVING_POUND=BOrec.fCARBON_SAVING_POUND(D,CARBON_SAVING_NO);

    //impact is always low
    int IMPACT_ENERGY=3;
```

```
    int IMPACT CO2=3;

    //write the EPCR3 recommendation keyword
    string EPC XX="EPC-R3";
    Write Rec Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);

  }
  //---- END recommendation EPC-R3 ----

}


void  PROJECT::PVS Recommendation(PROJECTDefaults&  D,  PROJECTTables&
T, PROJECT& Notional) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"PROJECT::PVS_Recommendation";
/*
  EPC recommendations function

  This function:
     - assesses the impact of recommendations for photvoltaic systems
(PVS)

    Project level recommendations for PVS are:

         EPC-R4 - install PV system
           -impact is always low

*/


  //---- recommendation EPC-R4 ----

  if (vPV.size()==0) {
  //trigger recommendation EPCR4
    //calculate  payback  (PAYBACK)  and  Carbon  saving  per  pound
(CARBON SAVING POUND)

    double PAYBACK=D.Value("dEPC R4");

    //fuel payback adjustment, weight with energy - used to calculate
carbon saved/pound spent
    double
Fuel_Payback=T.Value("tFuel_Payback",1,(int)D.Value("dELEC_CODID"));

    double CARBON SAVING NO=PAYBACK*Fuel Payback;
    int CARBON_SAVING_POUND;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    //impact is always low
    int IMPACT ENERGY=3;
    int IMPACT CO2=3;

    //write the EPCR4 recommendation keyword
    string EPC XX="EPC-R4";
    Write_Rec_Keyword(D,   T,   EPC_XX,   IMPACT_ENERGY,   IMPACT_CO2,
PAYBACK, CARBON SAVING POUND);

  }
  //---- END recommendation EPC-R4 ----

}


void  PROJECT::WIND Recommendation(PROJECTDefaults&  D,  PROJECTTables&
```

```
T, PROJECT& Notional) {
  if(func report)    cout    <<    endl    <<    "         "    <<
"PROJECT::WIND Recommendation";
/*
  EPC recommendations function

  This function:
     - assesses the impact of recommendations for wind turbines

    Project level recommendations for Wind are:

         EPC-R2 - install wind turbine
           -impact is always low

*/

  //---- recommendation EPC-R2 ------

  if (vWG.size()==0) {
    //trigger recommendation EPCR2

    //calculate  payback  (PAYBACK)  and  Carbon  saving  per  pound
(CARBON SAVING POUND)

    double PAYBACK=D.Value("dEPC R2");

    //fuel payback adjustment, weight with energy - used to calculate
carbon saved/pound spent
    double
Fuel Payback=T.Value("tFuel Payback",1,(int)D.Value("dELEC CODID"));

    double CARBON SAVING NO=PAYBACK*Fuel Payback;
    int CARBON SAVING POUND;

    //determine CARBON SAVING POUND value (GOOD, FAIR, POOR)

CARBON SAVING POUND=BOrec.fCARBON SAVING POUND(D,CARBON SAVING NO);

    //impact is always low
    int IMPACT ENERGY=3;
    int IMPACT CO2=3;

    //write the EPCR2 recommendation keyword
    string EPC XX="EPC-R2";
    Write_Rec_Keyword(D,   T,   EPC XX,   IMPACT ENERGY,   IMPACT CO2,
PAYBACK, CARBON SAVING POUND);

  }
  //---- END recommendation EPC-R2 ------

}


void  PROJECT::CHP Recommendation(PROJECTDefaults&  D,  PROJECTTables&
T, PROJECT& Notional) {
  if(func report)    cout    <<    endl    <<    "         "    <<
"PROJECT::CHP_Recommendation";
/*
  EPC recommendations function

  This function:
     - calculates the percentage of total CO2 emissions produced
(rather than displaced) by these systems in project
*/

  //total building CO2 for all end uses (to be used when assessing
impact of recommendation)
  //and for calculating the % CO2 for this end use:
  //this keyword is the bulding CO2 if there were no renewables:
```

```
  double mKG_CO2_PROJECT=P.Vnum(gId,"KG-CO2-EXCL-RENEWABLES");
  //percentage of CO2 attributable to CHP (produced rather than
displaced)
  double CHP_PERCENT_CO2=0.0;

  if (vCHP.size()>0)
    if (mKG_CO2_PROJECT!=0.0)
      CHP_PERCENT_CO2=(P.Vnum(vCHP[0].Id,"KG-
CO2")/mKG_CO2_PROJECT)*100.0;

  {
   ostringstream res; res << CHP_PERCENT_CO2;
   P.ModKey(RP.Id,"CHP-PERCENT-CO2",res.str());
  }

}

// End Object: RECOMMENDATION
```

SES

The SES command defines the performance parameters of a solar thermal energy system. A U-name is required. An SES command applies to the DHW-GENRATOR instruction preceding it and describes a solar energy system attached to that hot water system.

Example input:

```
"Solar panel 1" = SES
  AREA            = 3
  SES-ORIENTATION = South
  SES-INCLINATION = 60
  ..
```

AREA
Specifies the solar collector aperture area (collector area without the bearing construction), in m^2. This keyword is required.

SES-ORIENTATION
Specifies the orientation of the solar panel, as one of the values listed below. The default is 'South'.

North
North-East
East
South-East
South
South-West
West
North-West

SES-INCLINATION
Specifies the inclination of the solar panel in degrees from the horizontal, as one of the values listed below. The default is '60'.

90
75
60
45
30
15
0

MULTIPLIER
This keyword specifies how many solar thermal energy systems with this exact description exist in a particular hot water system. E.g., if the value of the multiplier is 3, there are 3 solar thermal energy systems, each with identical keyword values serving the hot water system. This enables repeating solar thermal energy systems to be entered quickly and easily. The default value is 1.

COLLECTOR-PAR-KNOWN
Indicates whether the solar collector performance parameters from EN 12975-2 are known by the user. It can be either YES or NO. The default is NO.

COLLECTOR-TYPE
Specifies the type of the solar collector, as one of the values listed below. The default is 'Flat Panel'. This keyword is included in the input if the SES object's COLLECTOR-PAR-KNOWN keyword is set to NO, i.e., the solar collector performance parameters from EN 12975-2 are not known by the user.

Unglazed
Flat Panel
Evacuated tube

SIGMA-ZERO
Specifies the sigma-zero (zero-loss collector efficiency) performance parameter of the solar collector from EN 12975-2. This keyword is included in the input if the SES object's COLLECTOR-PAR-KNOWN keyword is set to YES, i.e., the solar collector performance parameters from EN 12975-2 are known by the user. The default value is 0.6.

A1
Specifies the a_1 (collector heat loss coefficient, in W/m^2K) performance parameter of the solar collector from EN 12975-2. This keyword is included in the input if the SES object's COLLECTOR-PAR-KNOWN keyword is set to YES, i.e., the solar collector performance parameters from EN 12975-2 are known by the user. The default value is 20.

A2
Specifies the a_2 (temperature dependence of the collector heat loss coefficient, in W/m^2K) performance parameter of the solar collector from EN 12975-2. This keyword is included in the input if the SES object's COLLECTOR-PAR-KNOWN keyword is set to YES, i.e., the solar collector performance parameters from EN 12975-2 are known by the user. The default value is 0.

IAM
Specifies the IAM (collector Incidence Angle Modifier) performance parameter of the solar collector from EN 12975-2. This keyword is included in the input if the SES object's COLLECTOR-PAR-KNOWN keyword is set to YES, i.e., the solar collector performance parameters from EN 12975-2 are known by the user. The default value is 1.

SES-STORE-VOL
Specifies the solar collector storage volume, in litres. This is a required keyword.

SES-PRE-HEAT-TYPE
Specifies the solar thermal energy system pre-heating type, as one of the values listed below. The default is 'Combined cylinder'.

Value	Description
Separate solar cylinder	The solar thermal energy system has an independent storage vessel.
Combined cylinder	The solar thermal energy system shares the same storage vessel with the hot water system.

STORE-INSULAT-TYPE
Specifies the insulation type on the solar thermal storage vessel, as one of the values listed below. The default value is 'Uninsulated'. This keyword is included in the input if the SES object's SES-PRE-HEAT-TYPE keyword is set to 'Separate solar cylinder'.

Uninsulated
Loose jacket
Factory insulated

STORE-INSULAT-THICK
Specifies the insulation thickness on the solar thermal storage vessel, in mm. The default value is 0. This keyword is included in the input if the SES object's SES-PRE-HEAT-TYPE keyword is set to 'Separate solar cylinder'.

HEAT-EXCHANGE
Indicates whether there is a heat exchanger in the solar collector loop. It can be either YES or NO. The default is YES.

HEAT-EXCH-LOSS-KNOWN
Indicates whether the heat transfer rate of the heat exchanger in the solar collector loop is known by the user. It can be either YES or NO. The default is NO.

HEAT-EXCH-LOSS
Specifies the heat transfer rate of the heat exchanger in the solar collector loop, in W/K. The default is 0. This keyword is included in the input if the SES object's HEAT-EXCH-LOSS-KNOWN keyword is set to YES.

U-LOOP-P-KNOWN
Indicates whether the overall heat loss coefficient of all the pipes in the collector loop is known by the user. It can be either YES or NO. The default is NO.

U-LOOP-P
Specifies the overall heat loss coefficient of all the pipes in the collector loop, in W/K. The default is 0. This keyword is included in the input if the SES object's U-LOOP-P-KNOWN keyword is set to YES.

SES-CIRC-SYS
Specifies the solar thermal circulation system, as one of the values listed below. The default is 'thermosiphon system'.

thermosiphon system
forced circulation system with PV
forced circulation system with no PV

AUX-POWER-KNOWN
Indicates whether the nominal power of the solar circulation system pump is known by the user. It can be either YES or NO. The default is NO.

AUX-POWER
Specifies the nominal power of the solar circulation system pump, in Watts. The default value is 0. This keyword is included in the input if the SES object's AUX-POWER-KNOWN keyword is set to YES.

PIPE-INSULAT
Indicates whether the distribution pipes between the solar thermal energy system and the back-up hot water system are insulated. It can be either YES or NO. The default is NO. This keyword is included in the input if the SES object's SES-PRE-HEAT-TYPE keyword is set to 'Separate solar cylinder'.

c++ code

```
//: ses.cpp

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void SES::TEST() {
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test SES" << endl;
//  cout << "Fin de Test SES" << endl;
  BEEMObject::TEST();
}

// SES::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P, BOC&N,
int i)
void SES::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P, BOC
&N, int i) {
  if(func report) cout << endl << "  " << "SES::Notional  " << i;
  BObject Temp;
  Temp.Ntype=13;
  Temp.Type="SES";
  Temp.Name=P.vINP[i].Name;

  KeywordValue KV;

  {
   ostringstream res; res << 0.0;
   KV.Reset(); KV.Init(1,res.str());
```

```
   Temp.Prop.insert(make pair("AREA",KV));
  }

  N.vINP.push back(Temp);
// */(req) is the same as in actual building CD

}

// SES::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P, BOC&N,
int i)

void SES::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "SES::Calc";
  if(calc_report) cout << endl << P.vINP[Id].Name;
// its own calcs below...
  fQ RAD(D, T, P);
  fQ_SES(D, T, P);
}

void SES::fQ_RAD(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "SES::fQ RAD";
  //calculate monthly and total radiation values on solar collector

  double mQ RAD M[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  double mQ RAD TOTAL=0;

  for(int i=0;i<12;i++){

mQ RAD M[i]=T.Value(T.TValue($T033 tables,P.Vcodid(gId,"WEATHER"),i+1
),
              P.Vcodid(Id,"SES-ORIENTATION"),P.Vcodid(Id,"SES-
INCLINATION"));
    mQ RAD TOTAL+=mQ RAD M[i];
  }

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
    res << mQ_RAD_M[i] << ", ";
   res << mQ RAD M[11] << " }";
   P.ModKey(Id,"Q-RAD-M",res.str());
  }
  {
   ostringstream res; res << mQ RAD TOTAL;
   P.ModKey(Id,"Q-RAD",res.str());
  }


} // end fQ RAD


void SES::fQ SES(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "SES::fQ SES";
  // mQ_SES

  //!! Check that the type of storage is not combined cylinder with a
parent HW system with no storage defined
  if (P.Vcodid(Id,"SES-PRE-HEAT-TYPE")==2) { // 2=Combined cylinder
    if (P.Vcodid(pId,"STORE-SYSTEM")==2) { // 2=no
      { // display a warning in the log file
       ostringstream logtext;
       logtext << endl << "WARNING: Storage for SES " <<
P.vINP[Id].Name << " was changed to \"Separate solar cylinder\" as
the HW system " << P.vINP[pId].Name << " has no storage." << endl;
       SendLog(logtext.str()); WriteLog();
      }
      { // Save the updated keyword
       ostringstream res; res << "Separate solar cylinder";
       P.ModKey(Id,"SES-PRE-HEAT-TYPE",res.str());
```

```
      }
    } // end if
  } // end if

  // save the area served by the parent HW system into the SES so it
can be accessed later
  double mAREA SERVED=P.Vnum(pId,"AREA-SERVED");
  {
  ostringstream res; res << mAREA SERVED;
  P.ModKey(Id,"AREA-SERVED",res.str());
  }


  //retrieving demand for the dhw generator

  //gross demand for hot water from the parent HW system
  deque<double> mQ_DEM_M=P.Vlnum(pId,"Q-DEM-TOT");   // in MJ

  double mKWH DHW M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
  double mKWH_DHW=0.0;

  for (int i=0;i<12;i++) {
    mKWH DHW M[i]=mQ DEM M[i]/3.6;    // in kWh
    mKWH DHW+=mKWH DHW M[i];
  } // end for


  //CALCULATION OF SOLAR SYSTEM COEFFICIENTS - monthly calculation
  double mAREA=P.Vnum(Id,"AREA");
  double m DHW FRAC M[12]={ 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 };

  //CALCULATION OF COLLECTOR PARAMETERS
  double mSIGMA ZERO;
  double mA1;
  double mA2;
  double mIAM;  // INCIDENT ANGLE MODIFIER

  if(P.Vcod(Id,"COLLECTOR-PAR-KNOWN")=="YES") {
    mSIGMA_ZERO=P.Vnum(Id,"SIGMA-ZERO");   //   zero-loss   collector
efficiency
    mA1=P.Vnum(Id,"A1");                       // collector heat loss
coefficient, in W/m2K
    mA2=P.Vnum(Id,"A2");                  // temperature dependence of
the collector heat loss coefficient, in W/m2K
    mIAM=P.Vnum(Id,"IAM");                // collector Incidence Angle
Modifier, value should be taken from EN 12975-2
  } // end if
  else { //i.e, collector parameters unknown

mSIGMA ZERO=T.Value("tSES SIGMA ZERO DEFAULT",1,P.Vcodid(Id,"COLLECTO
R-TYPE"));
    mA1=T.Value("T102 SES A1",1,P.Vcodid(Id,"COLLECTOR-TYPE"));
    mA2=D.Value("dSES A2 DEFAULT");
    mIAM=T.Value("T102_SES_IAM",1,P.Vcodid(Id,"COLLECTOR-TYPE"));
  } // end else


  //CALCULATION OF U-LOOP (heat loss coeficient of the collector)
  double mU LOOP;
  double mU_LOOP_P;
  if(P.Vcod(Id,"U-LOOP-P-KNOWN")=="YES")
    mU LOOP P=P.Vnum(Id,"U-LOOP-P");    // in W/K
  else

mU LOOP P=D.Value("dSES U LOOP 1")+(D.Value("dSES U LOOP 2")*mAREA);

  if (mAREA>0)
    mU LOOP=mA1+(mA2*D.Value("dSES U LOOP 3"))+mU LOOP P/mAREA;
  else
```

```
    mU LOOP=0;

  //CALCULATION OF SIGMA-LOOP (efficiency factor of the collector
taking into account the influence of the heat exchanger)
  double mSIGMA_LOOP;
  double mDELTA SIGMA;

  if (P.Vcod(Id,"HEAT-EXCHANGE")=="YES") {
    if (P.Vcod(Id,"HEAT-EXCH-LOSS-KNOWN")=="YES") {
      if (P.Vnum(Id,"HEAT-EXCH-LOSS")>0)                    // in W/K
        mDELTA_SIGMA=mSIGMA_ZERO*mAREA*mA1/P.Vnum(Id,"HEAT-EXCH-
LOSS");
      else
        mDELTA_SIGMA=0;
      mSIGMA LOOP=1-mDELTA SIGMA;
    } // end if
    else
      mSIGMA LOOP=D.Value("dSES SIGMA LOOP");  //  penalty  value  for
sigma loop as given in EN 15316
  } // end if
  else {
    mDELTA SIGMA=0;
    mSIGMA LOOP=1-mDELTA SIGMA;
  } // end else

  //CALCULATION  OF  REFERENCE  TEMPERATURE  DIFFERENCE  DELTA-TEMP  -
monthly calculation

  double mTEMP REF HW[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  double mDELTA TEMP HW[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };

  double mTEMP EXT[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };

  string mWEATHER;

  if(gId<0) mWEATHER=" LON"; else mWEATHER=" "+P.Vcod(gId,"WEATHER");
  for (int i=0;i<12;i++)
    mTEMP EXT[i]=T.Value("T003"+mWEATHER,1,i+1);

  for (int i=0;i<12;i++) {
    // In order to use the same values used in the DHW calculation,
in the line below:
    // TR's default dSES MAINS WATER TEMP=12 has been replaced with
table T.Value("tDHW TEMP",1,2)=10

mTEMP REF HW[i]=D.Value("dSES TEMP REF 1")+(D.Value("dSES TEMP REF 2"
)*D.Value("dSES HW TEMP"))+(D.Value("dSES TEMP REF 3")*T.Value("tDHW
TEMP",1,2))-(D.Value("dSES_TEMP_REF_4")*mTEMP_EXT[i]);
    mDELTA TEMP HW[i]=mTEMP REF HW[i]-mTEMP EXT[i];
  } // end for


  // CALCULATION OF STORAGE TANK CAPACITY FACTOR STORAGE-COR-FAC

  double mSTOR COR FAC HW M[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
};
  double mSES_STOR_VOL;
  double mSES STOR VOL HW M[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
};

  // Solar volume input by the user
  mSES STOR VOL=P.Vnum(Id,"SES-STORE-VOL");   // litres

  if(mSES STOR VOL==0.0) {
    mSES STOR VOL=D.Value("dSES SES VOL 1")*mAREA;  // if the storage
volume is 0, assume 5 litres per m2 of collector
    // save the new value for the keyword
    {
     ostringstream res; res << mSES_STOR_VOL;
```

```
     P.ModKey(Id,"SES-STORE-VOL",res.str());
    }
  }

  // Update total solar storage of parent HW system
  double mTOTAL STOR SES=0.0;
  if (P.Vcodid(Id,"SES-PRE-HEAT-TYPE")==2) { // 2=Combined cylinder
    mTOTAL_STOR_SES=P.Vnum(pId,"STORE-VOL-SES");   // solar volume of
the parent HW system
    mTOTAL STOR SES+=mSES STOR VOL*P.Vnum(Id,"MULTIPLIER");     // add
solar storage of this SES (including multiplier)
    // save the new value for the keyword
    {
     ostringstream res; res << mTOTAL STOR SES;       // Save the
updated solar volume of the parent HW system
     P.ModKey(pId,"STORE-VOL-SES",res.str());
    }
  }

  //monthly calculations of Solar Storage and storage correction
factor
  for(int i=0;i<12;i++){
    mSES STOR VOL HW M[i]=m DHW FRAC M[i]*mSES STOR VOL;

    if(mSES STOR VOL HW M[i]>0)

mSTOR COR FAC HW M[i]=pow((m DHW FRAC M[i]*D.Value("dSES VOL REF")*mA
REA)/(mSES STOR VOL HW M[i]),0.25);
    else
      mSTOR COR FAC HW M[i]=0;
  } // end for


  //calculation of length of the month
  double mHOURS MONTH [12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  for (int i=0;i<12;i++)
    mHOURS MONTH[i]=T.Value("NJ",1,i+1)*24;


  //CALCULATION OF X AND Y FACTORS - MONTHLY CALCULATIONS -

  deque<double> mQ_RAD_M=P.Vlnum(Id,"Q-RAD-M");
  double mQ RAD TOTAL=P.Vnum(Id,"Q-RAD");

  double mX_FACTOR_HW[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  double mY FACTOR HW[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };

  for (int i=0;i<12;i++) {
    if (mKWH DHW M[i]>0) {

mX_FACTOR_HW[i]=m_DHW_FRAC_M[i]*mAREA*mU_LOOP*mSIGMA_LOOP*mDELTA_TEMP
 HW[i]*mSTOR COR FAC HW M[i]*mHOURS MONTH[i]/(mKWH DHW M[i]*1000);

mY_FACTOR_HW[i]=1000*mAREA*mIAM*mSIGMA_ZERO*mSIGMA_LOOP*(mQ_RAD_M[i]/
(3.6*mHOURS MONTH[i]))*mHOURS MONTH[i]/(mKWH DHW M[i]*1000);
    } // end if
  } // end for


  //CALCULATION OF OUTPUT OF SOLAR THERMAL SYSTEM

  double mF HW[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  double mKWH_SES_HW_M[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };

  double  a HW=T.Value("T102 SES CORREL FACTORS",1,1); //this assumes
that the connection type is water storage
  double b HW=T.Value("T102 SES CORREL FACTORS",1,2);
  double c HW=T.Value("T102 SES CORREL FACTORS",1,3);
  double d_HW=T.Value("T102_SES_CORREL_FACTORS",1,4);
```

```
  double e HW=T.Value("T102 SES CORREL FACTORS",1,5);
  double f HW=T.Value("T102 SES CORREL FACTORS",1,6);

  for (int i=0;i<12;i++){

mF HW[i]=a HW*mY FACTOR HW[i]+b HW*mX FACTOR HW[i]+c HW*pow(mY FACTOR
 HW[i],2)+d HW*pow(mX FACTOR HW[i],2)+e HW*pow(mY FACTOR HW[i],3)+f H
W*pow(mX_FACTOR_HW[i],3);
    if (mF HW[i]<0)
      mF HW[i]=0;
    if (mF_HW[i]>1)
      mF HW[i]=1;

    mKWH_SES_HW_M[i]=mF_HW[i]*mKWH_DHW_M[i];
  } // end for


  //CALCULATION OF AUXILIARY ENERGY CONSUMPTION

  double mQ_AUX_POWER;
  double mQ SES AUX[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  double mPUMP TIME[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  double mIrrad=0.0; //in W/m2

  if(P.Vcod(Id,"AUX-POWER-KNOWN")=="YES")
    mQ AUX POWER=P.Vnum(Id,"AUX-POWER");    // in Watts
  else
    mQ AUX POWER=25+2*mAREA;

  for (int i=0;i<12;i++)
    mIrrad+=(1000*mQ RAD M[i])/(3.6*mHOURS MONTH[i]);

  for (int i=0;i<12;i++) {
    if ( (mHOURS MONTH[i]>0) && (mIrrad>0) )

mPUMP TIME[i]=D.Value("dSES PUMP TIME")*(1000*mQ RAD M[i]/(3.6*mHOURS
 MONTH[i])/mIrrad);

    if(P.Vcodid(Id,"SES-CIRC-SYS")!=3)     //  3=forced  circulation
system with no PV
      mQ SES AUX[i]=0;
    else
      mQ SES AUX[i]=mQ AUX POWER*mPUMP TIME[i]/1000; // units in kW
and hours
  } // end for

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
    res << mQ SES AUX[i] << ", ";
   res << mQ_SES_AUX[11] << " }";
   P.ModKey(Id,"KWH-SES-AUX-M",res.str());
  }


  // SYSTEM THERMAL LOSSES FROM SES

  //tank thermal losses
  double mSES LOSS HW[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 }; //
thermal losses of the solar storage allocated to hot water
production, in KWh

  // Calculate the storage losses if the SES has separate storage
  if  (P.Vcodid(Id,"SES-PRE-HEAT-TYPE")==1)  {  //  1=Separate  solar
cylinder

    double mSTORE LOSS=0.0;      // storage losses in MJ/month
    double  mSTORE VOL5=D.Value("DHW 10")*mAREA SERVED;  //  storage
volume (in litres) when the annual DHW demand is 5 MJ/m2
```

```
    double mpwr1=1.0/3.0;
    double mpwr2=2.0/3.0;
    double   mINSUL THICK=P.Vnum(Id,"STORE-INSULAT-THICK");         //
insulation thickness, in mm, on storage vessel
    double mKWH LOSS DAY=0.0;    // daily storage losses in kWh/litre
of storage

    // Ensure user has not input a negative value for insulation
thickness
    if (mINSUL THICK<0)
      mINSUL_THICK=0;

    switch(P.Vcodid(Id,"STORE-INSULAT-TYPE")) {
      case 1: { //Uninsulated
                mKWH LOSS DAY=D.Value("DHW 4");      // daily storage
losses in kWh/litre of storage
               }
               break;
      case 2: { //Loose jacket

mKWH LOSS DAY=D.Value("DHW 11")+(D.Value("DHW 12")/(mINSUL THICK+D.Va
lue("DHW 13")));     // daily storage losses in kWh/litre of storage
               }
               break;
      case 3: { //Factory insulated

mKWH LOSS DAY=D.Value("DHW 11")+(D.Value("DHW 14")/(mINSUL THICK+D.Va
lue("DHW 15")));     // daily storage losses in kWh/litre of storage
               }
               break;
    } // end switch


mSTORE LOSS=mKWH LOSS DAY*(pow(mSTORE VOL5,mpwr1))*(365.0/12.0)*(pow(
mSES STOR VOL,mpwr2));     // kWh/month

    for (int i=0;i<12;i++) {
      if(mKWH DHW M[i]>0)
        mSES_LOSS_HW[i]=mSTORE_LOSS*mKWH_SES_HW_M[i]/mKWH_DHW_M[i];
// storage losses modulated by fraction of solar contribution to HW
demand, in kWh
      else
        mSES LOSS HW[i]=0;
    } // end for

  } // end if


  //thermal losses due to distribution between SES and back-up heater

  double mHEAT LOSS PIPES HW[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
};

  // Calculate the distribution losses if the SES has separate
storage
  if  (P.Vcodid(Id,"SES-PRE-HEAT-TYPE")==1)  {  //  1=Separate  solar
cylinder
    //domestic hot water
    for (int i=0;i<12;i++) {
      if(mKWH_SES_HW_M[i]>0)

mHEAT LOSS PIPES HW[i]=T.Value("T102 SES PIPE LOSSES",1,P.Vcodid(Id,"
PIPE-INSULAT"))*(mKWH_DHW_M[i])*(mKWH_SES_HW_M[i])/(mKWH_DHW_M[i]);
      else
        mHEAT LOSS PIPES HW[i]=0;
    } // end for
  } // end if
  else // i.e., combined cylinder
    for (int i=0;i<12;i++)
```

```
      mHEAT LOSS PIPES HW[i]=0;


  //calculations of useful/delivered solar energy - after losses
  double mKWH SES HW USE M[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
};

  for (int i=0;i<12;i++){
    if(mKWH SES HW M[i]-mSES LOSS HW[i]-mHEAT LOSS PIPES HW[i]>0)
      mKWH SES HW USE M[i]=mKWH SES HW M[i]-mSES LOSS HW[i]-
mHEAT_LOSS_PIPES_HW[i];
    else
      mKWH SES HW USE M[i]=0;
  } // end for

  // 18 April 2010 JO: Shell and core. SES contribution reduced by
shell and core ratio
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
    res  <<  mKWH SES HW USE M[i]*3.6*P.Vnum(pId,"SC-ACTUAL-RATIO")  <<
", ";
   res  <<  mKWH SES HW USE M[11]*3.6*P.Vnum(pId,"SC-ACTUAL-RATIO")  <<
" }";
   P.ModKey(Id,"Q-SES-HW-M",res.str());
  }

} // end Q SES


void  SES::Recommendation(PROJECTDefaults&  D,  PROJECTTables&  T,  BOC&
P) {
  if(func report) cout << endl << "  " << "SES::Recommendation";

}



// End Object: SES
```

SMOKE-VENT (NO LONGER IN USE)

Specifies the size and construction of a smoke vent.

Example input:

```
"z3/01/SV" = SMOKE-VENT
   AREA         = 45
   U-VALUE      = 2.0
..
```

AREA (NO LONGER IN USE)

Specifies the area of the smoke vent, in m^2. This is a required keyword.

U-VALUE (NO LONGER IN USE)

Stores the U-value for the smoke vent in, W/m^2K.

MULTIPLIER (NO LONGER IN USE)

This keyword specifies how many smoke vents with this description exist in the surface. E.g. if the value of the multiplier is 3, there are 3 smoke vents each with identical keyword values in the roof considered. This enables repeating smoke vents to be entered quickly and easily. The default value is 1.

TSC

The TSC command defines the performance parameters of a Transpired Solar Collector (TSC) system. A U-name is required.

Example input:

```
"TSC 1" = TSC
   TYPE                = Standard operation
   OPERATION-TYPE      = Constant flow
   CONTROL-TYPE        = Only manual
   ABSORTIVITY         = Good
   FAN-INDEPENDENT     = YES
   SFP                 = 1.5
   SHADING-FACTOR      = 1
   AIR-FLOW-RATE-DES   = 0.2
  ..
```

TYPE
Specifies the type of the TSC system, as one of the values listed below. The default is 'Standard operation'.

Standard operation
High temperature rise
High air volume

OPERATION-TYPE
Specifies the operation type of the TSC system, as one of the values listed below. The default is 'Constant flow'.

Constant flow
Variable flow

CONTROL-TYPE
Specifies the control type of the TSC system, as one of the values listed below. The default is 'Only manual'.

Only manual
Automatic

ABSORTIVITY
Specifies the absorptivity of the TSC system, as one of the values listed below. The default is 'Good'.

Very high
High
Good
Moderate
Low

FAN-INDEPENDENT
Specifies whether the TSC system is provided with an independent fan. It can be either YES or NO. The default is YES.

SFP
Specifies the specific the fan power for the TSC system, in W/(l/s). The default value is 2.

SHADING-FACTOR
Specifies the shading correction (reduction) factor for the TSC system, in decimal form. A value of 1 means the TSC system is unshaded. This keyword is required if a TSC object is included in the input.

AIR-FLOW-RATE-DES
Specifies the design air flow rate for the TSC system, in m^3/s. This keyword is required if a TSC object is included in the input.

c++ code

```
//: tsc.cpp

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void TSC::TEST() {
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test TSC" << endl;
//  cout << "Fin de Test TSC" << endl;
  BEEMObject::TEST();
}

// TSC::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P, BOC&N,
int i)

void TSC::fQtsc(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "    " << "TSC::fCollEffV";
  if(calc report) cout << endl << P.vINP[Id].Name;

  //Calculate efficiency and flows (if variable)
  fCollEff(D,T,P);
}

void TSC::fQaux(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "    " << "TSC::fQaux";
  if(calc report) cout << endl << P.vINP[Id].Name;

  //Calculate auxiliary energy

  string mWEATHER;
  if(gId<0) mWEATHER="_LON"; else mWEATHER="_"+P.Vcod(gId,"WEATHER");
  double mNJ[12];
  double mTdlH[12];
  double mTSCuse[12];
  deque<double> mAirFlow=P.Vlnum(Id,"AIR-FLOW-RATE");
  double mSFP=P.Vnum(Id,"SFP");
  double mKWH_AUX[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };

  if(P.Vcodid(Id,"FAN-INDEPENDENT")==1 && P.Vnum(Id,"ZONE-SUPPLY")>0)
    for(int m=0;m<12;m++) {
      mTdlH[m]=T.Value("TdlH"+mWEATHER,1,m+1);
      mNJ[m]=T.Value("NJ",1,m+1);
      mTSCuse[m]=T.Value("tTSC_use",1,m+1);
```

```
      mKWH AUX[m]=mAirFlow[m]*mSFP*mTdlH[m]*mNJ[m]*mTSCuse[m];
    }

  {
   ostringstream res; res << "{ ";
   for (int l=0;l<11;l++)
     res << mKWH AUX[l] << ", ";
   res << mKWH_AUX[11] << " }";
   P.ModKey(Id,"KWH-AUX",res.str());
  }

}

void TSC::fQheatrec(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "    " << "TSC::fQheatrec";
  if(calc report) cout << endl << P.vINP[Id].Name;

  //Calculate heat recapture

  string mWEATHER;
  if(gId<0) mWEATHER=" LON"; else mWEATHER=" "+P.Vcod(gId,"WEATHER");
  double mNJ[12];
  double mTdlH[12];

  double mQ HEAT REC[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };

  {
   ostringstream res; res << "{ ";
   for (int l=0;l<11;l++)
     res << mQ HEAT REC[l] << ", ";
   res << mQ HEAT REC[11] << " }";
   P.ModKey(Id,"Q-HEAT-REC",res.str());
  }

}

void TSC::fQtot(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "    " << "TSC::fQtot";
  if(calc_report) cout << endl << P.vINP[Id].Name;

  //Calculate heat recapture

  double mQ TOT[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  double mQ TOT DIRECT[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  deque<double> mQtsc=P.Vlnum(Id,"Q-TSC");
  deque<double> mKWHaux=P.Vlnum(Id,"KWH-AUX");
  deque<double> mQheatrec=P.Vlnum(Id,"Q-HEAT-REC");

  double mHeatDirect;

  for(int m=0;m<12;m++) {
    mHeatDirect=T.Value("tTSC HeatDirect",1,m+1);
    if(mHeatDirect<0)
      mQ_TOT[m]=mQtsc[m]+mQheatrec[m]+mKWHaux[m]*3.6;
    else {
      mQ TOT[m]=(1.0-mHeatDirect)*(mQtsc[m]+mQheatrec[m]);

mQ TOT DIRECT[m]=mHeatDirect*(mQtsc[m]+mQheatrec[m])+mKWHaux[m]*3.6;
    }
  }

  {
   ostringstream res; res << "{ ";
   for (int l=0;l<11;l++)
     res << mQ TOT[l] << ", ";
   res << mQ_TOT[11] << " }";
   P.ModKey(Id,"Q-TOT",res.str());
  }
  {
```

```
    ostringstream res; res << "{ ";
    for (int l=0;l<11;l++)
      res << mQ TOT DIRECT[l] << ", ";
    res << mQ TOT DIRECT[11] << " }";
    P.ModKey(Id,"Q-TOT-DIRECT",res.str());
   }
}

void TSC::fCollEff(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "    " << "TSC::fCollEffV";
  if(calc_report) cout << endl << P.vINP[Id].Name;

  // Reporting
  deque<double> rep_IncTdel=P.Vlnum(Id,"REP-INC-TDEL");
  deque<double> rep Tact=P.Vlnum(Id,"REP-TACT");
  deque<double> mZONE SET POINT=P.Vlnum(Id,"ZONE-SET-POINT");

  if(rep IncTdel[0]==-5555) rep IncTdel.clear();
  if(rep Tact[0]==-5555) rep Tact.clear();

  deque<double> rep Gsolar, rep dCeff;
  deque<double> rep Qtsc; rep Qtsc.clear();

  string mWEATHER;
  if(gId<0) mWEATHER=" LON"; else mWEATHER=" "+P.Vcod(gId,"WEATHER");
  double mNJ[12];
  double mTA[12];
  double mTrec[12];
  double mTdlH[12];
  for(int m=0;m<12;m++) {
    mTA[m]=T.Value("T003"+mWEATHER,1,m+1);//+273.5;
    mTdlH[m]=T.Value("TdlH"+mWEATHER,1,m+1);
    mNJ[m]=T.Value("NJ",1,m+1);
    mTrec[m]=mZONE SET POINT[m]+276; //273.5+5/2
  }

  int nWalls=P.Vint(Id,"WALL-N");
  double mVol, maxcheck;
//  deque<double> dMULTIPLIER=P.Vlnum(Id,"MULTIPLIER");
  deque<double> dAREA=P.Vlnum(Id,"INWALL-AREA");
  deque<double> dWALLAREA=P.Vlnum(Id,"WALL-AREA");
  double mAirDens=D.Value("dTSC_AirDens");
  double mCp=D.Value("dTSC Cp");
  double mToffset=D.Value("dTSC Offset");

  deque<double> rep AirFlow=P.Vlnum(Id,"REP-AIR-FLOW-RATE");
  double mZONE SUPPLY=P.Vnum(Id,"ZONE-SUPPLY");

  int nInt;
  if(P.Vcodid(Id,"OPERATION-TYPE")==2)
    nInt=4;
  else
    nInt=1;

  double temp;
  double mAREA=P.Vnum(Id,"AREA");
  double mVS[12];
  double mAirFlow[12];
  double mAirFlowNew[12];
  double mQtsc[12];
  double mQ;
  for(int x=0;x<nInt;x++) {
    fCollEff_fixflow(D,T,P);
    rep Gsolar=P.Vlnum(Id,"REP-SOLAR-GAIN");
    rep dCeff=P.Vlnum(Id,"REP-CEFF");
    int rep_AirFlow_N=rep_AirFlow.size();
    for(int m=0;m<12;m++) {
      mAirFlowNew[m]=0.0;
      mQtsc[m]=0.0;
```

```
    }

    for(int j=0;j<nWalls;j++) {
      for(int m=0;m<12;m++) {
        mAirFlow[m]=rep_AirFlow[rep_AirFlow_N-12+m]*dAREA[j]/mAREA;
        mVS[m]=mAirFlow[m]/dAREA[j];
      }
      double mIncTdel[12];
      double mTact[12];
      double mTdel[12];
      for (int m=0;m<12;m++) {
        mQ=rep dCeff[m+12*j+12*nWalls*x]*rep Gsolar[m+12*j]*dAREA[j];
        if(mZONE SUPPLY>0) mQtsc[m]+=mQ;

mTact[m]=rep dCeff[m+12*j+12*nWalls*x]*rep Gsolar[m+12*j]*dAREA[j]/(m
AirFlow[m]*mAirDens*mCp);

//mIncTdel[m]=rep dCeff[m+12*j+12*nWalls*x]*rep Gsolar[m+12*j]*dAREA[
j]/(mAirFlow[m]*mAirDens*mCp);
        mIncTdel[m]=mTA[m]+mToffset+mTact[m];
//        if(x==0) {
//                                maxcheck=7.5;   if(mIncTdel[m]>maxcheck)
maxcheck=mIncTdel[m];
//          mAirFlowNew[m]+=7.5/maxcheck*mAirFlow[m];
//        }
//        else {
          temp=(mTrec[m]-mIncTdel[m])/(mTrec[m]-mTact[m]);
//if(temp>1) temp=1;
          mAirFlowNew[m]+=temp*mAirFlow[m];
//        }

        // Reporting
        rep IncTdel.push back(mIncTdel[m]);
        rep Tact.push back(mTact[m]);
        if(x==nInt-1) {
          if(mZONE SUPPLY>0)
            rep Qtsc.push back(mQ);
          else
            rep_Qtsc.push_back(0.0);
        }
      }
    }

    // Reporting
    for(int m=0;m<12;m++) rep_AirFlow.push_back(mAirFlowNew[m]);
    {
     ostringstream res; res << "{ ";
     for (int l=0;l<rep_AirFlow.size()-1;l++)
       res << rep AirFlow[l] << ", ";
     res << rep AirFlow[rep AirFlow.size()-1] << " }";
     P.ModKey(Id,"REP-AIR-FLOW-RATE",res.str());
    }

  }
    if(nInt>1)
      {
       ostringstream res; res << "{ ";
       for (int l=0;l<11;l++)
         res << mAirFlow[l] << ", ";
       res << mAirFlow[11] << " }";
       P.ModKey(Id,"AIR-FLOW-RATE",res.str());
      }
    {
     ostringstream res; res << "{ ";
     for (int l=0;l<rep IncTdel.size()-1;l++)
       res << rep_IncTdel[l] << ", ";
     res << rep IncTdel[rep IncTdel.size()-1] << " }";
     P.ModKey(Id,"REP-INC-TDEL",res.str());
    }
```

```
    {
     ostringstream res; res << "{ ";
     for (int l=0;l<rep_Tact.size()-1;l++)
       res << rep_Tact[l] << ", ";
     res << rep_Tact[rep_Tact.size()-1] << " }";
     P.ModKey(Id,"REP-TACT",res.str());
    }
    {
     ostringstream res; res << "{ ";
     for (int l=0;l<rep_Qtsc.size()-1;l++)
       res << rep_Qtsc[l] << ", ";
     res << rep_Qtsc[rep_Qtsc.size()-1] << " }";
     P.ModKey(Id,"REP-Q-TSC",res.str());
    }
    {
     ostringstream res; res << "{ ";
     for (int l=0;l<11;l++)
       res << mQtsc[l] << ", ";
     res << mQtsc[11] << " }";
     P.ModKey(Id,"Q-TSC",res.str());
    }

}

void TSC::fCollEff_fixflow(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func_report) cout << endl << "      " << "TSC::fCollEff";
  if(calc_report) cout << endl << P.vINP[Id].Name;

    // Reporting
    deque<double> rep_Uinf=P.Vlnum(Id,"REP-WIND-SPEED");
    deque<double> rep_Gsolar=P.Vlnum(Id,"REP-SOLAR-GAIN");
    deque<double> rep_hc=P.Vlnum(Id,"REP-HC");
    deque<double> rep_t=P.Vlnum(Id,"REP-T");
    deque<double> rep_hr=P.Vlnum(Id,"REP-HR");
    deque<double> rep_dCeff=P.Vlnum(Id,"REP-CEFF");
    deque<double> rep_AirFlow=P.Vlnum(Id,"REP-AIR-FLOW-RATE");

    if(rep_Uinf[0]==-5555) rep_Uinf.clear();
    if(rep_Gsolar[0]==-5555) rep_Gsolar.clear();
    if(rep_hc[0]==-5555) rep_hc.clear();
    if(rep_t[0]==-5555) rep_t.clear();
    if(rep_hr[0]==-5555) rep_hr.clear();
    if(rep_dCeff[0]==-5555) rep_dCeff.clear();

    // Calc
    deque<double> dCollEff;
    double mAREA=P.Vnum(Id,"AREA");
    int nWalls=P.Vint(Id,"WALL-N");

    deque<double> dORIENTATION=P.Vlnum(Id,"ORIENTATION");
    deque<double> dINCLINATION=P.Vlnum(Id,"INCLINATION");
    deque<double> mCOLL_EFF;

    int mOrientation[8]={ 5, 4, 3, 2, 1, 8, 7, 6 }; //Relation
between wall and ses orientation. Convert from Wall to SES
orientation

    string mWEATHER;
    if(gId<0)                   mWEATHER=" LON";                   else
mWEATHER=" "+P.Vcod(gId,"WEATHER");

    double
mAbsortivity=T.Value("tTSC_absortivity",1,P.Vcodid(Id,"ABSORTIVITY"))
;
    double mAirDens=D.Value("dTSC_AirDens");
    double mTE=D.Value("dTSC_TE");
    double mCp=D.Value("dTSC_Cp");
    double mSb=D.Value("dTSC_sb");
```

```
    double mNJ[12];
    double mTA[12];
    double mTdlH[12];
    for(int m=0;m<12;m++) {
      mTA[m]=T.Value("T003"+mWEATHER,1,m+1)+273.5;
      mTdlH[m]=T.Value("TdlH"+mWEATHER,1,m+1);
      mNJ[m]=T.Value("NJ",1,m+1);

    }

    int rep AirFlow N=rep AirFlow.size();
    double mVS[12];
    double eHX[12];
    double mAirFlow[12];
    for(int m=0;m<12;m++) {
      mAirFlow[m]=rep_AirFlow[rep_AirFlow_N-12+m];
      mVS[m]=mAirFlow[m]/mAREA;
      eHX[m]=1.0-5.0*mVS[m];
    }

    for(int i=0;i<nWalls;i++) {
      double mEFF=0.0;
      double mUinf[12];
      double mGsolar[12];
      double hc[12];
      double t[12];
      double hr[12];
      double dCeff[12];
      double Ceff=0;

      for(int m=0;m<12;m++) {

mUinf[m]=T.Value("tVinf"+mWEATHER,m+1,(int)dORIENTATION[i])*D.Value("
dTSC WindCorr");
        mGsolar[m]=P.Vnum(Id,"SHADING-
FACTOR")*T.Value(T.TValue($T033 tables,P.Vcodid(gId,"WEATHER"),m+1),
              mOrientation[(int)dORIENTATION[i]-
1],(int)dINCLINATION[i])*1000000/(mNJ[m]*mTdlH[m]*3600);
        hc[m]=6.0+4*mUinf[m]-76.0*mVS[m];

t[m]=mAbsortivity*mGsolar[m]/(mAirDens*mCp*mVS[m]*eHX[m]+4.0*mTE*mSb*
mTA[m]*mTA[m]*mTA[m]+hc[m]);
        hr[m]=4.0*mTE*mSb*mTA[m]*mTA[m]*(mTA[m]+1.6*t[m]);

dCeff[m]=mAbsortivity/(1.0+(hr[m]+hc[m])/(mAirDens*mCp*mVS[m]*eHX[m])
);
//        Ceff+=1.0/12.0*dCeff[m];

//cout                    <<                    endl                    <<
T.Value(T.TValue($T033_tables,P.Vcodid(gId,"WEATHER"),m+1),mOrientati
on[(int)dORIENTATION[i]-1],(int)dINCLINATION[i]);

        // Reporting
        rep Uinf.push back(mUinf[m]);
        rep Gsolar.push back(mGsolar[m]);
        rep_hc.push_back(hc[m]);
        rep t.push back(t[m]);
        rep hr.push back(hr[m]);
        rep_dCeff.push_back(dCeff[m]);

      }
      //cout << endl << Ceff << "  " << mSb << endl;
//      dCollEff.push back(Ceff);
    }

    int rep size=rep Uinf.size();
    // Reporting
    {
```

```
     ostringstream res; res << "{ ";
     for (int l=0;l<rep_size-1;l++)
       res << rep_Uinf[l] << ", ";
     res << rep_Uinf[rep_size-1] << " }";
     P.ModKey(Id,"REP-WIND-SPEED",res.str());
    }
    {
     ostringstream res; res << "{ ";
     for (int l=0;l<rep_size-1;l++)
       res << rep_Gsolar[l] << ", ";
     res << rep_Gsolar[rep_size-1] << " }";
     P.ModKey(Id,"REP-SOLAR-GAIN",res.str());
    }
rep_size=rep_hc.size();
    {
     ostringstream res; res << "{ ";
     for (int l=0;l<rep_size-1;l++)
       res << rep_hc[l] << ", ";
     res << rep_hc[rep_size-1] << " }";
     P.ModKey(Id,"REP-HC",res.str());
    }
    {
     ostringstream res; res << "{ ";
     for (int l=0;l<rep_size-1;l++)
       res << rep_t[l] << ", ";
     res << rep_t[rep_size-1] << " }";
     P.ModKey(Id,"REP-T",res.str());
    }
    {
     ostringstream res; res << "{ ";
     for (int l=0;l<rep_size-1;l++)
       res << rep_hr[l] << ", ";
     res << rep_hr[rep_size-1] << " }";
     P.ModKey(Id,"REP-HR",res.str());
    }
    {
     ostringstream res; res << "{ ";
     for (int l=0;l<rep_size-1;l++)
       res << rep_dCeff[l] << ", ";
     res << rep_dCeff[rep_size-1] << " }";
     P.ModKey(Id,"REP-CEFF",res.str());
    }

}

void TSC::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "  " << "TSC::Calc";
  if(calc_report) cout << endl << P.vINP[Id].Name;
// its own calcs below...

  // Area Calc and relation to Walls

  double mAREA=0;
  deque<double> dAREA=P.Vlnum(Id,"INWALL-AREA");
  int nWalls=P.Vint(Id,"WALL-N");;
  for(int i=0;i<nWalls;i++) mAREA+=dAREA[i];

  if(mAREA<=0) {
    // Through error
    {
     ostringstream res; res << "{ ";
     for (int l=0;l<11;l++)
       res << 0 << ", ";
     res << 0 << " }";
     P.ModKey(Id,"KWH-AUX",res.str());
    }

    {
     ostringstream res; res << "{ ";
```

```
     for (int l=0;l<11;l++)
       res << 0 << ", ";
     res << 0 << " }";
     P.ModKey(Id,"Q-HEAT-REC",res.str());
    }

    {
     ostringstream res; res << "{ ";
     for (int l=0;l<11;l++)
       res << 0 << ", ";
     res << 0 << " }";
     P.ModKey(Id,"Q-TOT",res.str());
    }
    {
     ostringstream res; res << "{ ";
     for (int l=0;l<11;l++)
       res << 0 << ", ";
     res << 0 << " }";
     P.ModKey(Id,"Q-TOT-DIRECT",res.str());
    }

  }
  else {
    {
     ostringstream res; res << mAREA;
     P.ModKey(Id,"AREA",res.str());
    }

    double mDES FLOW=P.Vnum(Id,"AIR-FLOW-RATE-DES");
    {
     ostringstream res; res << "{ ";
     for (int l=0;l<11;l++)
       res << mDES FLOW << ", ";
     res << mDES FLOW << " }";
     P.ModKey(Id,"REP-AIR-FLOW-RATE",res.str());
     P.ModKey(Id,"AIR-FLOW-RATE",res.str());
    }

    fQtsc(D,T,P);
    fQaux(D,T,P);
    fQheatrec(D,T,P);
    fQtot(D,T,P);

  }


}


void TSC::Recommendation(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func report) cout << endl << "  " << "TSC::Recommendation";

}



// End Object: TSC
```

WALL

Specifies the size and construction of an envelope element. A U-name must be specified, but it is only used for output reports.

Example input:

```
"Left_Wall" = WALL
  TYPE          = Exterior
  TYPE-ENV      = Wall
  AREA          = 60
  ORIENTATION   = West
  CONSTRUCTION  = "Default construction for walls"
  ..
```

TYPE

Specifies the type of the WALL object, with regards to the condition of the adjacent space, as one of the values listed below. This is a required keyword.

Value	Description
Exterior	For constructions separating the considered zone from the outside air or water.
Strongly ventilated spaces	For constructions separating the considered zone from a space provided with one or more permanent openings (that cannot be closed), with a capacity for the supply of fresh air and extract of inside air, determined according to 5.3 of NEN 1087, of at least $3x10^{-3}$ m^3/s per m^2 useable area.
Unheated adjoining space	For constructions separating the considered zone from an unheated adjoining space, other than meant under 'Strongly ventilated spaces'.
Conditioned adjoining space	For constructions separating the considered zone from another conditioned zone.
Underground	For constructions separating the considered zone from the ground or a crawl space.
Same space	For constructions representing the internal envelopes that separate contiguous zones which have been merged into one zone (according to the zoning rules), i.e., the envelope is "contained" within the merged zone.

TYPE-ENV

Specifies the type of envelope element, as one of the values listed below. This is a required keyword. Any part of a roof with a pitch angle greater than or equal to 70^o from the horizontal is considered a wall.

Value	Description
Wall	Envelope element is a wall
Roof	Envelope element is a roof
Floor or Ceiling	Envelope element is a floor or ceiling

AREA

Specifies the area of the envelope element, in m^2, including windows and doors. This is a required keyword.

LENGTH

For an envelope element of type Wall (in the WALL object's TYPE-ENV keyword), this specifies the perimeter length, including windows and doors, in m. The default value is 1000.

PITCH
Specifies the pitch angle from the horizontal of the envelope element measured in degrees. The default value is 45.

ORIENTATION
Specifies the orientation of the envelope element, as one of the values listed below.

South
South-East
East
North-East
North
North-West
West
South-West
Horizontal

CONSTRUCTION
Specifies the U-name of a previously-defined CONSTRUCTION instruction to define the construction of the WALL. This is a required keyword.

***FLAT-ROOF* (NO LONGER IN USE)**
For a WALL element which is a roof, this keyword indicates whether the roof is flat or pitched. If it has the value YES, the roof is flat, and if the value is NO, the roof is pitched. The default is NO.

***ABSORP-COEFF* (NO LONGER IN USE)**
This is the absorption coefficient for the external surface of a roof or wall. The default value is 0.9.

TSC-SYSTEM
For an envelope element of type Wall (in the WALL object's TYPE-ENV keyword), this specifies whether a Transpired Solar Collector (TSC) system is installed on the exterior of the envelope element. It can be either YES or NO. The default is NO.

TSC
This keyword is included if the WALL object's TSC-SYSTEM keyword has been set to YES. It specifies the U-name of a previously-defined TSC object to define the name of the TSC system installed on the exterior of the envelope element.

TSC-AREA
This keyword is included if the WALL object's TSC-SYSTEM keyword has been set to YES. It specifies the area of the TSC system, in m^2, installed on the exterior of the envelope element.

MULTIPLIER
This keyword specifies how many surfaces with this exact description exist in the zone. E.g., if the value of the multiplier is 3, there are 3 surfaces (walls, roofs, or

ceilings/floors) each with identical keyword values in the zone considered. This enables repeating envelope elements to be entered quickly and easily. The default value is 1.

c++ code

```
//: wall.cpp

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void WALL::TEST() {
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test WALL" << endl;
//  cout << "Fin de Test WALL" << endl;
  BEEMObject::TEST();
  for(int i=0;i<vTM.size();i++)
    vTM[i].TEST();
  for(int i=0;i<vW.size();i++)
    vW[i].TEST();
  for(int i=0;i<vD.size();i++)
    vD[i].TEST();
  for(int i=0;i<vSV.size();i++)
    vSV[i].TEST();
}

void WALL::Reset() {
  if(func report) cout << endl << "  " << "WALL::Reset";
  dW.clear();
//  vW.clear();
  dTM.clear();
//  vTM.clear();
  dD.clear();
//  vD.clear();
  dSV.clear();
//  vSV.clear();
}

void WALL::DtoV() {
  if(func_report) cout << endl << "  " << "WALL::DtoV";
//  for(int i=0;i<dW.size();i++)  dW[i].DtoV();
//  vW.reserve(dW.size());
//  vW.assign(dW.begin(),dW.end());
//  dW.clear();
//  vTM.reserve(dTM.size());
//  vTM.assign(dTM.begin(),dTM.end());
//  dTM.clear();
//  for(int i=0;i<dD.size();i++)  dD[i].DtoV();
//  vD.reserve(dD.size());
//  vD.assign(dD.begin(),dD.end());
//  dD.clear();
//  vSV.reserve(dSV.size());
//  vSV.assign(dSV.begin(),dSV.end());
//  dSV.clear();
}


void WALL::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P, BOC
&N) {
  if(func report) cout << endl << "  " << "WALL::Notional";
//creates notional wall object

  char C=34;
  BObject Temp;
  Temp.Ntype=4;
  Temp.Type="WALL";
  Temp.Name=P.vINP[Id].Name;
```

```
  KeywordValue KV;

  int not case=P.Vint(gId,"NOT-CASE");      // type of the notional
building
  if (not_case>=9)  // 9=EW_NOT_2010
     not case+=P.Vint(pId,"DRIVER2A");     // type of the notional
building of the parent zone


  switch(P.Vcodid(Id,"TYPE-ENV")) {
 //wall:
    case 1: {
       {
        ostringstream res;
        if(P.Vcodid(Id,"TYPE")==4)  // 4=Conditioned adjoining space
          res << C << "Notional Internal Wall" << C;
        else { // i.e., the wall is exposed
          if (not case==11) // i.e., 11=EW NOT 2010 Top Lit
            res << C << "Notional Exposed Wall MC" << C;
          else // i.e., side-lit or not-lit
            res << C << "Notional Exposed Wall" << C;
        } // end else
        KV.Reset(); KV.Init(3,res.str());
        Temp.Prop.insert(make pair("CONSTRUCTION",KV));
       }
       {
        ostringstream res; res << P.Vnum(Id,"LENGTH");
        KV.Reset(); KV.Init(1,res.str());
        Temp.Prop.insert(make pair("LENGTH",KV));
       }
    }
    break;
 //roof:
    case 2: {
 //new keyword needed for pitched roof
       if  (P.Vcodid(Id,"ROOF-PITCH")>2)  //  2  =  10  degrees,  i.e.,
pitched roof
        {
         ostringstream res;
         if (not case==11) // i.e., 11=EW NOT 2010 Top Lit
           res << C << "Notional Pitched Roof MC" << C;
         else // i.e., side-lit or not-lit
           res << C << "Notional Pitched Roof" << C;
         KV.Reset(); KV.Init(3,res.str());
         Temp.Prop.insert(make_pair("CONSTRUCTION",KV));
        }
       else  // i.e., flat roof
        {
         ostringstream res;
         if (not case==11) // i.e., 11=EW NOT 2010 Top Lit
           res << C << "Notional Flat Roof MC" << C;
         else // i.e., side-lit or not-lit
           res << C << "Notional Flat Roof" << C;
         KV.Reset(); KV.Init(3,res.str());
         Temp.Prop.insert(make pair("CONSTRUCTION",KV));
        }
    }
    break;
 //floor:
    case 3: {
      {
        ostringstream res;
        if(P.Vcodid(Id,"TYPE")==4)  // 4=Conditioned adjoining space
          res << C << "Notional Internal Floor" << C;
        else  // i.e., the floor is exposed
          res << C << "Notional Exposed Floor" << C;
        KV.Reset(); KV.Init(3,res.str());
        Temp.Prop.insert(make pair("CONSTRUCTION",KV));
      }
```

```
    }
    break;
  }

 {
  ostringstream res; res << P.Vcod(Id,"TYPE");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make_pair("TYPE",KV));
 }

 {
  ostringstream res; res << P.Vcod(Id,"TYPE-ENV");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make_pair("TYPE-ENV",KV));
 }
 {
  ostringstream res; res << P.Vcod(Id,"ORIENTATION");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("ORIENTATION",KV));
 }
 {
  ostringstream res; res << P.Vnum(Id,"AREA");
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("AREA",KV));
 }
 {
  ostringstream res; res << P.Vnum(Id,"MULTIPLIER");
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("MULTIPLIER",KV));
 }
 {
  ostringstream res; res << P.Vnum(Id,"PITCH");
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("PITCH",KV));
 }

  N.vINP.push back(Temp);

//Adding doors, windows, smoke vents to notional building - changed
in v1.1.a
//also changed in v1.2.a

 double mTOTAL WIDTH=0.0;  // sum the total width of openings in the
wall

//1 -add Display windows in external walls from actual building -
changed priority v1.1.a

 double DWin Area=0.0;
 double DThis Win Area=0.0;
 double DWin_Area_Add=0.0;
 double mASPECT RATIO=0.0;

 // !!! Display windows are no longer copied over to the 2010
notional building in England and Wales
 if (not case<9) { // i.e., not England and Wales 2010 notional
   for (int j=0;j<vW.size();j++) {
     //display window:
     if (P.Vcodid(vW[j].Id,"DISP-WIND")==1) {
       //area of this display window:

DThis Win Area=P.Vnum(vW[j].Id,"AREA")*P.Vnum(vW[j].Id,"MULTIPLIER");
       //total display window area in envelope:
       DWin Area+=DThis Win Area;
       //if total display window area is bigger than area of wall,
add display window with area = remaining wall area
       if                          ((P.Vnum(Id,"AREA")-(DWin Area))<0)
DWin Area Add=P.Vnum(Id,"AREA")-(DWin Area-DThis Win Area);
       else DWin_Area_Add=DThis_Win_Area;
```

```
       // Aspect ratio input by the user for the display window
       mASPECT RATIO=P.Vnum(vW[j].Id,"ASPECT-RATIO");
       if (mASPECT RATIO<=0)   // If not input or input as <=0
         mASPECT_RATIO=D.Value("dWindow_ratio");
       // Add total width of display windows
       mTOTAL WIDTH+=sqrt(DWin Area Add/mASPECT RATIO);

       // The next line was modified to prevent the addition of a
disp. wind. with a negative area causing a crash
       // or with an area of 0 - AN 31/01/07
       //     if (DWin Area<0) j=vW.size();
       if (DWin Area Add<=0) j=vW.size();
       else {

vW[j].NotionalDisp(D,T,P,N,DWin Area Add,mASPECT RATIO,not case);
       }
     }
   }
 } // end if

//2 add pedestrian doors in external walls from actual building:

 double P Area=0.0;
 double PThis Area=0.0;
 double P Area Add=0.0;

 for (int j=0;j<vD.size();j++) {
   //pedestrian door:
   if (P.Vcodid(vD[j].Id,"TYPE")==1) {
     //area of this pedestrian door:

PThis Area=P.Vnum(vD[j].Id,"AREA")*P.Vnum(vD[j].Id,"MULTIPLIER");
     //total pedestrian door area in envelope:
     P Area+=PThis Area;
     //if total (pedestrian door area + display window area) is
bigger than area of wall, add door with area = remaining wall area
     if                    ((P.Vnum(Id,"AREA")-(DWin Area+P Area))<0)
P_Area_Add=P.Vnum(Id,"AREA")-(DWin_Area+P_Area-PThis_Area);
     else P Area Add=PThis Area;

     // Add total width of pedestrian doors
     mTOTAL WIDTH+=sqrt(D.Value("dDoor ratio")*P Area Add);

// The next line was modified to prevent the addition of a pers. door
with a negative area causing a crash
// or with an area of 0 - AN 31/01/07
//     if (P_Area<0) j=vD.size();
     if (P Area Add<=0) j=vD.size();
     else {
       vD[j].Notional(D,T,P,N,P_Area_Add);
     }
   }
 }

//3 add access doors in external walls from actual building - changed
priority v1.1.a

 double V Area=0.0;
 double VThis_Area=0.0;
 double V Area Add=0.0;

 for (int j=0;j<vD.size();j++) {
   //vehicle access door:
   if (P.Vcodid(vD[j].Id,"TYPE")==2) {
     //area of this vehicle access door:

VThis Area=P.Vnum(vD[j].Id,"AREA")*P.Vnum(vD[j].Id,"MULTIPLIER");
     //total vehicle access door area in envelope:
```

```
     V Area+=VThis Area;
     //if total (pedestrian door area + +vehicle access door area +
display window area) is bigger than area of wall, add door with area
= remaining wall area
     if              ((P.Vnum(Id,"AREA")-(DWin Area+P Area+V Area))<0)
V Area Add=P.Vnum(Id,"AREA")-(DWin Area+P Area+V Area-VThis Area);
     else V Area Add=VThis Area;

     // Add total width of vehicle doors
     mTOTAL WIDTH+=sqrt(D.Value("dDoor ratio")*V Area Add);

// The next line was modified to prevent the addition of a veh. door
with a negative area causing a crash
// or with an area of 0 - AN 31/01/07
//     if (V Area<0) j=vD.size();
     if (V Area Add<=0) break; //j=vD.size();
     else {
       vD[j].NotionalVdoor(D,T,P,N,V Area Add);
     }
   }
   // !!! High usage entrance doors are copied over to the 2010
notional building in England and Wales and in Scotland as before
   //high usage entrance door:
   if    (    (P.Vcodid(vD[j].Id,"TYPE")==3)    &&    (not case==4    ||
not case>=9) ) {
     //area of this vehicle access door:

VThis Area=P.Vnum(vD[j].Id,"AREA")*P.Vnum(vD[j].Id,"MULTIPLIER");
     //total high usage entrance door area in envelope:
     V Area+=VThis Area;
     //if total (pedestrian door area + +vehicle access door area +
high usage entrance +display window area) is bigger than area of
wall, add door with area = remaining wall area
     if              ((P.Vnum(Id,"AREA")-(DWin Area+P Area+V Area))<0)
V Area Add=P.Vnum(Id,"AREA")-(DWin Area+P Area+V Area-VThis Area);
     else V Area Add=VThis Area;

     // Add total width of high usage entrance doors
     mTOTAL_WIDTH+=sqrt(D.Value("dDoor_ratio")*V_Area_Add);

     if (V Area Add<=0) j=vD.size();
     else {
       vD[j].NotionalHdoor(D,T,P,N,V Area Add);
     }
   }
 }


//4 -add Smoke vents in roofs from actual building - changed priority
v1.1.a
// double SV_Area=0.0;
// double SVThis Area=0.0;
// double SV Area Add=0.0;

// for (int j=0;j<vSV.size();j++) {
//   //area of this smoke vent:
//
SVThis Area=P.Vnum(vSV[j].Id,"AREA")*P.Vnum(vSV[j].Id,"MULTIPLIER");
//   //total smoke vent area in envelope:
//   SV_Area+=SVThis_Area;
// //if total door+display window+smoke vent area is bigger than area
of wall, add smoke vent with area = remaining wall area
//      if  ((P.Vnum(Id,"AREA")-(P Area+V Area+DWin Area+SV Area))<0)
SV Area Add=P.Vnum(Id,"AREA")-(P Area+V Area+DWin Area+SV Area-
SVThis Area);
//   else SV_Area_Add=SVThis_Area;
//   if (SV Area<0) j=vSV.size();
//   else {
//     vSV[j].Notional(D,T,P,N,SV_Area_Add);
```

```
//   }
// }

//5 -add Windows as % of wall area (wall area is inclusive of any
door/win/sm vent area) - changed priority and calculation v1.1.a
//subtract p-door, v-door, d-win and s-vent area from window
percentage

 deque<double> mBR LIGHT CHECK=P.Vlnum(pId,"BR-LIGHT-CHECK");
 double Win Area=0.0;
 double Win_Area_Not=0.0;
 int TYPE ENV=P.Vcodid(Id,"TYPE-ENV");
 double WIN WIDTH=0.0;

 //create notional window if it is an Exterior wall or a roof
 if(  ((P.Vcodid(Id,"TYPE-ENV")==1)  ||  (P.Vcodid(Id,"TYPE-ENV")==2))
&& (P.Vcodid(Id,"TYPE")==1) ) {
   // Check the type of the notional building of the parent zone
   if (not case>=9) { // 9=EW NOT 2010
     // Get % window area from table "Not Glazing Percent" depending
on whether it is a Wall, Roof, or Floor or Ceiling,
     //  and  the  location  of  windows  in  the  parent  zone  (i.e.,
according to the keyword DRIVER2A)

Win Area Not=P.Vnum(Id,"AREA")*T.Value("Not Glazing Percent",(P.Vcodi
d(Id,"TYPE-ENV")),not case)/100;
     // If this is a wall, take the notional glazing as the lesser of
(1.5m high x full facade width) and
     // (40% of exposed facade area)
     if ((P.Vcodid(Id,"TYPE-ENV"))==1) // 1=wall
       if
(Win Area Not>(P.Vnum(Id,"LENGTH")*D.Value("dNOT WIN HEIGHT")))
         Win Area Not=P.Vnum(Id,"LENGTH")*D.Value("dNOT WIN HEIGHT");
   } // end if
   else // i.e. other types of notional building
     //get % window areas from table "Not Win Door Area" depending on
if it is wall or roof and type of building

Win_Area_Not=P.Vnum(Id,"AREA")*T.Value("Not_Win_Door_Area",(P.Vcodid(
Id,"TYPE-ENV")-1)*3+(int)mBR LIGHT CHECK[4],not case)/100.0; // JO 24
Feb 09

   // if (Win Area Not>(P Area+V Area+DWin Area+SV Area)) {
   if (Win Area Not>(P Area+V Area+DWin Area)) {
     // Win_Area=Win_Area_Not-(P_Area+V_Area+DWin_Area+SV_Area);
     Win Area=Win Area Not-(P Area+V Area+DWin Area);

     //  Width  of  window  is  length  of  wall  minus  total  width  of
openings already added above
     WIN WIDTH=P.Vnum(Id,"LENGTH")-mTOTAL WIDTH;
     if ( (WIN_WIDTH>0) && ((P.Vcodid(Id,"TYPE-ENV"))==1) ) // 1=wall
       // Aspect ratio of window=height/width, i.e., Aspect ratio of
window=(window area/width)/width
       mASPECT_RATIO=Win_Area/pow(WIN_WIDTH,2);
     else
       mASPECT RATIO=D.Value("dWindow ratio");

     if  (Win Area>=0.001)  //  to  ensure  that  the  notional  building
does not try to create windows of very tiny areas that cause a crash
       {                                WINDOW                          Wi;
Wi.Notional(D,T,P,N,P.vINP[Id].Name,Win Area,mASPECT RATIO,TYPE ENV,n
ot case); }
   } // end if
 } // end if

}


void WALL::Init(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
```

```
  if(func_report)  cout  <<  endl  <<  "    "  <<  "WALL::Init    "  <<
P.vINP[Id].Name;
//applies rotation of building

  int mORIENTATION=P.Vcodid(Id,"ORIENTATION");
  int mROTATION=P.Vcodid(gId,"BUILD-ORIENTATION");
  int newOri=mORIENTATION;

  if(mORIENTATION<9) {
    newOri=mORIENTATION-mROTATION+1;
    if(newOri<=0) newOri+=8;

  } // end if

  //ROOF-PITCH from PITCH
  double mPITCH=P.Vnum(Id,"PITCH");

  if(mPITCH<0)
    switch(P.Vcodid(Id,"TYPE-ENV")) {
      case 1: mPITCH=90; break;
      case 2: mPITCH=45; break;
      case 3: mPITCH=0;  break;
      default: mPITCH=90;
    }

  string mROOF_PITCH="45";
  if(mPITCH<=10)         { mROOF_PITCH="0"; newOri=9; }
  else if(mPITCH<12.5) mROOF_PITCH="10";
  else if(mPITCH<17.5) mROOF_PITCH="15";
  else if(mPITCH<22.5) mROOF_PITCH="20";
  else if(mPITCH<27.5) mROOF_PITCH="25";
  else if(mPITCH<32.5) mROOF_PITCH="30";
  else if(mPITCH<37.5) mROOF_PITCH="35";
  else if(mPITCH<42.5) mROOF_PITCH="40";
  else if(mPITCH<47.5) mROOF_PITCH="45";
  else if(mPITCH<52.5) mROOF_PITCH="50";
  else if(mPITCH<57.5) mROOF_PITCH="55";
  else if(mPITCH<62.5) mROOF_PITCH="60";
  else if(mPITCH<67.5) mROOF_PITCH="65";
  else if(mPITCH<=70)  mROOF_PITCH="70";
  else                 mROOF_PITCH="90";

  // Ensure pitch of walls =90
  if ( (P.Vcodid(Id,"TYPE-ENV")==1) && (mPITCH<90) ) { // 1=wall
    mPITCH=90;
    mROOF_PITCH="90";
  }

  {
   ostringstream res; res << mROOF_PITCH;
   P.ModKey(Id,"ROOF-PITCH",res.str());
  }
  {
   ostringstream res; res << mPITCH;
   P.ModKey(Id,"PITCH",res.str());
  }
    {
     ostringstream res; res << P.Vcod(Id,"ORIENTATION",newOri);
     P.ModKey(Id,"ORIENTATION",res.str());
    }

}


void WALL::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "  " << "WALL::Calc";
  if(calc_report) cout << endl << P.vINP[Id].Name;

  // Check if the total areas of windows and doors in a wall exceed
```

```
the area of the wall. If this is the case,
  // increase the area of the wall to fit the windows and doors
exactly, i.e., area of wall net of openings = 0
  if (P.Vcod(gId,"ACT-NOT")=="ACT") {
    double mAREA=P.Vnum(Id,"AREA");  // the original area of the wall
as input
    double mAREA OPEN=0.0;           // the total area of openings in
the wall

    for(int i=0;i<vW.size();i++)     //windows

mAREA OPEN+=P.Vnum(vW[i].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER");
    for(int i=0;i<vD.size();i++)      //doors

mAREA OPEN+=P.Vnum(vD[i].Id,"AREA")*P.Vnum(vD[i].Id,"MULTIPLIER");
    for(int i=0;i<vSV.size();i++)     //smoke vents

mAREA OPEN+=P.Vnum(vSV[i].Id,"AREA")*P.Vnum(vSV[i].Id,"MULTIPLIER");

    if(mAREA<mAREA_OPEN) {
      mAREA=mAREA OPEN;
      { ostringstream logtext;
      //logtext << endl << "Total Area of Windows+Doors+Smoke vents
is bigger than the area of the wall in WALL: " << P.vINP[Id].Name <<
endl;  SendLog(logtext.str()); WriteLog();
      logtext << endl << "WARNING: Total Area of Windows+Doors is
bigger than the area of WALL: " << P.vINP[Id].Name <<  ". Wall area
has been increased to fit." << endl;    SendLog(logtext.str());
WriteLog();
      //exit(1);
      }
      {
       ostringstream res; res << mAREA;   // save the increased area
of the wall into the keyword
       P.ModKey(Id,"AREA",res.str());
      }
    } // end if
  } // end if


//calculations for envelope elements

  //all envelope thermal bridges
  for (int i=0; i<vTM.size();i++)
    vTM[i].Calc(D, T, P);
  //all windows in envelope element
  for (int i=0; i<vW.size();i++)
    vW[i].Calc(D, T, P);
  //all doors in envelope element
  for (int i=0; i<vD.size();i++)
    vD[i].Calc(D, T, P);
  for (int i=0; i<vSV.size();i++)               // smoke vents, no
longer used
    vSV[i].Calc(D, T, P);
// its own calcs below...
  // Calculates estimated height and length for vertical walls
  fGEOMETRY(D, T, P);
  //calculates net area of envelope element (envelope area-
(door+window+smoke vent area))
  //fNET_AREA(D, T, P);  // JO 24 May 2010: This is now done before
we do the TSC calc to ensure we are not assuming more TSC area than
feaseble
  //correct ground floor u-values in notional building and actual
building if the value input has not already been corrected
  fU VALUE(D, T, P);
  //multiply themal capacity of envelope element by net area:
  fCM(D, P);
  //calculate transmission heat transfer coefficient for envelope
element
```

```
  //including contribution from windows, thermal bridges, doors and
smoke vents:
  fH TR(D, P);
  //calculate infiltration due to stack and wind effects:
  fU_INF_WIND(D, T, P);
  //calculate solar gain through opaque constructions (envelope
element + doors + smoke vents):
  fQ_SUN_NT(D, T, P);
}

void WALL::fGEOMETRY(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "WALL::fGEOMETRY";
  double mAREA=P.Vnum(Id,"AREA");
  double mHEIGHT=sqrt(mAREA);
//   double mLENGTH=mHEIGHT;    // LENGTH of WALL is now a required
input so no need to calculate it

  if(P.Vcodid(Id,"TYPE-ENV")==1) {
    mHEIGHT=P.Vnum(pId,"HEIGHT"); if(mHEIGHT==0) mHEIGHT=3.0;
//    mLENGTH=mAREA/mHEIGHT;
  }

 {
  ostringstream res; res << mHEIGHT;
  P.ModKey(Id,"HEIGHT",res.str());
 }
// {
//  ostringstream res; res << mLENGTH;
//  P.ModKey(Id,"LENGTH",res.str());
// }

}

void WALL::fNET AREA(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "WALL::fNET AREA";
  double mAREA=P.Vnum(Id,"AREA");

  for(int i=0;i<vW.size();i++)     //windows
    mAREA-=P.Vnum(vW[i].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER");
  for(int i=0;i<vD.size();i++)      //doors
    mAREA-=P.Vnum(vD[i].Id,"AREA")*P.Vnum(vD[i].Id,"MULTIPLIER");
  for(int i=0;i<vSV.size();i++)     //smoke vents
    mAREA-=P.Vnum(vSV[i].Id,"AREA")*P.Vnum(vSV[i].Id,"MULTIPLIER");

  if(mAREA<0) // this should not happen any more since we added the
check in Calc to increase the wall area to fit.
    mAREA=0;

 {
  ostringstream res; res << mAREA;
  P.ModKey(Id,"NET-AREA",res.str());
 }

}

void WALL::fU VALUE(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "  " << "WALL::fU_VALUE";
//corrects ground floor U-values in notional and actual buildings (if
the input value has not already been corrected) using method in CIBSE
Guide Vol A 2007

  double mU VALUE;
  double mU_VALUE_MAX;

  // Check if this is the 2010 notional as it affects the U-value
which in this case, depends on the DRIVER2A keyword of the
  // parent zone
  if ( (P.Vcod(gId,"ACT-NOT")=="NOT")   && (P.Vint(pId,"NOT-CASE")>=9)
)  { // 9=EW_NOT_2010
```

```
    if (P.Vcodid(Id,"TYPE-ENV")==1)  // 1=Wall
      mU VALUE MAX=T.Value("Not U Values",3,P.Vint(pId,"NOT-CASE"));
    else if (P.Vcodid(Id,"TYPE-ENV")==2) {  // 2=Roof
          if (P.Vcodid(Id,"ROOF-PITCH")>2) // 2 = 10 degrees, i.e.,
pitched roof
            mU VALUE MAX=T.Value("Not U Values",1,P.Vint(pId,"NOT-
CASE"));
          else // i.e., flat roof
            mU VALUE MAX=T.Value("Not U Values",2,P.Vint(pId,"NOT-
CASE"));
        } // end if
        else // i.e., Floor or Ceiling
          mU VALUE MAX=T.Value("Not U Values",4,P.Vint(pId,"NOT-
CASE"));
  } // end if

  else // not 2010 notional
    mU VALUE MAX=P.Vnum(P.Vrefid(Id,"CONSTRUCTION"),"U-VALUE");


//  if ((P.Vcodid(Id,"TYPE-ENV")==3)&&(P.Vcodid(Id,"TYPE")==5)) {
//if it is a floor, adjacent to underground, and the U-value has NOT
already been corrected to account for heat loss of floor in contact
with the ground
  if                                              ((P.Vcodid(Id,"TYPE-
ENV")==3)&&(P.Vcodid(Id,"TYPE")==5)&&(P.Vcodid(P.Vrefid(Id,"CONSTRUCT
ION"),"U-VALUE-CORR")==2)) {
    double mLENGTH WALL=0.0;
//    int mOrient[9]={0, 0, 0, 0, 0, 0, 0, 0, 0};

    //sum of exposed wall area in zone:
    for (int i=pId+1;i<P.vINP.size();i++)
      switch (P.vINP[i].Ntype) {
        //wall
        case 4: {
          if (P.Vcodid(i,"TYPE-ENV")==1) {
            //if it is adjacent to exterior or unheated space:
            if ((P.Vcodid(i,"TYPE")==1)||(P.Vcodid(i,"TYPE")==3))

mLENGTH WALL+=P.Vnum(i,"LENGTH")*P.Vnum(i,"MULTIPLIER");
          }
        }
        break;
        //zone
        case 9: {
          i=P.vINP.size();
        }
        break;
      }

      if(mLENGTH_WALL>0) {
        //calc floor exposed perimeter:
//        double mHEIGHT=P.Vnum(pId,"HEIGHT");
        double  mPERIMETER=mLENGTH WALL;                    //  exposed
perimeter of floor
//        mPERIMETER=mAREA WALL/mHEIGHT;

        double mAREA ZONE=P.Vnum(pId,"AREA");
        double   B=mAREA ZONE/(0.5*mPERIMETER);                     //
B'=characteristic dimension of floor

        double equiv thick;                          // Total equivalent
thickness of the thermal resistance
        double  therm res;                                  //  Thermal
resistance of the floor, in m2K/w

//        if (P.Vcod(gId,"ACT-NOT")=="NOT")
//           therm res=0.0;                              // 0.0 because
ground floor is assumed uninsulated in notional building.
```

```
//        else
//          therm res=1/mU VALUE MAX;

        // If this is the actual building
        if(P.Vcod(gId,"ACT-NOT")=="ACT") {
          if (mU VALUE MAX>0)
            therm res=1/mU VALUE MAX;
          else
            therm res=0.0;
        } // end if
        else { // i.e., not the actual
          // if this is the UK aggregate notional
          if (P.Vint(pId,"NOT-CASE")>=9) // 9=EW NOT 2010
            therm_res=D.Value("Floor_1");
          else // other notional buildings, i.e., NOT-CASE<9
            therm res=0.0;     // 0.0 because ground floor is assumed
uninsulated in notional building.
        } // end else

//        JO 29 Sept 2010
//          Following the CEN Standard it is clear that we ought to
consider all the layers, including insulation layers, thus the
following code lines are commented

        //new default=> In Res Floor=0.17 m2K/W
        //new default=> Ex Res Floor=0.04 m2K/W
        //new default=> Floor 1=0.457            // a factor in the
CIBSE calculation

equiv thick=D.Value("Wall Width")+(D.Value("Earth Cond")*(D.Value("In
 Res Floor")+therm res+D.Value("Ex Res Floor")));

        //calculate the U-Value:
        if (equiv thick<B)         // this is usually the case: for
uninsulated and moderately insulated floors

mU VALUE=(2*D.Value("Earth Cond")/(M PI*B+equiv thick))*log((M PI*B/e
quiv thick)+1);
        else     // i.e., equiv_thick>=B, may occur for a small well-
insulated floor

mU_VALUE=D.Value("Earth_Cond")/(D.Value("Floor_1")*B+equiv_thick);
// equiv thick will always be > 0 so there is no risk of division by
0

        if((mU VALUE>mU VALUE MAX)&&(P.Vcod(gId,"ACT-NOT")=="NOT"))
          mU VALUE=mU VALUE MAX;
      }
      else
        mU VALUE=mU VALUE MAX;

  }
  //if it is not a ground floor, or if the input U-value has already
been corrected for heat loss of floor in contact with the ground
  else
        mU VALUE=mU VALUE MAX;

  {
   ostringstream res; res << mU VALUE;
   P.ModKey(Id,"U-VALUE",res.str());
  }
}

void WALL::fCM(PROJECTDefaults& D, BOC& P) {
  if(func report) cout << endl << "   " << "WALL::fCM";
//multiply themal capcity of envelope element by net area

  deque<double> mCM=P.Vlnum(P.Vrefid(Id,"CONSTRUCTION"),"CM");
```

```
//   if (P.Vcod(gId,"ACT-NOT")=="NOT") {   //adjust notional values
only:
    //if it is a NOTIONAL ground floor:
//    if ((P.Vcodid(Id,"TYPE-ENV")==3) && (P.Vcodid(Id,"TYPE")==5))
//      for(int i=0;i<2;i++) mCM[i]=D.Value("dCM_GROUND_F");

    //if it is a NOTIONAL external wall:
//    if ((P.Vcodid(Id,"TYPE-ENV")==1) && (P.Vcodid(Id,"TYPE")==1))
//      for(int i=0;i<2;i++) mCM[i]=D.Value("dCM EXT W");
//  }

  if (P.Vcodid(Id,"TYPE")==6)    // 6=Same space
    for(int i=0;i<2;i++)
      mCM[i]*=2*P.Vnum(Id,"NET-AREA");
  else // i.e., all other adjacencies of the wall
    for(int i=0;i<2;i++)
      mCM[i]*=P.Vnum(Id,"NET-AREA");

 {
  ostringstream res; res << "{ ";
  res << mCM[0] << ", ";
  res << mCM[1] << " }";
  P.ModKey(Id,"CM",res.str());      //  AN(08/08/07): This does not
include doors as they were excluded from NET-AREA in the function
fNET AREA above.
 }
}


void WALL::fH TR(PROJECTDefaults& D, BOC& P) {
  if(func report) cout << endl << "  " << "WALL::fH TR";
//calculate transmission heat transfer coefficient for envelope
element
//including contribution from windows, thermal bridges, doors and
smoke vents

  double mA, mH TR;
  double mNETAREA=P.Vnum(Id,"NET-AREA");
  double mU_VALUE=P.Vnum(Id,"U-VALUE");

  switch(P.Vcodid(Id,"TYPE")) {
  case 1: {                      //exterior
           mA=D.Value("D011");
          }
           break;
  case 2: {                     //strongly ventilated space
           mA=D.Value("D011");
          }
           break;
  case 3: {                      //unheated space
           mA=D.Value("D011");
          }
           break;
  case 4: {                      //conditioned space
           mA=0;
          }
           break;
  case 5: {                      //underground
//           mA=1/(1+mU VALUE);
// The line above has been changed as below so that the heat transfer
coefficient calculation for ground floors is the same as
// that for the other envelopes (except those adjacent to CAS where
coefficient is 0).
// NEN2916 says that the old equation is based on transmission heat
losses of floors of dwellings in which the effect of thermal
// bridges at edges is incorporated. For larger floor areas this can
lead to overestimation of heat losses. Hence, the change. - AN
15/08/2007
           mA=D.Value("D011");
```

```
          }
           break;
  case 6: {                    //Same space
           mA=0;
          }
           break;
  }

  mH TR=mA*mNETAREA*mU VALUE;

  for (int i=0;i<vW.size();i++)
    mH TR+=P.Vnum(vW[i].Id,"H-TR")*P.Vnum(vW[i].Id,"MULTIPLIER");
  for (int i=0;i<vTM.size();i++)
    mH_TR+=P.Vnum(vTM[i].Id,"H-TR")*P.Vnum(vTM[i].Id,"MULTIPLIER");
//adding doors:
  for (int i=0;i<vD.size();i++)
    mH_TR+=P.Vnum(vD[i].Id,"H-TR")*P.Vnum(vD[i].Id,"MULTIPLIER");
//adding smoke vents:
  for (int i=0;i<vSV.size();i++)
    mH_TR+=P.Vnum(vSV[i].Id,"H-TR")*P.Vnum(vSV[i].Id,"MULTIPLIER");

 {
  ostringstream res; res << mH TR;
  P.ModKey(Id,"H-TR",res.str());           //includes doors and smoke
vents
 }

// ID4 added:
//new keyword for wall -> overall U-value for wall including windows
 double mWALL WIN U=0.0;
 double mAREAwin=0.0;
 double mAREAALLwin=0.0;
 double mAREAdoor=0.0;

 for (int i=0;i<vW.size();i++) {
   mAREAwin=P.Vnum(vW[i].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER");
   mAREAALLwin+=mAREAwin;
   //Avg U-VALUE
   mWALL_WIN_U+=P.Vnum(vW[i].Id,"U-VALUE")*mAREAwin;
 }
 mWALL WIN U+=mU VALUE*mNETAREA;


 //area of wall - door and smoke vent area
 //Corrected in v1.2.a by JO
 //double mAREA WALL WIN=P.Vnum(Id,"AREA");
 //for      (int      i=0;i<vD.size();i++)             mAREA WALL WIN-
=P.Vnum(vD[i].Id,"AREA")*P.Vnum(vD[i].Id,"MULTIPLIER");
 //for        (int        i=0;i<vSV.size();i++)        mAREA WALL WIN-
=P.Vnum(vSV[i].Id,"AREA")*P.Vnum(vSV[i].Id,"MULTIPLIER");      //smoke
vents
 double mAREA WALL WIN=mNETAREA+mAREAALLwin;

 if(mAREA_WALL_WIN==0)
   mWALL WIN U=0.0;
 else
   mWALL_WIN_U/=mAREA_WALL_WIN;

 {
  ostringstream res; res << mWALL_WIN_U;
  P.ModKey(Id,"U-VALUE-WALL/WIN",res.str());
 }

// end ID4 added
}

void WALL::fQ SUN NT(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "   " << "WALL::fQ SUN NT";
```

```
  double mAREA=P.Vnum(Id,"NET-AREA");


  switch(P.Vcodid(Id,"TYPE")) {
  case 1: {                    //exterior
             // Q SUN
             double Q SUN[12];
             string mWEATHER;
             if(gId<0)             mWEATHER=" LON";             else
mWEATHER=" "+P.Vcod(gId,"WEATHER");
             for (int i=0;i<12;i++)

Q SUN[i]=T.Value("T010"+mWEATHER,i+1,P.Vcodid(Id,"ORIENTATION"));


             // Q SUN NT
             double mQ_SUN_NT[12];
             for (int i=0;i<12;i++)
               mQ SUN NT[i]=D.Value("D039")*Q SUN[i]*P.Vnum(Id,"U-
VALUE")*mAREA;

           //add for doors:
             deque<double> mQ SUN NT DOOR M;
             for (int j=0;j<vD.size();j++) {
               mQ SUN NT DOOR M=P.Vlnum Mul(vD[j].Id,"Q-SUN-
NT",P.Vnum(vD[j].Id,"MULTIPLIER"));
               for                  (int                 i=0;i<12;i++)
mQ SUN NT[i]+=mQ SUN NT DOOR M[i];
             }
           //add for SMOKE VENTS:
             deque<double> mQ SUN NT SV M;
             for (int j=0;j<vSV.size();j++) {
               mQ SUN NT SV M=P.Vlnum Mul(vSV[j].Id,"Q-SUN-
NT",P.Vnum(vSV[j].Id,"MULTIPLIER"));
               for                  (int                 i=0;i<12;i++)
mQ SUN NT[i]+=mQ SUN NT SV M[i];
             }

            {
             ostringstream res; res << "{ ";
             for (int i=0;i<11;i++)
               res << mQ_SUN_NT[i] << ", ";
             res << mQ SUN NT[11] << " }";
             P.ModKey(Id,"Q-SUN-NT",res.str());                //MJ,
includes doors and smoke vents
             }
          }
           break;
  default: {
             {
             ostringstream res; res << "{ ";
             for (int i=0;i<11;i++)
               res << 0.0 << ", ";
             res << 0.0 << " }";
             P.ModKey(Id,"Q-SUN-NT",res.str());                //MJ,
includes doors and smoke vents
             }
          }
  }

}

void WALL::fU_INF_WIND(PROJECTDefaults& D, PROJECTTables& T, BOC &P){
  if(func report) cout << endl << "  " << "WALL::fU INF WIND";
//infiltration due to stack and wind effects

  switch(P.Vcodid(Id,"TYPE")) {
  case 1: {                    //exterior
           double V_INF[12];
```

```
            string mWEATHER;

            if (gId<0) mWEATHER=" LON";
            else mWEATHER=" "+P.Vcod(gId,"WEATHER");
            for (int month=0; month<12; month++)

V INF[month]=T.Value("tVinf"+mWEATHER,month+1,P.Vcodid(Id,"ORIENTATIO
N"));

            //U INF WIND

            double mQ50 INF=P.Vnum(pId,"Q50-INF");
            deque<double>      mBR LIGHT CHECK=P.Vlnum(pId,"BR-LIGHT-
CHECK");
            double
mQ4 INF=mQ50 INF*T.Value("tn50Q4",1,(int)mBR LIGHT CHECK[7]);
//m3/hm2 outer envelope
            //double          mQ4 INF=mQ50 INF*T.Value("tn50Q4",1,1);
//m3/hm2 outer envelope
            double mU_INF_WIND_WALL[12];

// Code modified to apply a different dcp for roofs - AN - 13/03/07
            for (int month=0;month<12;month++) {
            // check if it's a roof and it's flat
              if          (((P.Vcodid(Id,"TYPE-ENV"))==2)          &&
((P.Vcodid(Id,"ROOF-PITCH"))<=2))  // 2=10 degrees, i.e., flat

mU INF WIND WALL[month]=D.Value("dU INF1")*mQ4 INF*pow(T.Value("tU IN
F3",1,2)*pow(V INF[month],2.0),D.Value("dU INF2"));  //  m3/hm2  outer
envelope
              // check if it's a roof and it's pitched
              else     if     (((P.Vcodid(Id,"TYPE-ENV"))==2)     &&
((P.Vcodid(Id,"FLAT-ROOF"))>2)) // 2=10 degrees, i.e., pitched

mU INF WIND WALL[month]=D.Value("dU INF1")*mQ4 INF*pow(T.Value("tU IN
F3",1,3)*pow(V INF[month],2.0),D.Value("dU INF2"));  //  m3/hm2  outer
envelope
              // Otherwise, it's a wall
                    else
mU INF WIND WALL[month]=D.Value("dU INF1")*mQ4 INF*pow(T.Value("tU IN
F3",1,1)*pow(V INF[month],2.0),D.Value("dU INF2"));  //  m3/hm2  outer
envelope
            }
             {
              ostringstream res; res << "{ ";
              for (int i=0;i<11;i++)
                res << mU INF WIND WALL[i] << ", ";
              res << mU_INF_WIND_WALL[11] << " }";
              P.ModKey(Id,"U-INF-WIND-WALL",res.str());
             }

            //U INF STACK

            double mU_INF_STACK_WALL[12];
            deque<double>      mHEAT SET POINT=P.Vlnum(pId,"HEAT-SET-
POINT");

            for (int month=0;month<12;month++) {
              if(mHEAT SET POINT[month]!=-9999)

mU INF STACK WALL[month]=D.Value("dU INF4")*mQ4 INF*pow(D.Value("dU I
NF5")*P.Vnum(pId,"HEIGHT")*fabs(T.Value("T003"+mWEATHER,1,month+1)-
mHEAT_SET_POINT[month]),D.Value("dU_INF6"));         //  m3/hm2  outer
enevelope
              else
                mU_INF_STACK_WALL[month]=0;
            }
             {
              ostringstream res; res << "{ ";
```

```
              for (int i=0;i<11;i++)
                res << mU INF STACK WALL[i] << ", ";
              res << mU INF STACK WALL[11] << " }";
              P.ModKey(Id,"U-INF-STACK-WALL",res.str());
             }
          }
          break;
  default: {
             {
              ostringstream res; res << "{ ";
              for (int i=0;i<11;i++)
                res << 0.0 << ", ";
              res << 0.0 << " }";
              P.ModKey(Id,"U-INF-WIND-WALL",res.str());
             }
             {
              ostringstream res; res << "{ ";
              for (int i=0;i<11;i++)
                res << 0.0 << ", ";
              res << 0.0 << " }";
              P.ModKey(Id,"U-INF-STACK-WALL",res.str());
             }
          }
  }

}

void WALL::Recommendation(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func report) cout << endl << "  " << "WALL::Recommendation";
  // First Child Recommendation?
  //all envelope thermal bridges
  for (int i=0; i<vTM.size();i++)
    vTM[i].Recommendation(D,T,P);
  //all windows in envelope element
  for (int i=0; i<vW.size();i++)
    vW[i].Recommendation(D,T,P);
  //all doors in envelope element
  for (int i=0; i<vD.size();i++)
    vD[i].Recommendation(D,T,P);

  // Then Wall Recommendation?


}
```

WALL-TM-BRIDGE

Specifies the linear thermal transmittance and length of a thermal bridge associated with a WALL object.

Example input:

```
"TM1" = WALL-TM-BRIDGE
  PSI             = 2
  L               = .4
  ..
```

PSI

Specifies the linear thermal transmittance of a thermal bridge, in W/(mK). The default value is 2.

L

Specifies the length of an individual linear thermal bridge, in m. The default value is 10.

MULTIPLIER
This keyword specifies how many linear thermal bridges with this exact description exist in the current envelope element. E.g., if the value of the multiplier is 3, there are 3 linear thermal bridges each with identical keyword values in the envelope considered. This enables repeating linear thermal bridges to be entered quickly and easily. The default value is 1.

c++ code

```
//: tm bridge.cpp

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void TM BRIDGE::TEST() {
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test TM B" << endl;
//  cout << "Fin de Test TM B" << endl;
  BEEMObject::TEST();
}

// v4.3.0 Addition of
// TM BRIDGE::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P,
BOC&N, int i)
void  TM BRIDGE::Notional(PROJECTDefaults&  D,  PROJECTTables&  T,  BOC
&P, BOC &N, int i) {
  if(func report) cout << endl << "   " << "TM BRIDGE::Notional " <<
i;

  N.vINP.push back(P.vINP[i]);
}
// end v4.3.0 Addition of
// TM BRIDGE::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P,
BOC&N, int i)

void TM BRIDGE::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "  " << "TM_BRIDGE::Calc";
  if(calc report) cout << endl << P.vINP[Id].Name;
    fH TR(P);
}

void TM BRIDGE::Init(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "  " << "TM_BRIDGE::Init";
//  fH TR(P);
}

void TM BRIDGE::fH TR(BOC& P) {
  if(func report) cout << endl << "  " << "TM BRIDGE::fH TR";
  double mH_TR=P.Vnum(Id, "PSI")*P.Vnum(Id, "L");

  {
  ostringstream res; res << mH_TR;
  P.ModKey(Id,"H-TR",res.str());
  }

}

void  TM_BRIDGE::Recommendation(PROJECTDefaults&  D,  PROJECTTables&  T,
BOC& P) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"TM_BRIDGE::Recommendation";
```

```
}


// End Object: TM_BRIDGE
```

WIND-GENERATOR

Specifies the wind generator parameters.

Example input:

```
"Wind Generator 1" = WIND-GENERATOR
   MULTIPLIER          = 1
   TERRAIN-TYPE        = Suburban or industrial area
   AREA                = 5
   HUB-HEIGHT          = 10
   POWER               = 50
   ..
```

***DIAMETER* (NO LONGER IN USE)**

This is the wind turbine rotor diameter, in metres. This is a required keyword.

HUB-HEIGHT

Specifies the wind turbine hub height, in metres. The default value is 10.

AREA

Specifies the area swept by the rotor blades of the wind turbine, in m^2. This is a required keyword.

POWER

This is the wind turbine rated power (electrical power delivered at rated wind speed), in kW. The default value is 50.

TERRAIN-TYPE

Specifies the terrain type where the wind turbine is located, as one of the values listed below. The default is 'Suburban or industrial area'.

Smooth flat country (no obstacles)
Farm land with boundary hedges
Suburban or industrial area
Urban with average building height > 15m

MULTIPLIER

This keyword specifies how many wind generators with this exact description exist in the project. E.g., if the value of the multiplier is 3, there are 3 WIND-GENERATORS, each with identical keyword values in the project. This enables repeating wind generators to be entered quickly and easily. The default value is 1.

c++ code

```
//: wind generator.cpp
```

```
#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void WIND GENERATOR::TEST() {
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test WG" << endl;
//  cout << "Fin de Test WG" << endl;
  BEEMObject::TEST();
}

void WIND GENERATOR::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func report) cout << endl << "  " << "WIND GENERATOR::Calc";
  if(calc_report) cout << endl << P.vINP[Id].Name;
// its own calcs below...
  fKWH DISP(D, T, P);
}

void WIND GENERATOR::fKWH DISP(PROJECTDefaults& D, PROJECTTables& T,
BOC& P) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"WIND GENERATOR::fKWH DISP";
//calculate energy displaced by Wind generator (MJ and kWh)

  string mWEATHER;
  double mKWH=0;
//  double mAREA=3.14159265358979/4*pow(P.Vnum(Id,"DIAMETER"),2);
  double mAREA=P.Vnum(Id,"AREA");
/*
 {
  ostringstream res; res << mAREA;
  P.ModKey(Id,"AREA",res.str());
 }
*/
  if(gId<0) mWEATHER=" LON"; else mWEATHER=" "+P.Vcod(gId,"WEATHER");

  double mSPEED CORR M[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  double mAVG POWER DENS M[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
};
  double mEFF M[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  double mKWH M[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };

  int col=2;
  if (P.Vnum(Id,"POWER")<D.Value("dWind size")) col=1;

  for (int m=0;m<12;m++) {
     //old                                                    version
mSPEED_CORR_M[m]=T.Value("tWind_Vgen"+mWEATHER,1,m+1)*
                    //pow((P.Vnum(Id,"HUB-
HEIGHT")/D.Value("dWind height")),(T.Value("tWind Shear",1,P.Vcodid(I
d,"TERRAIN-TYPE"))));

     //modified 22-03-06 TR

mSPEED CORR M[m]=T.Value("tWind Vgen"+mWEATHER,1,m+1)*(T.Value("tTerr
ainFactorKR",1,P.Vcodid(Id,"TERRAIN-TYPE")))*log(P.Vnum(Id,"HUB-
HEIGHT")/(T.Value("tRoughnessZo",1,P.Vcodid(Id,"TERRAIN-TYPE"))));


mAVG_POWER_DENS_M[m]=0.5*D.Value("dWind_airdens")*pow(mSPEED_CORR_M[m
],3)*T.Value("tWind EPF"+mWEATHER,1,m+1);

    if       (mSPEED CORR M[m]>=0       &&       mSPEED CORR M[m]<3)
mEFF M[m]=T.Value("tWind EFF",1,col);
    else     if     (mSPEED CORR M[m]>=3     &&     mSPEED CORR M[m]<4)
mEFF_M[m]=T.Value("tWind_EFF",2,col);
```

```
    else     if     (mSPEED CORR M[m]>=4     &&     mSPEED CORR M[m]<5)
mEFF M[m]=T.Value("tWind EFF",3,col);
    else     if     (mSPEED CORR M[m]>=5     &&     mSPEED CORR M[m]<6)
mEFF M[m]=T.Value("tWind EFF",4,col);
    else     if     (mSPEED CORR M[m]>=6     &&     mSPEED CORR M[m]<7)
mEFF M[m]=T.Value("tWind EFF",5,col);
    else     if     (mSPEED CORR M[m]>=7     &&     mSPEED CORR M[m]<8)
mEFF_M[m]=T.Value("tWind_EFF",6,col);
    else     if     (mSPEED CORR M[m]>=8     &&     mSPEED CORR M[m]<9)
mEFF M[m]=T.Value("tWind EFF",7,col);
    else                    if                   (mSPEED CORR M[m]>=9)
mEFF M[m]=T.Value("tWind EFF",8,col);
    else
mEFF_M[m]=T.Value("tWind_EFF",8,col);


mKWH_M[m]=mAREA*mAVG_POWER_DENS_M[m]*mEFF_M[m]/1000*T.Value("NJ",1,m+
1)*24;
    mKWH+=mKWH M[m];
  }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mSPEED CORR M[i] << ", ";
  res << mSPEED CORR M[11] << " }";
  P.ModKey(Id,"SPEED-CORR-M",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mAVG POWER DENS M[i] << ", ";
  res << mAVG POWER DENS M[11] << " }";
  P.ModKey(Id,"AVG-POWER-DENS-M",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mEFF_M[i] << ", ";
  res << mEFF M[11] << " }";
  P.ModKey(Id,"EFF-M",res.str());
 }

  // Adjust for proportion applicable to core
  double mSC_ACTUAL_RATIO=P.Vnum(cId,"SC-ACTUAL-RATIO");
  for (int m=0;m<12;m++)
    mKWH M[m]*=mSC ACTUAL RATIO;

  mKWH*=mSC ACTUAL RATIO;


 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH M[i] << ", ";
  res << mKWH M[11] << " }";
  P.ModKey(Id,"KWH-DISP-M",res.str());
 }
 {
  ostringstream res; res << mKWH;
  P.ModKey(Id,"KWH-DISP",res.str());
 }
 double mCO2_factor;
 if((P.Vcodid(cId,"BR-TYPE")==5) || (P.Vcodid(cId,"EPC-TYPE")==6))
    mCO2 factor=T.Value("tKGCO2 IR",1,11);
 else if((P.Vcodid(cId,"BR-TYPE")==6))
    mCO2 factor=T.Value("tKGCO2 JER",1,11);
 else // all UK
    mCO2_factor=T.Value("tKGCO2",1,11);
```

```
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_M[i]*mCO2_factor << ", ";
  res << mKWH_M[11]*mCO2_factor << " }";
  P.ModKey(Id,"KG-CO2-DISP-M",res.str());
 }
 {
  ostringstream res; res << mKWH*mCO2_factor;
  P.ModKey(Id,"KG-CO2-DISP",res.str());
 }

//cout << "SPEED-CORR-M: " << P.Vtext(Id,"SPEED-CORR-M") << endl;
//cout << "AVG-POWER-DENS-M: " << P.Vtext(Id,"AVG-POWER-DENS-M") <<
endl;
//cout << "EFF-M: " << P.Vtext(Id,"EFF-M") << endl;

}

void        WIND GENERATOR::Recommendation(PROJECTDefaults&        D,
PROJECTTables& T, BOC& P) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"WIND GENERATOR::Recommendation";

}



// End Object: WIND GENERATOR
```

WINDOW

Specifies the size and glazing properties of a window or rooflight. Each WINDOW command applies to the WALL instruction preceding it and describes a window or rooflight in that envelope element. A GLASS and a WALL instructions must precede a WINDOW instruction. A U-name must be used for output reports.

Example input:

```
"Simple glass" = GLASS
  ...
  ..
"RIGHT" = WALL
  ...
  ..
"RIGHT-W1" = WINDOW
  AREA                = 6
  DEV-PROJ-RATIO      = 1
  GLASS               = "Simple glass"
  SHADING-SYSTEM      = Ext. solar protection. User moveable
  SHADING-FACTOR      = 0.8
  DISP-WIND           = NO
  FRAME-FACTOR        = 0.1
  ROOF-LIGHT-COVER    = 1
  ASPECT-RATIO        = 0.7
  ..
```

AREA
Specifies the projected area of the window or rooflight, in m^2 (i.e., area of the structural opening in the envelope element). This is a required keyword.

DEV-PROJ-RATIO
Specifies the ratio between the developed area and the projected area of the window or rooflight, in decimal form. The default value is 1 for a window and 1.3 for a rooflight.

FRAME-FACTOR
Specifies the ratio of the window or rooflight area which is occupied by the frame, in decimal form. The default value is 0.1 for a window and 0.3 for a rooflight.

ASPECT-RATIO
Specifies the ratio of the window's (or rooflight's) height to its width, in decimal form.

GLASS
Specifies the U-name of a previously-defined GLASS instruction that defines the construction of the window element. This is a required keyword.

SHADING-SYSTEM
Specifies the shading system available for the window or rooflight for the purposes of calculating the reduction factor due to moveable solar protection devices, as one of the values listed below. The default is 'All other cases'.

Value	Description
Ext. solar protection. User moveable	External, user-moveable solar protection devices.
Ext. solar protection with automatic control	External, automatically-controlled solar protection devices.
All other cases	All other cases

SHADING-FACTOR
Specifies the shading correction (reduction) factor due to overhangs or fins, in decimal form. In BS EN ISO 13790:2008, it is determined according to section G.5.4.3. The default value is 1 (i.e., no shading from fins or overhangs).

DISP-WIND
Indicates whether the window is a display window. If YES, the window is a display window. If not, the keyword should have the value NO. The default is NO.

MULTIPLIER
This keyword specifies how many windows or rooflight with this exact description exist in the current envelope element. E.g. if the value of the multiplier is 3, there are 3 windows or rooflights, each with identical keyword values in the envelope considered. This enables repeating windows or rooflights to be entered quickly and easily. The default value is 1.

ROOF-LIGHT-COVER
Specifies the ratio between the area of the floor covered by an array of rooflights to the total area of rooflight glazing, in decimal form. The default value is 1.

c++ code

```
//: window.cpp

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void WINDOW::TEST() {
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test WINDOW" << endl;
//  cout << "Fin de Test WINDOW" << endl;
  BEEMObject::TEST();
  for(int i=0;i<vTM.size();i++)
    vTM[i].TEST();
}

void WINDOW::Reset() {
  if(func report) cout << endl << "  " << "WINDOW::Reset";
  dTM.clear();
//  vTM.clear();
}

void WINDOW::DtoV() {
  if(func report) cout << endl << "  " << "WINDOW::DtoV";
//  vTM.reserve(dTM.size());
//  vTM.assign(dTM.begin(),dTM.end());
//  dTM.clear();
}


void WINDOW::NotionalDisp(PROJECTDefaults& D, PROJECTTables& T, BOC
&P, BOC &N, double mAREA, double mASPECT RATIO, int not case) {
  if(func report) cout << endl << "  " << "WINDOW::NotionalDisp " <<
mAREA;
//creates notional display window

  char C=34;
  BObject Temp;
  Temp.Ntype=6;
  Temp.Type="WINDOW";
  Temp.Name=P.vINP[Id].Name + " NotDisp Win";
  KeywordValue KV;

 {
  ostringstream res; res << "YES";
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("DISP-WIND",KV));
 }
 {
  ostringstream res; res << C << "Notional Display Glass" << C;
  KV.Reset(); KV.Init(3,res.str());
  Temp.Prop.insert(make pair("GLASS",KV));
 }
 {
  ostringstream res; res << mAREA;
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make_pair("AREA",KV));
 }
 {
  ostringstream res; res << mASPECT_RATIO;
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("ASPECT-RATIO",KV));
 }
 {
  ostringstream res; res << 1;           //this is already considered
```

```
when calculating area
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("MULTIPLIER",KV));
 }
//now using defaults for shading system and shading factor for
notional display window:

 { // display windows can only be in walls (not roofs) so row 1 of
table Not Frame Factor is used
  ostringstream res; res << T.Value("Not Frame Factor",1,not case);
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("FRAME-FACTOR",KV));
 }
 { // display windows can only be in walls (not roofs)
  ostringstream res; res << D.Value("dDev Proj Win");
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make_pair("DEV-PROJ-RATIO",KV));
 }

  N.vINP.push_back(Temp);
}

void WINDOW::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P,
BOC &N, string mWALLNAME, double mAREA, double mASPECT RATIO, int
TYPE ENV, int not case) {
  if(func report) cout << endl << "   " << "WINDOW::Notional   " <<
mWALLNAME << "  " << mAREA << "  " << TYPE ENV;
//creates notional window

  char C=34;
  BObject Temp;
  Temp.Ntype=6;
  Temp.Type="WINDOW";
  Temp.Name=mWALLNAME + " Not Win";
  KeywordValue KV;

//added - developed to projected ratio is 1 for notional window and
1.3 for notional rooflight:
  switch(TYPE_ENV) {
      case  1: { // 1=wall
       {
        ostringstream res; res << C << "Notional Glass" << C;
        KV.Reset(); KV.Init(3,res.str());
        Temp.Prop.insert(make pair("GLASS",KV));
       }
       {
        ostringstream res; res << D.Value("dDev Proj Win");
        KV.Reset(); KV.Init(1,res.str());
        Temp.Prop.insert(make pair("DEV-PROJ-RATIO",KV));
       }
       {
        ostringstream             res;             res             <<
T.Value("Not Frame Factor",1,not case);
        KV.Reset(); KV.Init(1,res.str());
        Temp.Prop.insert(make pair("FRAME-FACTOR",KV));
       }
      }
      break;
      case  2: { // 2=roof
       if (not case>=9)  // if this notional building is for E, W, or
NI
          {
           ostringstream res; res << C << "Notional Rooflight Glass"
<< C;
           KV.Reset(); KV.Init(3,res.str());
           Temp.Prop.insert(make_pair("GLASS",KV));
          }
       else // other countries
          {
```

```
            ostringstream res; res << C << "Notional Glass" << C;
            KV.Reset(); KV.Init(3,res.str());
            Temp.Prop.insert(make pair("GLASS",KV));
           }
        {
         ostringstream res; res << D.Value("dDev Proj Roof");
         KV.Reset(); KV.Init(1,res.str());
         Temp.Prop.insert(make_pair("DEV-PROJ-RATIO",KV));
        }
        {
         ostringstream             res;             res             <<
T.Value("Not Frame Factor",2,not case);
         KV.Reset(); KV.Init(1,res.str());
         Temp.Prop.insert(make_pair("FRAME-FACTOR",KV));
        }
        {
         ostringstream             res;             res             <<
T.Value("Not Rooflight Cover",1,not case);
         KV.Reset(); KV.Init(1,res.str());
         Temp.Prop.insert(make_pair("ROOF-LIGHT-COVER",KV));
        }
       }
       break;
   }
  {
   ostringstream res; res << mAREA;
   KV.Reset(); KV.Init(1,res.str());
   Temp.Prop.insert(make pair("AREA",KV));
  }
  {
   ostringstream res; res << mASPECT RATIO;
   KV.Reset(); KV.Init(1,res.str());
   Temp.Prop.insert(make pair("ASPECT-RATIO",KV));
  }

   N.vINP.push back(Temp);

}

void WINDOW::Init(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "WINDOW::Init";

  vTM.clear();

  double mLIG_SOL_TRANS;
  double mTOT SOL TRANS;
  double mLIG SOL TRANS ORI;       // AN 29/08/07: a new parameter to
save the original normal incidence light transmittance before any
corrections

  // Check if this is the 2010 notional building for England and
Wales
  if ( (P.Vcod(gId,"ACT-NOT")=="NOT") && (P.Vint(ppId,"NOT-CASE")>=9)
) {
    mLIG SOL TRANS ORI=T.Value("NotGlass L Value",1,P.Vint(ppId,"NOT-
CASE"));
    mLIG_SOL_TRANS=mLIG_SOL_TRANS_ORI;
    mTOT SOL TRANS=T.Value("NotGlass G Value",1,P.Vint(ppId,"NOT-
CASE"));
  } // end if
  else { // i.e., actual building or other notional buildings
    if (P.Vcod(Id,"DISP-WIND")=="YES") {
      // if the calculation is for Scotland
      if(    (P.Vcodid(cId,"BR-TYPE")==3)    ||    (P.Vcodid(cId,"EPC-
TYPE")==4) )
        mLIG_SOL_TRANS=0.0;
      else                                                          {
mLIG SOL TRANS ORI=P.Vnum(P.Vrefid(Id,"GLASS"),"LIG-SOL-TRANS");
        mLIG_SOL_TRANS=mLIG_SOL_TRANS_ORI;
```

```
      } // end else
    } // end if
    else { //i.e., not a display window
      mLIG SOL TRANS ORI=P.Vnum(P.Vrefid(Id,"GLASS"),"LIG-SOL-
TRANS");
      mLIG SOL TRANS=mLIG SOL TRANS ORI;
    }
    mTOT_SOL_TRANS=P.Vnum(P.Vrefid(Id,"GLASS"),"TOT-SOL-TRANS");
  }// end else

  // If this is the actual or the 2010 notional
  if    (    (P.Vcod(gId,"ACT-NOT")=="ACT")    ||    ((P.Vcod(gId,"ACT-
NOT")=="NOT") && (P.Vint(ppId,"NOT-CASE")>=9)) ) {
    // Adjust the transmittances based on the orientation and pitch
of the window or rooflight (of parent envelope)
    mLIG SOL TRANS*=T.Value("tG Value Adjust",P.Vcodid(pId,"ROOF-
PITCH"),P.Vcodid(pId,"ORIENTATION"));
    mTOT SOL TRANS*=T.Value("tG Value Adjust",P.Vcodid(pId,"ROOF-
PITCH"),P.Vcodid(pId,"ORIENTATION"));
  } // end if
  else { // i.e., other notional buildings
    //apply transmission factors to NOTIONAL ROOFLIGHTS AND WINDOWS
as described in scenario 2
    switch(P.Vcodid(pId,"TYPE-ENV")) {
      case  1: {
        //window
        //check orientation of parent wall
        //all vertical glazing from E through to W via S has reduced
transmittance by Not Window T factor
        //if                        ((P.Vcod(pId,"ORIENTATION")!="North-
East")&&(P.Vcod(pId,"ORIENTATION")!="North")&&(P.Vcod(pId,"ORIENTATIO
N")!="North-West")) {
        // AN - 18/03/08 - The above line has been modified to fit
with the NCM Modelling Guide so that the orienations with
        // the reduced transmittance (by Not Window T factor) do NOT
include South.
        if                        ((P.Vcod(pId,"ORIENTATION")!="North-
East")&&(P.Vcod(pId,"ORIENTATION")!="North")&&(P.Vcod(pId,"ORIENTATIO
N")!="South")&&(P.Vcod(pId,"ORIENTATION")!="North-West")) {
          mLIG SOL TRANS*=D.Value("Not Window T factor");
          mTOT SOL TRANS*=D.Value("Not Window T factor");
          mLIG_SOL_TRANS_ORI*=D.Value("Not_Window_T_factor");
        }
      }
      break;
      case  2: {
        //rooflight
        mLIG_SOL_TRANS*=D.Value("Not_Rooflight_T_factor");
        mTOT SOL TRANS*=D.Value("Not Rooflight T factor");
        mLIG SOL TRANS ORI*=D.Value("Not Rooflight T factor");
      }
      break;
    } // end switch

    // multiply by 0.9 to obtain figure averaged over all angles,
except mLIG SOL TRANS ORI which needs to remain as
    // normal incidence
    mLIG SOL TRANS*=D.Value("dGlazingGfactor");
    mTOT SOL TRANS*=D.Value("dGlazingGfactor");
  }// end else



 {
  ostringstream res; res << mLIG SOL TRANS;
  P.ModKey(Id,"LIG-SOL-TRANS",res.str());
 }
 {
  ostringstream res; res << mTOT_SOL_TRANS;
```

```
  P.ModKey(Id,"TOT-SOL-TRANS",res.str());
 }
 // Save the original normal incidence light transmittance before
adjusting for different angles of incidence
 {
  ostringstream res; res << mLIG SOL TRANS ORI;
  P.ModKey(Id,"LIG-SOL-TRANS-ORI",res.str());
 }

  // check if frame factor was not included in the input file for the
actual building, and if that is the case, use
  // the appropriate default value
  if (P.Vcod(gId,"ACT-NOT")=="ACT") {
    // check if frame factor was not included in the input file for
the actual building, and if that is the case, use
    // the appropriate default value
    if (P.Vnum(Id,"FRAME-FACTOR")==-6666) {
      switch(P.Vcodid(pId,"TYPE-ENV")) {
        case  1: { // wall
          {
           ostringstream res; res << D.Value("dFrame Fac Win");
           P.ModKey(Id,"FRAME-FACTOR",res.str());
          }
        }  // end case 1
        break;
        case  2: { // roof
          //rooflight
          {
           ostringstream res; res << D.Value("dFrame Fac Roof");
           P.ModKey(Id,"FRAME-FACTOR",res.str());
          }
        } // end case 2
        break;
      }// end switch
    } // end if

    // check if developed to projected are was not included in the
input file for the actual building, and if that is
    // the case, use the appropriate default value
    if (P.Vnum(Id,"DEV-PROJ-RATIO")==-6666) {
      switch(P.Vcodid(pId,"TYPE-ENV")) {
        case  1: { // wall
          {
           ostringstream res; res << D.Value("dDev Proj Win");
           P.ModKey(Id,"DEV-PROJ-RATIO",res.str());
          }
        }  // end case 1
        break;
        case  2: { // roof
          //rooflight
          {
           ostringstream res; res << D.Value("dDev Proj Roof");
           P.ModKey(Id,"DEV-PROJ-RATIO",res.str());
          }
        } // end case 2
        break;
      }// end switch
    } // end if

  } // end if

}

void WINDOW::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "   " << "WINDOW::Calc";
  if(calc_report) cout << endl << P.vINP[Id].Name;
//calculations for windows

  //for all thermal bridges in window
```

```
  for (int i=0; i<vTM.size();i++)
    vTM[i].Calc(D, T, P);

//its own calc below...
  //u-value corrections for inclination
  fU VALUE(D, T, P);
  //calculate transmission heat transfer coefficient for window
  //including contribution from thermal bridges:
  fH TR(D, P);
  //calculate solar gain through window:
  fQ_SUN_T(D, T, P);
}


void WINDOW::fU VALUE(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "WINDOW::fU VALUE";
//u-value corrections for inclination

  double mU VALUE;
  double mU_VALUE_VERT;
//Ensure U values for window/ rooflights used in calculations are
those applicable to the inclination
//Use a correction for pitched roofs and flat roofs, irrespective of
the type of glazing used

  double Win Cor incl=0.0;

  //if it is a roof:
  if (P.Vcod(pId,"TYPE-ENV")=="Roof")
    //if it is a flat roof use a correction of +0.3:
    if (P.Vcodid(pId,"ROOF-PITCH")<=2) // 2 = 10 degrees, i.e., flat
roof
      Win Cor incl=D.Value("DWin Cor Flat");
    //if it is a pitched roof use a correction of +0.2:
    else // i.e., pitched roof
      Win Cor incl=D.Value("DWin Cor Pitch");


  // AN (25/11/08): The U-value for a display window modified from
above to below

  // Check if this is the notional building, and retrieve the U-value
from the table
  // For the 2010 notional, the U-value differs according to the
DRIVER2A keyword of the parent zone
  if (P.Vcod(gId,"ACT-NOT")=="NOT")   {
    if (P.Vcod(Id,"DISP-WIND")=="YES")
      mU_VALUE=T.Value("NotGlass_U_Value",2,P.Vint(ppId,"NOT-
CASE"))+Win Cor incl;
    else { // i.e., not a display window
      mU_VALUE_VERT=T.Value("NotGlass_U_Value",1,P.Vint(ppId,"NOT-
CASE"));
      mU VALUE=mU VALUE VERT+Win Cor incl;
    } // end else
  } // end if
  else { // i.e., this is the actual building
    // In the actual building, the U-value of the window (whether
display or not) is as defined for its GLASS except
    // for Scotland (display) where it is the same as for the
notional building
    if(    ((P.Vcodid(cId,"BR-TYPE")==3)    ||    (P.Vcodid(cId,"EPC-
TYPE")==4))  &&  P.Vcod(Id,"DISP-WIND")=="YES")  // if the calculation
is for Scotland and Display
      mU VALUE VERT=T.Value("NotGlass U Value",2,P.Vint(ppId,"NOT-
CASE"));
    else // all other calcualtions
      mU VALUE VERT=P.Vnum(P.Vrefid(Id,"GLASS"),"U-VALUE");
    mU VALUE=mU VALUE VERT+Win Cor incl;
  } // end else
```

```
 {
  ostringstream res; res << mU VALUE;
  P.ModKey(Id,"U-VALUE",res.str());
 }
 {
  ostringstream res; res << mU VALUE VERT;
  P.ModKey(Id,"U-VALUE-VERT",res.str());
 }

}


void WINDOW::fH_TR(PROJECTDefaults& D, BOC& P) {
  if(func report) cout << endl << "  " << "WINDOW::fH TR";
//calculate transmission heat transfer coefficient for window
//including contribution from thermal bridges:

  double mA, mH TR;
  double mAREA=P.Vnum(Id,"AREA");
  double mDEV PROJ RATIO=P.Vnum(Id,"DEV-PROJ-RATIO");
  double mU VALUE=P.Vnum(Id,"U-VALUE");

  switch(P.Vcodid(pId,"TYPE")) {
  case 1: {                    //exterior
           mA=D.Value("D011");
          }
           break;
  case 2: {                    //strongly ventilated space
           mA=D.Value("D011");
          }
           break;
  case 3: {                    //unheated space
           mA=D.Value("D011");
          }
           break;
  case 4: {                    //conditioned space
           mA=0;
          }
           break;
  case 5: {                    //underground
//           mA=1/(1+mU_VALUE);
// The line above has been changed as below so that the heat transfer
coefficient calculation for ground floors is the same as
// that for the other envelopes (except those adjacent to CAS where
coefficient is 0).
// NEN2916 says that the old equation is based on transmission heat
losses of floors of dwellings in which the effect of thermal
// bridges at edges is incorporated. For larger floor areas this can
lead to overestimation of heat losses. Hence, the change. - AN
15/08/2007
           mA=D.Value("D011");
          }
           break;
  case 6: {                    //Same space
           mA=0;
          }
           break;
  }

  mH TR=mA*mAREA*mDEV PROJ RATIO*mU VALUE;

  for (int i=0;i<vTM.size();i++)
    mH TR+=P.Vnum(vTM[i].Id,"H-TR")*P.Vnum(vTM[i].Id,"MULTIPLIER");

 {
  ostringstream res; res << mH TR;
  P.ModKey(Id,"H-TR",res.str());
 }
```

```
}

void WINDOW::fQ SUN T(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "WINDOW::fQ SUN T";
//calculate solar gain through window:

  double mAREA=P.Vnum(Id,"AREA");

  // Q SUN
  double Q SUN[12];
  string mWEATHER;
  if(gId<0) mWEATHER=" LON"; else mWEATHER=" "+P.Vcod(gId,"WEATHER");
  for (int i=0;i<12;i++)

Q SUN[i]=T.Value("T010"+mWEATHER,i+1,P.Vcodid(pId,"ORIENTATION"));

  // F_SUN
  double F SUN[12];
  for (int i=0;i<12;i++)
    if ((i>=0 && i<=3) || (i>=9))
      F SUN[i]=T.Value("T011",P.Vcodid(Id,"SHADING-SYSTEM"),1);
    else
      F SUN[i]=T.Value("T011",P.Vcodid(Id,"SHADING-SYSTEM"),2);

  // SHADING FACTOR
  double mSHADING FACTOR=P.Vnum(Id,"SHADING-FACTOR");

  // ENERGY-TRANS
//  Total solar energy transmittance, already corrected by 0.9 factor
in WINDOW::Init to include all angles of incidence
  double mENERGY TRANS=P.Vnum(Id,"TOT-SOL-TRANS");
  // Q SUN T
  double mQ SUN T[12];
  double mAREA G;

  mAREA G=mSHADING FACTOR*mENERGY TRANS*mAREA*(1-P.Vnum(Id,"FRAME-
FACTOR"));

  for (int i=0;i<12;i++)
    mQ SUN T[i]=Q SUN[i]*F SUN[i]*mAREA G;
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ SUN T[i] << ", ";
  res << mQ_SUN_T[11] << " }";
  P.ModKey(Id,"Q-SUN-T",res.str());
 }
 {
  ostringstream               res;               res               <<
mAREA G*T.Value("T011",P.Vcodid(Id,"SHADING-SYSTEM"),2);           //
Corrected with F_SUN for September (check)
  P.ModKey(Id,"AREA-G",res.str());
 }
}

void  WINDOW::Recommendation(PROJECTDefaults&  D,  PROJECTTables&  T,
BOC& P) {
  if(func report) cout << endl << "  " << "WINDOW::Recommendation";
  // First Child Recommendation?
  //all window thermal bridges
  for (int i=0; i<vTM.size();i++)
    vTM[i].Recommendation(D,T,P);

  // Then Window Recommendation?

}


// End Object: WINDOW
```

WINDOW-TM-BRIDGE

Specifies the linear thermal transmittance and length of a thermal bridge associated with a window or rooflight.

Example input:

```
"TM1" = WINDOW-TM-BRIDGE
  PSI             = 2
  L              = .4
  ..
```

PSI

Specifies the linear thermal transmittance of a thermal bridge, in W/(mK). This is a required keyword. The default value is 2.

L

Specifies the length of the individual linear thermal bridge, in m. This is a required keyword. The default value is 10.

MULTIPLIER

This keyword specifies how many linear thermal bridges with this exact description exist in the current window or rooflight. E.g., if the value of the multiplier is 3, there are 3 linear thermal bridges, each with identical keyword values in the window or rooflight considered. This enables repeating linear thermal bridges to be entered quickly and easily. The default value is 1.

c++ code

```
//: tm bridge.cpp

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void TM BRIDGE::TEST() {
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test TM B" << endl;
//  cout << "Fin de Test TM B" << endl;
  BEEMObject::TEST();
}

// v4.3.0 Addition of
// TM BRIDGE::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P,
BOC&N, int i)
void TM BRIDGE::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC
&P, BOC &N, int i) {
  if(func report) cout << endl << "   " << "TM BRIDGE::Notional " <<
i;

  N.vINP.push back(P.vINP[i]);
}
// end v4.3.0 Addition of
// TM BRIDGE::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P,
BOC&N, int i)
```

```
void TM BRIDGE::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "TM BRIDGE::Calc";
  if(calc report) cout << endl << P.vINP[Id].Name;
    fH_TR(P);
}

void TM_BRIDGE::Init(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "TM BRIDGE::Init";
//  fH TR(P);
}

void TM BRIDGE::fH TR(BOC& P) {
  if(func_report) cout << endl << "  " << "TM_BRIDGE::fH_TR";
  double mH TR=P.Vnum(Id, "PSI")*P.Vnum(Id, "L");

  {
  ostringstream res; res << mH TR;
  P.ModKey(Id,"H-TR",res.str());
  }

}

void TM BRIDGE::Recommendation(PROJECTDefaults& D, PROJECTTables& T,
BOC& P) {
  if(func report)    cout    <<    endl    <<    "         "    <<
"TM BRIDGE::Recommendation";

}



// End Object: TM BRIDGE
```

ZONE

Specifies information associated with a space or 'zone'. Includes information on space loads and HVAC systems. A U-name must be specified, but it is only used for output reports. It is the second object in the hierarchy of geometry definitions (after HVAC-SYSTEM), followed by WALL, (WALL-TM-BRIDGE), WINDOW, (WINDOW-TM-BRIDGE), or DOOR, (DOOR-TM-BRIDGE).

Example input:

```
"Zone 1" = ZONE
   ISBEM-ID              = 11
   ACTIVITY              = 1
   AREA                  = 375
   HEIGHT                = 4
   Q-V-TYPE              = Mechanical
   Q-V-FACILITY          = None
   Q50-INF               = 15
   LIGHT-CASE            = UNKNOWN
   LIGHT-TYPE            = C-T12-F-H-L
   LIGHT-CONTROL         = MANUAL
   LIGHT-OCC-SENS-T      = NONE
   LIGHT-DISPLAY-EFF     = NO
```

```
LAMP-BALLAST-EFF-DL    = 50
LIGHT-PARAS-PW         = 0
LIGHT-TIME-SWITCH      = NONE
JNCT-ROOF-WALL-MC      = 0.25
JNCT-WALL-GRND-MC      = 1.2
JNCT-WALL-WALL-MC      = 0.25
JNCT-WALL-FLR-MC       = 0.07
JNCT-LINTEL-MC         = 1.5
JNCT-SILL-MC           = 0.41
JNCT-JAMB-MC           = 1.5
JNCT-ROOF-WALL         = 0.12
JNCT-WALL-GRND         = 0.32
JNCT-WALL-WALL         = 0.18
JNCT-WALL-FLR          = 0.14
JNCT-LINTEL            = 0.6
JNCT-SILL              = 0.08
JNCT-JAMB              = 0.1
JNCT-ACCR-DETAIL       = { 0, 0, 0, 0, 0, 0, 0 }
JNCT-ACCR-DETAIL-MC    = { -1, -1, -1, -1, -1, -1, -1 }
..
```

ISBEM-ID
This keyword specifies the interface's ID number for the ZONE object, and it is used when calculated values relating to that object need to be reported within the interface.

ACTIVITY
This is the identification number for the activity in the zone. See the Appendix I manual for the list of activity types and their identification numbers.

AREA
Specifies the area of the zone, in m^2. This keyword is required.

HEIGHT
Specifies the height of the zone, in m. The default value is 3.

LIGHT-TYPE
This is the type of lamp used in the zone, as one of the values listed below. The default is 'Fluorescent - halophosphate - low frequency ballast'.

NOT1
NOT2
REF1
REF2
TYP1
TYP2
LED
Tungsten or Halogen
Fluorescent - compact

T12 Fluorescent - halophosphate - low frequency ballast
T8 Fluorescent - halophosphate - low frequency ballast
T8 Fluorescent - halophosphate - high frequency ballast
T8 Fluorescent - triphosphor - high frequency ballast
Metal Halide
High Pressure Mercury
High Pressure Sodium
T5 Fluorescent - triphosphor-coated - high frequency ballast
Fluorescent (no details)

LIGHT-CASE
Specifies which of the cases listed below applies to the lighting in the zone. The default is 'UNKNOWN'.

Value	Description
UNKNOWN	Lighting parameters are not available
DESIGN	A full lighting design has been carried out, including verification that illuminances meet the standard levels.
CHOSEN	Lighting has been chosen but a full illuminance calculation has not been carried out.

LIGHT-ACT-WATT
This keyword is included in the input if the ZONE object's LIGHT-CASE keyword has the value DESIGN. This keyword must contain the actual lighting circuit power in the zone, in Watts.

LIGHT-LUX-DESIGN
This keyword is included in the input if the ZONE object's LIGHT-CASE keyword has the value DESIGN. This specifies the lighting design illuminance in the zone, in Lux.

LAMP-BALLAST-EFF
This keyword is included in the input if the ZONE object's LIGHT-CASE keyword has the value CHOSEN. This keyword specifies the average lamp and ballast efficacy for the lighting in the zone, in lumens per circuit-Watt. The default value is 50.

LIGHT-OUTPUT-RATIO
This keyword is included in the input if the ZONE object's LIGHT-CASE keyword has the value CHOSEN. This keyword specifies the light output ratio of the luminaires. The default value is 0.25.

LIGHT-CONTROL
Specifies the type of lighting control in the zone, as one of the values listed below. The default is 'MANUAL'.

Value	Description
MANUAL	Lighting is controlled by local manual switching
PHOTOELECTRIC	Lighting has photoelectric control
BOTH	Lighting is controlled by both local manual switching and

	photoelectric control
NONE	Lighting has no controls.

***LIGHT-OCC-SENS* (NO LONGER IN USE)**
Specifies whether there is occupancy sensing in the zone. If the value of this keyword is YES, there is occupancy sensing. If the value is NO, there is no occupancy sensing. The default is NO.

LIGHT-OCC-SENS-T
Specifies the type of occupancy sensing in the zone, as one of the values listed below. The default is 'NONE'.

Value	Description
MAN-ON-OFF+EXT	Lights manually switched on and off, with the addition of an automatic extinction signal.
AUTO-ON-DIMMED	Lights automatically switched on whenever people enter a room and dimmed to a low level when no movement has been detected for a set time (usually 5-15 minutes).
AUTO-ON-OFF	Lights automatically switched on whenever people enter a room and switched off when no movement has been detected for a set time (usually 5-15 minutes).
MAN-ON-DIMMED	Lights manually switched on and automatically dimmed to a low level when no movement has been detected for a set time (usually 5-15 minutes).
MAN-ON-AUTO-OFF	Lights manually switched on and automatically switched off when no movement has been detected for a set time (usually 5-15 minutes).
NONE	No occupancy sensing

LIGHT-PHOTO-TYPE
Specifies the type of photoelectric lighting control in the zone, as one of the values listed below. The default is PH-SWITCHING.

PH-SWITCHING
PH-DIMMING

LIGHT-PH-BACK
For zones with photoelectric lighting control, this keyword specifies whether there is photoelectric control in the back of the room using a separate sensor. YES signifies that there is, and NO means that there is not. The default is NO.

LIGHT-PARAS-PW
Specifies the lighting parasitic power due to photoelectric controls, in W/m^2.

LIGHT-PARAS-PW-OCC
Specifies the lighting parasitic power due to occupancy sensing, in W/m^2. The default value is 0.3.

LIGHT-DISPLAY-EFF
Specifies whether display lighting in the zone uses efficient lamps. It can be either YES or NO. The default is NO.

LAMP-BALLAST-EFF-DL
Specifies the average lamp and ballast efficacy for display lighting, in lumens per circuit-Watt. The default value is 50.

LIGHT-TIME-SWITCH
Specifies whether there is time control for display lighting in the zone. It can be YES or NO. The default is NO.

LIGHT-DL-AUT-ZONE
Specifies whether SBEM will automatically sub-divide the input zones according to the daylight zoning rules. If the value of this keyword is YES, the sub-division is done by SBEM. If the value is NO, SBEM will assume any sub-divisions have already been done by the user and will not sub-divide the zone any further. The default is YES.

LIGHT-DL-AREA-PERC
This keyword is included in the input if the ZONE object's LIGHT-DL-AUT-ZONE keyword has the value NO. It specifies the percentage of the area of the zone that is daylight-controlled (i.e., lighting responds to daylight). The default value is 100.

***LIGHT-DISP-HOFF* (NO LONGER IN USE)**
If there is time control of the display lighting, this keyword specifies the number of hours for which the display lighting is dimmed or for which a fraction of the display lighting is turned off. The default value is 0.

***LIGHT-DISP-F* (NO LONGER IN USE)**
If the value of LIGHT-TIME-SWITCH is SWITCHING, this keyword specifies the fraction of display lighting turned off. If the value of LIGHT-TIME-SWITCH is DIMMING, this keyword specifies the fraction of the display light total illuminance to which the display lighting is dimmed. The default value is 1.

LIGHT-GAIN-F
The fraction of the lighting energy consumption which contributes to zone heat gain. The default value is 1. If there are air-extracting luminaires in the zone, then this keyword is set to -6666 in the input file. (The -6666 is replaced by a default fraction within SBEM when calculating the internal heat gains due to lighting).

SHELL-ASSUMED
Specifies whether the zone is a shell area (in a building which is a shell and core development) whose building services systems have been assumed by the user (i.e., not actually fitted). It can be YES or NO. The default is NO.

Q-V-TYPE
The specific air flow, due to mechanical ventilation and natural ventilation, of direct-entering fresh outside air to be heated depends on the way the fresh outside air is supplied to the considered zones. Fresh outside air may enter into the zone by natural ventilation or mechanical ventilation. This keyword specifies which of these should be considered in the calculation, as one of the values listed below. The default is 'Mechanical'.

Value	Description
Natural Supply	Only Natural ventilation

Mechanical	Only Mechanical ventilation

Q50-INF
Specifies the zone's air permeability at 50 Pa, in m^3/m^2.hour. This is the average volume of air, in m^3 per hour, which passes through unit area of the zone's envelope, in m^2, when subject to an internal to external pressure difference of 50 Pascals. The envelope area of the zone is defined as the total area of the floor, walls, and roof separating the interior volume from the outside environment. The default value is 10.

VENT-SFP
The specific fan power for zonal mechanical supply, in W/(l/s). The default value is 3.

Modified

SFP-TU
The specific fan power for system terminal units in the zone, in W/(l/s). It is included in the input if, for the HVAC system serving the zone, the HVAC-SYSTEM object's TYPE keyword has been set to 'Fan coil systems' or 'Indoor packaged cabinet (VAV)'. The default value is 0.8.

***F-CTRL-VENT-TYPE* (NO LONGER IN USE)**
Specifies whether the ZONE ventilation system has flow reduction or recirculation, as one of the values listed below. The default is 'no recirculation'.

no recirculation
less than 20%
20% to 40%
40% to 60%
more than 60%

F-CTRL-VENT-DEM
Specifies demand-controlled ventilation in the zone, as one of the values listed below. The default is 'No demand-controlled ventilation'.

Value	Applicable when ventilation type defined at zone level is
No demand-controlled ventilation	Natural or mechanical
Demand control dependent on number of occupants	Mechanical
Demand control dependent on gas sensors	Mechanical
Enhanced ventilation	Natural

FLOW-REGU-TYPE
Specifies the air flow regulation type for demand-controlled ventilation in the zone, as one of the values listed below. The default is 'Damper control'.

Value	Applicable when ventilation type defined at zone level is
Damper control	Mechanical
Speed control	Mechanical

DHW-GENERATOR
This is the U-name previously defined in a DHW-GENERATOR instruction to specify the hot water system which serves the zone. This is a required keyword.

HEAT-REC-SYSTEM
Specifies the heat recovery system for the zone ventilation, as one of the values listed below. The default is 'No heat recovery'.

No heat recovery
Plate heat exchanger (Recuperator)
Heat pipes
Thermal wheel
Run around coil

HEAT-REC-SEFF
Specifies the seasonal efficiency of the heat recovery system. The default values for the different types of heat recovery system options in the ZONE object's HEAT-REC-SYSTEM keyword are listed below.

Heat Recovery System	Default Efficiency (based on CIBSE suggestions)
No heat recovery	0
Plate heat exchanger (Recuperator)	0.65
Heat pipes	0.6
Thermal wheel	0.65
Run around coil	0.5

HEAT-REC-VAR-EFF
This keyword is included in the input if the ZONE object's HEAT-REC-SYSTEM keyword has a value other than 'No heat recovery'. It indicates whether the seasonal efficiency of the heat recovery system is variable. The value can be YES or NO. The default is NO.

DHW-PIPE-LEN
Specifies the length of the hot water system pipework in the zone, in m. The default value is 0.

VENT-MECH-EXH
Specifies whether there is mechanical exhaust at zone level. It can be either YES or NO. The default is NO.

VENT-SFP-EXH
This keyword in included in the input if the ZONE object's VENT-MECH-EXH keyword is set to YES. It specifies the specific the fan power for zonal mechanical exhaust, in W/(l/s). The default value is 3.

Q-V-M-EXH
This keyword in included in the input if the ZONE object's VENT-MECH-EXH keyword is set to YES. It specifies the air flow rate for zonal mechanical exhaust, in $l/s.m^2$. The default value is 5.

VENT-MECH-EXH-NCM

This keyword in included in the input if the ZONE object's VENT-MECH-EXH keyword is set to YES. It specifies the type of exhaust system in the zone as one of the values listed below. The default is 'Extract system serving a single room'.

Extract system serving multiple spaces
Extract system serving a single room

HEPA-FILTER
Specifies whether there are HEPA filters in the zone. It can be either YES or NO. The default is NO.

VENT-ZONE-HVAC
Specifies whether the ventilation type is determined at zone or HVAC. It can be one of the values listed below. The default is 'HVAC'.

Value	Description
ZONE	Ventilation is determined at zone level
HVAC	Ventilation is determined at HVAC level

NIGHT-COOLING
Specifies whether a night cooling strategy operates in the zone. It can be either YES or NO. The default is NO.

MAX-FLOW-NC
This keyword is included in the input if the ZONE object's NIGHT-COOLING keyword is set to YES. It specifies the maximum air flow rate in the zone, in $l/s.m^2$ of floor area, during the operation of night cooling. The default value is 1.

MAX-HOURS-NC
This keyword is included in the input if the ZONE object's NIGHT-COOLING keyword is set to YES. It specifies the maximum number of hours per month during which night cooling is operating in the zone. The default value is 200.

SFP-NC
This keyword is included in the input if the ZONE object's NIGHT-COOLING keyword is set to YES. It specifies the specific fan power for night cooling operation, in W/(l/s). The default value is 2.5.

TSC-SYSTEM
Specifies whether a Transpired Solar Collector (TSC) system supplies pre-heated air to this zone. It can be either YES or NO. The default is NO.

TSC
This keyword is included in the input if the ZONE object's TSC-SYSTEM keyword has been set to YES. It specifies a list of U-name(s) previously defined in a TSC instruction(s) to name the TSC system(s) supplying pre-heated air to the zone, in the form shown below (enclosed by braces and separated by commas).

{"tsc-sys$_1$", "tsc-sys$_2$", .., .., .., .., "tsc-sys$_n$" } where tsc-sys$_i$ is the U-name of a TSC object previously defined in the input file.

TSC-SUPPLY
This keyword is included in the input if the ZONE object's TSC-SYSTEM keyword has been set to YES. It specifies a list of percentage value(s) of the total air pre-heated by the TSC system(s) that is supplied to the zone. The list of percentage values in this keyword must correspond, in number and in order, to the TSC systems listed in the ZONE object's TSC keyword.

{ tsc-supply$_1$, tsc-supply$_2$, .., .., .., .., tsc-supply$_n$ } where tsc-supply$_i$ is a value for the supplied pre-heated air, in percentage form.

LIGHT-DF
Daylight factor for the zone. If this keyword is not included in the input file, its value is calculated by SBEM.

DESTRAT-FAN
Specifies whether there are destratification fans in the zone. It can have the value YES or NO. The default is NO.

MULTIPLIER
This keyword specifies how many zones with this exact description exist under the current HVAC-SYSTEM (i.e., served by it). E.g., if the value of the multiplier is 3, there are 3 zones, each with identical keyword values, being served by the HVAC-SYSTEM considered. This enables repeating zones to be entered quickly and easily. The default value is 1.

N-CORNERS **(NO LONGER IN USE)**
Specifies the net number of corners in contact with the exterior in the zone (i.e., the number of convex corners minus the number of concave corners as perceived from the exterior). The default value is 1.

JNCT-ROOF-WALL-MC
Specifies psi, the linear thermal transmittance of thermal bridges, in W/(mK), for roof to wall junctions involving metal cladding. The default value is 0.6.

JNCT-WALL-GRND-MC
Specifies psi, the linear thermal transmittance of thermal bridges, in W/(mK), for wall to ground floor junctions involving metal cladding. The default value is 1.15.

JNCT-WALL-WALL-MC
Specifies psi, the linear thermal transmittance of thermal bridges, in W/(mK), for wall to wall (i.e., corners) junctions involving metal cladding. The default value is 0.25.

JNCT-WALL-FLR-MC
Specifies psi, the linear thermal transmittance of thermal bridges, in W/(mK), for wall to floor (not ground floor) junctions involving metal cladding. The default value is 0.07.

JNCT-LINTEL-MC
Specifies psi, the linear thermal transmittance of thermal bridges, in W/(mK), for lintels (above windows or doors) involving metal cladding. The default value is 1.27.

JNCT-SILL-MC
Specifies psi, the linear thermal transmittance of thermal bridges, in W/(mK), for sill (below windows) involving metal cladding. The default value is 1.27.

JNCT-JAMB-MC
Specifies psi, the linear thermal transmittance of thermal bridges, in W/(mK), for jambs (at windows or doors) involving metal cladding. The default value is 1.27.

***JNCT-OTHER-MC* (NO LONGER IN USE)**
Specifies psi, the linear thermal transmittance of thermal bridges, in W/(mK), for other junctions involving metal cladding. The default value is 1.27.

JNCT-ACCR-DETAIL-MC
Specifies a list of values corresponding in number and in order to the thermal bridge junctions in the zone that involve metal-cladding, in the form shown below (enclosed by braces and separated by commas).
{ $junc_1$, $junc_2$, $junc_3$, $junc_4$, $junc_5$, $junc_6$, $junc_7$} where $junc_i$ is one of the values listed below

Value	Description
0	Default psi value to be used for junction (i.e., value not input by user).
-1	Psi value input for junction by user has not been assessed using a quality-assured accredited construction details approach.
1	Psi value input for junction by user has been assessed using a quality-assured accredited construction details approach.

and the order of the values in the list must correspond to the junctions as shown below.

Order of values in list	Applicable to psi value for junction (involving metal-cladding)
$junc_1$	JNCT-ROOF-WALL-MC
$junc_2$	JNCT-WALL-GRND-MC
$junc_3$	JNCT-WALL-WALL-MC
$junc_4$	JNCT-WALL-FLR-MC
$junc_5$	JNCT-LINTEL-MC
$junc_6$	JNCT-SILL-MC
$junc_7$	JNCT-JAMB-MC

JNCT-ROOF-WALL
Specifies psi, the linear thermal transmittance of thermal bridges, in W/(mK), for roof to wall junctions (not involving metal cladding). The default value is 0.12.

JNCT-WALL-GRND
Specifies psi, the linear thermal transmittance of thermal bridges, in W/(mK), for wall to ground floor junctions (not involving metal cladding). The default value is 0.28.

JNCT-WALL-WALL
Specifies psi, the linear thermal transmittance of thermal bridges, in W/(mK), for wall to wall (i.e., corners) junctions (not involving metal cladding). The default value is 0.09.

JNCT-WALL-FLR
Specifies psi, the linear thermal transmittance of thermal bridges, in W/(mK), for wall to floor (not ground floor) junctions (not involving metal cladding). The default value is 0.18.

JNCT-LINTEL
Specifies psi, the linear thermal transmittance of thermal bridges, in W/(mK), for lintels above windows or doors (not involving metal cladding). The default value is 0.53.

JNCT-SILL
Specifies psi, the linear thermal transmittance of thermal bridges, in W/(mK), for sill below windows (not involving metal cladding). The default value is 0.21.

JNCT-JAMB
Specifies psi, the linear thermal transmittance of thermal bridges, in W/(mK), for jambs at windows or doors (not involving metal cladding). The default value is 0.2.

JNCT-OTHER **(NO LONGER IN USE)**
Specifies psi, the linear thermal transmittance of thermal bridges, in W/(mK), for other junctions (not involving metal cladding). The default value is 0.2.

JNCT-ACCR-DETAIL
Specifies a list of values corresponding in number and in order to the thermal bridge junctions in the zone that do not involve metal-cladding, in the form shown below (enclosed by braces and separated by commas).

{ $junc_1$, $junc_2$, $junc_3$, $junc_4$, $junc_5$, $junc_6$, $junc_7$} where $junc_i$ is one of the values listed below

Value	Description
0	Default psi value used for junction (i.e., value not input by user).
-1	Psi value input for junction by user has not been assessed using a quality-assured accredited construction details approach.
1	Psi value input for junction by user has been assessed using a quality-assured accredited construction details approach.

and the order of the values in the list must correspond to the junctions as shown below.

Order of values in list	Applicable to Psi value for junction (not involving metal-cladding)
$junc_1$	JNCT-ROOF-WALL
$junc_2$	JNCT-WALL-GRND
$junc_3$	JNCT-WALL-WALL
$junc_4$	JNCT-WALL-FLR
$junc_5$	JNCT-LINTEL
$junc_6$	JNCT-SILL
$junc_7$	JNCT-JAMB

c++ code

```
//: space.cpp

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void SPACE::TEST() {
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test SPACE" << endl;
//  cout << "Fin de Test SPACE" << endl;
  BEEMObject::TEST();
  for(int i=0;i<vW.size();i++)
    vW[i].TEST();
}

void SPACE::Reset() {
  if(func report) cout << endl << "  " << "SPACE::Reset";
  dF.clear();
//  vF.clear();
  dW.clear();
//  vW.clear();
}

void SPACE::DtoV() {
  if(func report) cout << endl << "  " << "SPACE::DtoV";
//  for(int i=0;i<dW.size();i++)  dW[i].DtoV();
//  vW.reserve(dW.size());
//  vW.assign(dW.begin(),dW.end());
//  dW.clear();
//
//  vF.reserve(dF.size());
//  vF.assign(dF.begin(),dF.end());
//  dF.clear();
}

void SPACE::fH TR TM NOT(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func_report) cout << endl << "  " << "SPACE::fH_TR_TM_NOT";
//for notional building only: heat transfer coefficient = 0
//NOTE: notional thermal bridge heat loss is instead accounted for by
use of 'alpha' value multiplying u-values in CONSTRUCTION::Notional()
 {
  ostringstream res; res << 0.0;
  P.ModKey(Id,"H-TR-TM",res.str());
 }
}


void  SPACE::fH TR TM NOT S6(PROJECTDefaults&  D,  PROJECTTables&  T,
BOC& P) {
  if(func report) cout << endl << "  " << "SPACE::fH TR TM NOT S6";
//calculates  heat  transfer  coefficient  for  thermal  bridges  by
multiplying Length of juction by psi value for that type of junction
//psi value units: W/m/K
//calculates length of each junction type
//writes keywords "JNCT-L-..."  for junction lengths, 1 keyword per
junction type

  double mH_TR_TM=0;
  double mJNCT_L_ROOF_WALL=0;
  double mJNCT_L_WALL_GRND=0;
  double mJNCT_L_WALL_WALL=0;
  double mJNCT_L_WALL_FLR=0;
  double mJNCT_L_LINTEL=0;
```

```
  double mJNCT L SILL=0;
  double mJNCT L JAMB=0;
  double mJNCT L OTHER=0;
  double mRatio=1;
  double mWin_Area=1;
  double mDoor Area=1;
  double mRooflight Area=0.0;  // area of a single rooflight in the
notional building
  double mRoof Num=1.0;        // number of individual rooflights in
notional building
  double  mPitch=0.0;                 // pitch angle of roof where
rooflights are placed
  double mRoof Area=0.0;       // area of roof where rooflights are
placed

// Calculation of Lengths...

  bool flag Roof wall=0;
  bool flag Wall ground floor=0;
  bool flag_Wall_floor=0;
  double mAREA EXT WALL=0;
  double  mHEIGHT=P.Vnum(Id,"HEIGHT");  if(mHEIGHT==0)  mHEIGHT=3;  //-
D.Value("typical thicknes of walls")

  for (int i=0;i<vW.size();i++) {
    if(P.Vcodid(vW[i].Id,"TYPE")==1  ||  P.Vcodid(vW[i].Id,"TYPE")==5)
{ //Only if Envelope is exterior or underground
      switch (P.Vcodid(vW[i].Id,"TYPE-ENV")) {
        case 1: {

mAREA EXT WALL+=P.Vnum(vW[i].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER")
;  // multiplier of wall incorporated
                  for(int j=0;j<vW[i].vW.size();j++) {
                    if (P.Vcodid(vW[i].vW[j].Id,"DISP-WIND")!=1) { //
i.e., not a display window
                      mRatio=D.Value("dWindow ratio");

mWin Area=P.Vnum(vW[i].vW[j].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER")
*P.Vnum(vW[i].vW[j].Id,"MULTIPLIER");   // multipliers of walls and
windows
                      mJNCT L LINTEL+=sqrt(mWin Area/mRatio);
                      mJNCT_L_SILL+=sqrt(mWin_Area/mRatio);
                      mJNCT L JAMB+=2*sqrt(mWin Area*mRatio);
                    } // end if
                  } // end for
        }
        break;
        case 2: {
                flag Roof wall=1;
                //Rooflights
                mPitch=P.Vnum(vW[i].Id,"PITCH");       // roof pitch
angle, in degrees
                mRoof Area=P.Vnum(vW[i].Id,"AREA"); // roof area, in
m2
                // The number of rooflights in the notional building
is calculated using the following equation - AN (9/11/2010)

mRoof Num=round(mRoof Area/pow((2*mHEIGHT*tan(40*3.14159265/180.0)/co
s(mPitch*3.14159265/180.0)),2));

                for(int j=0;j<vW[i].vW.size();j++) {
                  mRatio=D.Value("dWindow ratio");   if   (mRatio==0)
mRatio=1; // Use default of 0.7 as aspect ratio of rooflights

mWin Area=P.Vnum(vW[i].vW[j].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER")
*P.Vnum(vW[i].vW[j].Id,"MULTIPLIER");   // multipliers of walls and
windows
                  // mJNCT L LINTEL+=sqrt(mWin Area/mRatio);
                  // mJNCT_L_SILL+=sqrt(mWin_Area/mRatio);
```

```
                  // mJNCT L JAMB+=2*sqrt(mWin Area*mRatio);
                  // Modification below (in consulation with Aecom on
15/09/2010) so that for all the sides of
                  //  rooflights  only  jamb  junctions  are  assumed,
i.e., no lintel or sill

//mJNCT L JAMB+=2*sqrt(mWin Area*mRatio)+2*sqrt(mWin Area/mRatio);
                  if(mRoof Num>0.0) {
                    mRooflight Area=mWin Area/mRoof Num;

mJNCT_L_JAMB+=mRoof_Num*(2*sqrt(mRooflight_Area*mRatio)+2*sqrt(mRoofl
ight Area/mRatio));
                  }
                  else

mJNCT L JAMB+=2*sqrt(mWin Area*mRatio)+2*sqrt(mWin Area/mRatio);
                } // end for
        }
        break;
        case 3: {
                flag Wall ground floor=1;
        }
        break;
        default: {

        }
        break;
      }
      //Doors
      for(int j=0;j<vW[i].vD.size();j++) {
         mRatio=D.Value("dDoor ratio"); if (mRatio==0) mRatio=1;

mDoor Area=P.Vnum(vW[i].vD[j].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER"
)*P.Vnum(vW[i].vD[j].Id,"MULTIPLIER");   // multipliers of walls and
doors
         mJNCT L LINTEL+=sqrt(mRatio*mDoor Area);
         mJNCT L JAMB+=2*sqrt(mDoor Area/mRatio);
      }
    }
    else
      if(P.Vcodid(vW[i].Id,"TYPE-ENV")==3) flag Wall floor=1;
  }

  if(flag Roof wall) {

mJNCT L ROOF WALL=mAREA EXT WALL/mHEIGHT/(flag Roof wall+flag Wall fl
oor);
  }

  if(flag Wall ground floor) {
    mJNCT_L_WALL_GRND=mAREA_EXT_WALL/mHEIGHT;
  }
  else
    if(flag_Wall_floor) {

mJNCT L WALL FLR=mAREA EXT WALL/mHEIGHT/(flag Roof wall+flag Wall flo
or);
    }

  mJNCT_L_WALL_WALL=P.Vnum(Id,"N-CORNERS")*mHEIGHT;

 {
  ostringstream res; res << 0.0;
  P.ModKey(Id,"JNCT-OTHER",res.str());
 }
  mH_TR_TM=mJNCT_L_ROOF_WALL*T.Value("Not_Psi",1,4) +
           mJNCT L WALL GRND*T.Value("Not Psi",2,4) +
           mJNCT L WALL WALL*T.Value("Not Psi",3,4) +
           mJNCT_L_WALL_FLR*T.Value("Not_Psi",4,4) +
```

```
          mJNCT_L_LINTEL*T.Value("Not Psi",5,4) +
          mJNCT_L_SILL*T.Value("Not Psi",6,4) +
          mJNCT_L_JAMB*T.Value("Not Psi",7,4) +
          mJNCT_L_OTHER*T.Value("Not Psi",8,4);

 {
  ostringstream res; res << mH_TR_TM;
  P.ModKey(Id,"H-TR-TM",res.str());
 }
 {
  ostringstream res; res << 0.0;
  P.ModKey(Id,"JNCT-L-ROOF-WALL-MC",res.str());
 }
 {
  ostringstream res; res << 0.0;
  P.ModKey(Id,"JNCT-L-WALL-GRND-MC",res.str());
 }
 {
  ostringstream res; res << 0.0;
  P.ModKey(Id,"JNCT-L-WALL-WALL-MC",res.str());
 }
 {
  ostringstream res; res << 0.0;
  P.ModKey(Id,"JNCT-L-WALL-FLR-MC",res.str());
 }
 {
  ostringstream res; res << 0.0;
  P.ModKey(Id,"JNCT-L-LINTEL-MC",res.str());
 }
 {
  ostringstream res; res << 0.0;
  P.ModKey(Id,"JNCT-L-SILL-MC",res.str());
 }
 {
  ostringstream res; res << 0.0;
  P.ModKey(Id,"JNCT-L-JAMB-MC",res.str());
 }
 {
  ostringstream res; res << 0.0;
  P.ModKey(Id,"JNCT-L-OTHER-MC",res.str());
 }
 {
  ostringstream res; res << mJNCT_L_ROOF_WALL;
  P.ModKey(Id,"JNCT-L-ROOF-WALL",res.str());
 }
 {
  ostringstream res; res << mJNCT_L_WALL_GRND;
  P.ModKey(Id,"JNCT-L-WALL-GRND",res.str());
 }
 {
  ostringstream res; res << mJNCT_L_WALL_WALL;
  P.ModKey(Id,"JNCT-L-WALL-WALL",res.str());
 }
 {
  ostringstream res; res << mJNCT_L_WALL_FLR;
  P.ModKey(Id,"JNCT-L-WALL-FLR",res.str());
 }
 {
  ostringstream res; res << mJNCT_L_LINTEL;
  P.ModKey(Id,"JNCT-L-LINTEL",res.str());
 }
 {
  ostringstream res; res << mJNCT_L_SILL;
  P.ModKey(Id,"JNCT-L-SILL",res.str());
 }
 {
  ostringstream res; res << mJNCT_L_JAMB;
  P.ModKey(Id,"JNCT-L-JAMB",res.str());
 }
```

```
 {
  ostringstream res; res << mJNCT L OTHER;
  P.ModKey(Id,"JNCT-L-OTHER",res.str());
 }

}

void SPACE::fH_TR_TM(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "SPACE::fH TR TM";
//calculates heat transfer coefficient for thermal bridges by
multiplying Length of juction by psi value for that type of junction
//psi value units: W/m/K
//calculates length of each junction type
//writes keywords "JNCT-L-..."  for junction lengths, 1 keyword per
junction type

  double mH_TR_TM=0;
  double mJNCT L ROOF WALL MC=0;
  double mJNCT L WALL GRND MC=0;
  double mJNCT_L_WALL_WALL_MC=0;
  double mJNCT L WALL FLR MC=0;
  double mJNCT L LINTEL MC=0;
  double mJNCT L SILL MC=0;
  double mJNCT L JAMB MC=0;
  double mJNCT L OTHER MC=0;
  double mJNCT L ROOF WALL=0;
  double mJNCT L WALL GRND=0;
  double mJNCT L WALL WALL=0;
  double mJNCT L WALL FLR=0;
  double mJNCT L LINTEL=0;
  double mJNCT L SILL=0;
  double mJNCT L JAMB=0;
  double mJNCT L OTHER=0;
  double mRatio=1;
  double mWin Area=1;
  double mDoor Area=1;
  double mRooflight Area=0.0;  // area of a single rooflight in the
notional building
  double mRoof_Num=1.0;        // number of individual rooflights in
notional building
  double  mPitch=0.0;                 // pitch angle of roof where
rooflights are placed
  double mRoof Area=0.0;       // area of roof where rooflights are
placed

// Calculation of Lengths...

  bool flag_Roof_wall=0;
  bool flag Wall ground floor=0;
  bool flag Wall floor=0;
  bool flag_RMC=0;
  bool flag WMC=0;
  bool flag MC=0;
  double mAREA_EXT_WALL_MC=0;
  double mAREA EXT WALL=0;
  double  mHEIGHT=P.Vnum(Id,"HEIGHT"); if(mHEIGHT==0) mHEIGHT=3; //-
D.Value("typical thicknes of walls")

  //To detect if Roof is Metal Cladding
  flag_RMC=0;
  for (int i=0;i<vW.size();i++)
    if(P.Vcodid(vW[i].Id,"TYPE-ENV")==2)
      if(P.Vcod(P.Vrefid(vW[i].Id,"CONSTRUCTION"),"METAL-
CLADDING")=="YES")
        flag_RMC=1;
//        { flag_RMC=1; flag_MC=1; }

  for (int i=0;i<vW.size();i++) {
    flag_WMC=0;
```

```
    if(P.Vcodid(vW[i].Id,"TYPE")==1  ||  P.Vcodid(vW[i].Id,"TYPE")==5)
{ //Only if Envelope is exterior or underground
      switch (P.Vcodid(vW[i].Id,"TYPE-ENV")) {
        case 1: {
                if(P.Vcod(P.Vrefid(vW[i].Id,"CONSTRUCTION"),"METAL-
CLADDING")=="YES" || flag RMC) {

mAREA_EXT_WALL_MC+=P.Vnum(vW[i].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIE
R");   // incoporating multipler of walls
                }
                else

mAREA EXT WALL+=P.Vnum(vW[i].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER")
;   // incoporating multipler of walls
                if(P.Vcod(P.Vrefid(vW[i].Id,"CONSTRUCTION"),"METAL-
CLADDING")=="YES") {
                  flag_WMC=1; flag_MC=1;
                }
                  //Windows
                for(int j=0;j<vW[i].vW.size();j++) {
//                    if (P.Vcod(gId,"ACT-NOT")=="ACT") {
                      //if   (P.Vcodid(vW[i].vW[j].Id,"DISP-WIND")!=1)
{ // i.e., not a display window
                      // Since display windows in the actual building
now take the U-value input by the user
                      // instead of being set to 5.7 W/m2), thermal
bridging losses can be added.
                  mRatio=P.Vnum(vW[i].vW[j].Id,"ASPECT-RATIO");
                  if (mRatio<=0)   // If not input or input as <=0
                    mRatio=D.Value("dWindow ratio");

mWin Area=P.Vnum(vW[i].vW[j].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER")
*P.Vnum(vW[i].vW[j].Id,"MULTIPLIER");  // incoporating multipliers of
walls and windows
                  if (flag WMC) {
                    mJNCT L LINTEL MC+=sqrt(mWin Area/mRatio);
                    mJNCT L SILL MC+=sqrt(mWin Area/mRatio);
                    mJNCT L JAMB MC+=2*sqrt(mWin Area*mRatio);
                  }
                  else {
                    mJNCT L LINTEL+=sqrt(mWin Area/mRatio);
                    mJNCT_L_SILL+=sqrt(mWin_Area/mRatio);
                    mJNCT L JAMB+=2*sqrt(mWin Area*mRatio);
                  }
                  //} // end if
//                    } // end if
//                          else { // i.e., notional, but only 2010
aggregate notional calls this function
//                            // Since display windows in the notional
building are set to have a high U-value,
//                           // no thermal bridging losses are added on
top.
//                      //double mLength=P.Vnum(vW[i].Id,"LENGTH");
//                      //if(mLength==0) {
//                                                                    //
mLength=sqrt(P.Vnum(vW[i].Id,"AREA")/D.Value("dWindow ratio"));
//                      //}
//                      if (P.Vnum(vW[i].Id,"LENGTH")!=0.0)
//                         mRatio=1/P.Vnum(vW[i].Id,"LENGTH"); // Use
the inverse of wall perimeter as aspect ratio of window
//                      else
//                        mRatio=D.Value("dWindow ratio");
//
mWin Area=P.Vnum(vW[i].vW[j].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER")
*P.Vnum(vW[i].vW[j].Id,"MULTIPLIER");  // incoporating multipliers of
walls and windows
//                            // Since display windows in the actual
building now take the U-value input by the user
//                       // instead of being set to 5.7 W/m2), thermal
```

```
bridging losses can be added.
////                                  if (P.Vcodid(vW[i].vW[j].Id,"DISP-
WIND")!=1) { // i.e., not a display window
//                            if (flag WMC) {
//
mJNCT L LINTEL MC+=sqrt(mWin Area/mRatio);
//                              mJNCT L SILL MC+=sqrt(mWin Area/mRatio);
//
mJNCT L JAMB MC+=2*sqrt(mWin Area*mRatio);
//                            }
//                            else {
//                              mJNCT L LINTEL+=sqrt(mWin Area/mRatio);
//                              mJNCT L SILL+=sqrt(mWin Area/mRatio);
//                              mJNCT_L_JAMB+=2*sqrt(mWin_Area*mRatio);
//                            }
////                          } // end if
//
//                       } // end else
                  } // end for
          }
          break;
          case 2: {
                  flag Roof wall=1;
                   //Rooflights
                  mPitch=P.Vnum(vW[i].Id,"PITCH");       //  roof  pitch
angle, in degrees
                  mRoof Area=P.Vnum(vW[i].Id,"AREA"); //  roof  area,  in
m2
                  // The number of rooflights in the notional building
is calculated using the following equation - AN (9/11/2010)

mRoof Num=round(mRoof Area/pow((2*mHEIGHT*tan(40*3.14159265/180.0)/co
s(mPitch*3.14159265/180.0)),2));

                  for(int j=0;j<vW[i].vW.size();j++) {
                    mRatio=P.Vnum(vW[i].vW[j].Id,"ASPECT-RATIO");
                    if (mRatio<=0)   // If not input or input as <=0
                      mRatio=D.Value("dWindow ratio");

mWin Area=P.Vnum(vW[i].vW[j].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER")
*P.Vnum(vW[i].vW[j].Id,"MULTIPLIER");  // incoporating multipliers of
walls and windows
                    if (flag RMC) { // involves metal-cladding
                      if(       (P.Vcod(gId,"ACT-NOT")=="NOT")       &&
(mRoof_Num>0.0) ) {
                         mRooflight Area=mWin Area/mRoof Num;

mJNCT_L_JAMB_MC+=mRoof_Num*(2*sqrt(mRooflight_Area*mRatio)+2*sqrt(mRo
oflight Area/mRatio));
                      } // end if
                      else { // i.e., this is the actual building
                        // mJNCT L LINTEL MC+=sqrt(mWin Area/mRatio);
                        // mJNCT L SILL MC+=sqrt(mWin Area/mRatio);
                        // mJNCT_L_JAMB_MC+=2*sqrt(mWin_Area*mRatio);
                        //  Modification  below  (in  consulation  with
Aecom on 15/09/2010) so that for all the sides of
                        // rooflights only jamb junctions are assumed,
i.e., no lintel or sill

mJNCT_L_JAMB_MC+=2*sqrt(mWin_Area*mRatio)+2*sqrt(mWin_Area/mRatio);
                      } // end else
                    }
                    else {
                      if(       (P.Vcod(gId,"ACT-NOT")=="NOT")       &&
(mRoof Num>0.0) ) {
                         mRooflight_Area=mWin_Area/mRoof_Num;

mJNCT L JAMB+=mRoof Num*(2*sqrt(mRooflight Area*mRatio)+2*sqrt(mRoofl
ight_Area/mRatio));
```

```
                    } // end if
                    else { // i.e., this is the actual building
                      // mJNCT L LINTEL+=sqrt(mWin Area/mRatio);
                      // mJNCT L SILL+=sqrt(mWin Area/mRatio);
                      // mJNCT_L_JAMB+=2*sqrt(mWin_Area*mRatio);
                      //  Modification  below  (in  consulation  with
Aecom on 15/09/2010) so that for all the sides of
                      // rooflights only jamb junctions are assumed,
i.e., no lintel or sill

mJNCT_L_JAMB+=2*sqrt(mWin_Area*mRatio)+2*sqrt(mWin_Area/mRatio);
                    } // end else
                  }
                }
        }
        break;
        case 3: {
                flag Wall ground floor=1;
        }
        break;
        default: {

        }
        break;
      }
      //Doors
      for(int j=0;j<vW[i].vD.size();j++) {
         mRatio=D.Value("dDoor ratio"); if (mRatio==0) mRatio=1;

mDoor Area=P.Vnum(vW[i].vD[j].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER"
)*P.Vnum(vW[i].vD[j].Id,"MULTIPLIER");
         if (flag WMC) {
           mJNCT L LINTEL MC+=sqrt(mRatio*mDoor Area);
           mJNCT L JAMB MC+=2*sqrt(mDoor Area/mRatio);
         }
         else {
           mJNCT L LINTEL+=sqrt(mRatio*mDoor Area);
           mJNCT L JAMB+=2*sqrt(mDoor Area/mRatio);
         }
      }
    }
    else
      if(P.Vcodid(vW[i].Id,"TYPE-ENV")==3) flag Wall floor=1;
  }

  if(flag Roof wall) {

mJNCT_L_ROOF_WALL_MC=mAREA_EXT_WALL_MC/mHEIGHT/(flag_Roof_wall+flag_W
all floor);

mJNCT_L_ROOF_WALL=mAREA_EXT_WALL/mHEIGHT/(flag_Roof_wall+flag_Wall_fl
oor);
  }

  if(flag Wall ground floor) {
    if(flag MC) { // check if walls have metal-cladding
      mJNCT_L_WALL_GRND_MC=mAREA_EXT_WALL_MC/mHEIGHT;
      mJNCT L WALL GRND=mAREA EXT WALL/mHEIGHT;
    }
    else // i.e., walls do not have metal cladding so the areas of
walls in mAREA EXT WALL MC are due to a roof with
         // metal-cladding which should not affect the wall to floor
junctions; hence the the summation in the line below

mJNCT L WALL GRND=(mAREA EXT WALL/mHEIGHT)+(mAREA EXT WALL MC/mHEIGHT
);
  }
  else
    if(flag_Wall_floor) {
```

```
      if(flag MC) { // check if walls have metal-cladding

mJNCT L WALL FLR MC=mAREA EXT WALL MC/mHEIGHT/(flag Roof wall+flag Wa
ll floor);

mJNCT L WALL FLR=mAREA EXT WALL/mHEIGHT/(flag Roof wall+flag Wall flo
or);
      }
      else // i.e., walls do not have metal cladding so the areas of
walls in mAREA EXT WALL MC are due to a roof with
           // metal-cladding which should not affect the wall to
floor junctions; hence the the summation in the line below

mJNCT_L_WALL_FLR=(mAREA_EXT_WALL/mHEIGHT/(flag_Roof_wall+flag_Wall_fl
oor))+(mAREA EXT WALL MC/mHEIGHT/(flag Roof wall+flag Wall floor));
    }

  if(flag MC)
    mJNCT L WALL WALL MC=P.Vnum(Id,"N-CORNERS")*mHEIGHT;
  else
    mJNCT L WALL WALL=P.Vnum(Id,"N-CORNERS")*mHEIGHT;

 {
  ostringstream res; res << 0.0;
  P.ModKey(Id,"JNCT-OTHER-MC",res.str());
 }
 {
  ostringstream res; res << 0.0;
  P.ModKey(Id,"JNCT-OTHER",res.str());
 }

 // !!! In the lines below, the psi values are adjusted (degraded) if
they are not accredited. If they are user-input
 // !!! values, they are degraded by the maximum of +0.02 W/mK and
+25%. If they are default values from BRE IP 01/06,
 // !!! they are degraded by the maximum of +0.04 W/mK and +50%.

  // Non-metal cladding junctions
  deque<double> mJNCT_ACCR_DETAILS=P.Vlnum(Id,"JNCT-ACCR-DETAIL");
  if(mJNCT ACCR DETAILS[0]==-5555) {
    mJNCT ACCR DETAILS[0]=0.0;
    mJNCT ACCR DETAILS.push back(0.0);
    mJNCT ACCR DETAILS.push back(0.0);
    mJNCT ACCR DETAILS.push back(0.0);
    mJNCT ACCR DETAILS.push back(0.0);
    mJNCT ACCR DETAILS.push back(0.0);
    mJNCT ACCR DETAILS.push back(0.0);
    mJNCT ACCR DETAILS.push back(0.0);
  }
  double mJNCT CORR[8]={1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0};
  double mJNCT_ROOF_WALL=P.Vnum(Id,"JNCT-ROOF-WALL");
  double mJNCT WALL GRND=P.Vnum(Id,"JNCT-WALL-GRND");
  double mJNCT WALL WALL=P.Vnum(Id,"JNCT-WALL-WALL");
  double mJNCT_WALL_FLR=P.Vnum(Id,"JNCT-WALL-FLR");
  double mJNCT LINTEL=P.Vnum(Id,"JNCT-LINTEL");
  double mJNCT SILL=P.Vnum(Id,"JNCT-SILL");
  double mJNCT_JAMB=P.Vnum(Id,"JNCT-JAMB");
  double mJNCT OTHER=P.Vnum(Id,"JNCT-OTHER");

  // Metal cladding junctions
  deque<double>    mJNCT ACCR DETAILS MC=P.Vlnum(Id,"JNCT-ACCR-DETAIL-
MC");
  if(mJNCT_ACCR_DETAILS_MC[0]==-5555) { // Always the case at the
moment since there is no accreditation scheme for metal-cladding
junctions yet.
    mJNCT_ACCR_DETAILS_MC[0]=0.0;
    mJNCT ACCR DETAILS MC.push back(0.0);
    mJNCT ACCR DETAILS MC.push back(0.0);
    mJNCT_ACCR_DETAILS_MC.push back(0.0);
```

```
    mJNCT ACCR DETAILS MC.push back(0.0);
    mJNCT ACCR DETAILS MC.push back(0.0);
    mJNCT ACCR DETAILS MC.push back(0.0);
    mJNCT ACCR DETAILS MC.push back(0.0);
  }
  double mJNCT CORR MC[8]={1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0};
  double mJNCT ROOF WALL MC=P.Vnum(Id,"JNCT-ROOF-WALL-MC");
  double mJNCT_WALL_GRND_MC=P.Vnum(Id,"JNCT-WALL-GRND-MC");
  double mJNCT WALL WALL MC=P.Vnum(Id,"JNCT-WALL-WALL-MC");
  double mJNCT WALL FLR MC=P.Vnum(Id,"JNCT-WALL-FLR-MC");
  double mJNCT_LINTEL_MC=P.Vnum(Id,"JNCT-LINTEL-MC");
  double mJNCT SILL MC=P.Vnum(Id,"JNCT-SILL-MC");
  double mJNCT JAMB MC=P.Vnum(Id,"JNCT-JAMB-MC");
  double mJNCT_OTHER_MC=P.Vnum(Id,"JNCT-OTHER-MC");

  if(P.Vcodid(cId,"EPC-TYPE")!=4 && P.Vcodid(cId,"BR-TYPE")!=3) {  //
Not for Scotland
    // Correction of psi values as per ADL2A 2010
    double psi106, psiprop, psiadd, psimax, psitemp;
    double dpsi prop1=T.Value("tPSI factors",2,1);  // adjustment for
user defined values
    double  dpsi add1=T.Value("tPSI factors",2,2);      //  adjustment
for user defined values

  // Non-metal cladding junctions

    // ROOF WALL
    psi106=T.Value("tPSI 106",1,1);
    psimax=T.Value("tPSI 106max",1,1);
    psitemp=mJNCT ROOF WALL;
    if(mJNCT ACCR DETAILS[0]==0.0)
      psitemp=psimax;
    else if(mJNCT ACCR DETAILS[0]<0.0) {
          psiprop=psitemp*dpsi prop1; psiadd=psitemp+dpsi add1;
          psitemp=psiprop;     if(psiadd>psiprop)     psitemp=psiadd;
if(psitemp>psimax) psitemp=psimax;
        }
    mJNCT CORR[0]=psitemp;
    // WALL_GRND
    psi106=T.Value("tPSI 106",1,2);
    psimax=T.Value("tPSI 106max",1,2);
    psitemp=mJNCT_WALL_GRND;
    if(mJNCT ACCR DETAILS[1]==0.0)
      psitemp=psimax;
    else if (mJNCT_ACCR_DETAILS[1]<0.0) {
          psiprop=psitemp*dpsi prop1; psiadd=psitemp+dpsi add1;
          psitemp=psiprop;     if(psiadd>psiprop)     psitemp=psiadd;
if(psitemp>psimax) psitemp=psimax;
        }
    mJNCT CORR[1]=psitemp;
    // WALL_WALL
    psi106=T.Value("tPSI 106",1,3);
    psimax=T.Value("tPSI 106max",1,3);
    psitemp=mJNCT_WALL_WALL;
    if(mJNCT ACCR DETAILS[2]==0.0)
      psitemp=psimax;
    else if(mJNCT_ACCR_DETAILS[2]<0.0) {
          psiprop=psitemp*dpsi prop1; psiadd=psitemp+dpsi add1;
          psitemp=psiprop;     if(psiadd>psiprop)     psitemp=psiadd;
if(psitemp>psimax) psitemp=psimax;
        }
    mJNCT CORR[2]=psitemp;
    // WALL_FLR
    psi106=T.Value("tPSI 106",1,4);
    psimax=T.Value("tPSI 106max",1,4);
    psitemp=mJNCT_WALL_FLR;
    if(mJNCT ACCR DETAILS[3]==0.0)
      psitemp=psimax;
    else if(mJNCT_ACCR_DETAILS[3]<0.0) {
```

```
           psiprop=psitemp*dpsi prop1; psiadd=psitemp+dpsi add1;
           psitemp=psiprop;    if(psiadd>psiprop)    psitemp=psiadd;
if(psitemp>psimax) psitemp=psimax;
         }
    mJNCT_CORR[3]=psitemp;
    // LINTEL
    psi106=T.Value("tPSI 106",1,5);
    psimax=T.Value("tPSI_106max",1,5);
    psitemp=mJNCT LINTEL;
    if(mJNCT ACCR DETAILS[4]==0.0)
      psitemp=psimax;
    else if(mJNCT ACCR DETAILS[4]<0.0) {
           psiprop=psitemp*dpsi prop1; psiadd=psitemp+dpsi add1;
           psitemp=psiprop;    if(psiadd>psiprop)    psitemp=psiadd;
if(psitemp>psimax) psitemp=psimax;
         }
    mJNCT_CORR[4]=psitemp;
    // SILL
    psi106=T.Value("tPSI 106",1,6);
    psimax=T.Value("tPSI_106max",1,6);
    psitemp=mJNCT SILL;
    if(mJNCT ACCR DETAILS[5]==0.0)
      psitemp=psimax;
    else if(mJNCT ACCR DETAILS[5]<0.0) {
           psiprop=psitemp*dpsi prop1; psiadd=psitemp+dpsi add1;
           psitemp=psiprop;    if(psiadd>psiprop)    psitemp=psiadd;
if(psitemp>psimax) psitemp=psimax;
         }
    mJNCT CORR[5]=psitemp;
    // JAMB
    psi106=T.Value("tPSI 106",1,7);
    psimax=T.Value("tPSI 106max",1,7);
    psitemp=mJNCT JAMB;
    if(mJNCT ACCR DETAILS[6]==0.0)
      psitemp=psimax;
    else if(mJNCT ACCR DETAILS[6]<0.0) {
           psiprop=psitemp*dpsi prop1; psiadd=psitemp+dpsi add1;
           psitemp=psiprop;    if(psiadd>psiprop)    psitemp=psiadd;
if(psitemp>psimax) psitemp=psimax;
         }
    mJNCT CORR[6]=psitemp;
//    // OTHER
//    psi106=T.Value("tPSI 106",1,8);
//    psimax=T.Value("tPSI 106max",1,8);
//    psitemp=mJNCT_OTHER;
//    if(mJNCT ACCR DETAILS[7]==0.0)
//      psitemp=psimax;
//    else if(mJNCT_ACCR_DETAILS[7]<0.0) {
//           psiprop=psitemp*dpsi prop1; psiadd=psitemp+dpsi add1;
//                psitemp=psiprop;  if(psiadd>psiprop)  psitemp=psiadd;
if(psitemp>psimax) psitemp=psimax;
//         }
    mJNCT CORR[7]=mJNCT OTHER;

     {
      ostringstream res; res << mJNCT CORR[0];
      P.ModKey(Id,"JNCT-ROOF-WALL",res.str());
     }
     {
      ostringstream res; res << mJNCT_CORR[1];
      P.ModKey(Id,"JNCT-WALL-GRND",res.str());
     }
     {
      ostringstream res; res << mJNCT CORR[2];
      P.ModKey(Id,"JNCT-WALL-WALL",res.str());
     }
     {
      ostringstream res; res << mJNCT CORR[3];
      P.ModKey(Id,"JNCT-WALL-FLR",res.str());
```

```
      }
      {
       ostringstream res; res << mJNCT CORR[4];
       P.ModKey(Id,"JNCT-LINTEL",res.str());
      }
      {
       ostringstream res; res << mJNCT CORR[5];
       P.ModKey(Id,"JNCT-SILL",res.str());
      }
      {
       ostringstream res; res << mJNCT_CORR[6];
       P.ModKey(Id,"JNCT-JAMB",res.str());
      }

  // Metal cladding junctions

    // ROOF_WALL
    psi106=T.Value("tPSI 106 MC",1,1);
    psimax=T.Value("tPSI 106 MCmax",1,1);
    psitemp=mJNCT_ROOF_WALL_MC;
    if(mJNCT ACCR DETAILS MC[0]==0.0)
      psitemp=psimax;
    else if(mJNCT ACCR DETAILS MC[0]<0.0) {
           psiprop=psitemp*dpsi prop1; psiadd=psitemp+dpsi add1;
           psitemp=psiprop;     if(psiadd>psiprop)     psitemp=psiadd;
if(psitemp>psimax) psitemp=psimax;
         }
    mJNCT CORR MC[0]=psitemp;
    // WALL GRND
    psi106=T.Value("tPSI 106 MC",1,2);
    psimax=T.Value("tPSI 106 MCmax",1,2);
    psitemp=mJNCT WALL GRND MC;
    if(mJNCT ACCR DETAILS MC[1]==0.0)
      psitemp=psimax;
    else if(mJNCT ACCR DETAILS MC[1]<0.0) {
           psiprop=psitemp*dpsi prop1; psiadd=psitemp+dpsi add1;
           psitemp=psiprop;     if(psiadd>psiprop)     psitemp=psiadd;
if(psitemp>psimax) psitemp=psimax;
         }
    mJNCT CORR MC[1]=psitemp;
    // WALL WALL
    psi106=T.Value("tPSI_106_MC",1,3);
    psimax=T.Value("tPSI 106 MCmax",1,3);
    psitemp=mJNCT WALL WALL MC;
    if(mJNCT_ACCR_DETAILS_MC[2]==0.0)
      psitemp=psimax;
    else if(mJNCT ACCR DETAILS MC[2]<0.0) {
           psiprop=psitemp*dpsi_prop1; psiadd=psitemp+dpsi_add1;
           psitemp=psiprop;     if(psiadd>psiprop)     psitemp=psiadd;
if(psitemp>psimax) psitemp=psimax;
         }
    mJNCT CORR MC[2]=psitemp;
    // WALL FLR
    psi106=T.Value("tPSI_106_MC",1,4);
    psimax=T.Value("tPSI 106 MCmax",1,4);
    psitemp=mJNCT WALL FLR MC;
    if(mJNCT_ACCR_DETAILS_MC[3]==0.0)
      psitemp=psimax;
    else if(mJNCT ACCR DETAILS MC[3]<0.0) {
           psiprop=psitemp*dpsi_prop1; psiadd=psitemp+dpsi_add1;
           psitemp=psiprop;     if(psiadd>psiprop)     psitemp=psiadd;
if(psitemp>psimax) psitemp=psimax;
         }
    mJNCT CORR MC[3]=psitemp;
    // LINTEL
    psi106=T.Value("tPSI_106_MC",1,5);
    psimax=T.Value("tPSI 106 MCmax",1,5);
    psitemp=mJNCT LINTEL MC;
    if(mJNCT_ACCR_DETAILS_MC[4]==0.0)
```

```
      psitemp=psimax;
    else if(mJNCT ACCR DETAILS MC[4]<0.0) {
           psiprop=psitemp*dpsi prop1; psiadd=psitemp+dpsi add1;
           psitemp=psiprop;     if(psiadd>psiprop)     psitemp=psiadd;
if(psitemp>psimax) psitemp=psimax;
         }
    mJNCT CORR MC[4]=psitemp;
    // SILL
    psi106=T.Value("tPSI 106 MC",1,6);
    psimax=T.Value("tPSI 106 MCmax",1,6);
    psitemp=mJNCT_SILL_MC;
    if(mJNCT ACCR DETAILS MC[5]==0.0)
      psitemp=psimax;
    else if(mJNCT_ACCR_DETAILS_MC[5]<0.0) {
           psiprop=psitemp*dpsi prop1; psiadd=psitemp+dpsi add1;
           psitemp=psiprop;     if(psiadd>psiprop)     psitemp=psiadd;
if(psitemp>psimax) psitemp=psimax;
         }
    mJNCT CORR MC[5]=psitemp;
    // JAMB
    psi106=T.Value("tPSI 106 MC",1,7);
    psimax=T.Value("tPSI 106 MCmax",1,7);
    psitemp=mJNCT JAMB MC;
    if(mJNCT ACCR DETAILS MC[6]==0.0)
      psitemp=psimax;
    else if(mJNCT ACCR DETAILS MC[6]<0.0) {
           psiprop=psitemp*dpsi prop1; psiadd=psitemp+dpsi add1;
           psitemp=psiprop;     if(psiadd>psiprop)     psitemp=psiadd;
if(psitemp>psimax) psitemp=psimax;
         }
    mJNCT CORR MC[6]=psitemp;
//    // OTHER
//    psi106=T.Value("tPSI 106 MC",1,8);
//    psimax=T.Value("tPSI 106 MCmax",1,8);
//    psitemp=mJNCT OTHER MC;
//    if(mJNCT ACCR DETAILS MC[7]==0.0)
//      psitemp=psimax;
//    else if(mJNCT ACCR DETAILS MC[7]<0.0) {
//           psiprop=psitemp*dpsi_prop1; psiadd=psitemp+dpsi_add1;
//                psitemp=psiprop;  if(psiadd>psiprop)  psitemp=psiadd;
if(psitemp>psimax) psitemp=psimax;
//         }
    mJNCT CORR MC[7]=mJNCT OTHER MC;

     {
      ostringstream res; res << mJNCT CORR MC[0];
      P.ModKey(Id,"JNCT-ROOF-WALL-MC",res.str());
     }
     {
      ostringstream res; res << mJNCT CORR MC[1];
      P.ModKey(Id,"JNCT-WALL-GRND-MC",res.str());
     }
     {
      ostringstream res; res << mJNCT_CORR_MC[2];
      P.ModKey(Id,"JNCT-WALL-WALL-MC",res.str());
     }
     {
      ostringstream res; res << mJNCT CORR MC[3];
      P.ModKey(Id,"JNCT-WALL-FLR-MC",res.str());
     }
     {
      ostringstream res; res << mJNCT CORR MC[4];
      P.ModKey(Id,"JNCT-LINTEL-MC",res.str());
     }
     {
      ostringstream res; res << mJNCT_CORR_MC[5];
      P.ModKey(Id,"JNCT-SILL-MC",res.str());
     }
     {
```

```
      ostringstream res; res << mJNCT CORR MC[6];
      P.ModKey(Id,"JNCT-JAMB-MC",res.str());
     }

  }
  else { // Scotland
    mJNCT CORR[0]=mJNCT ROOF WALL;
    mJNCT_CORR[1]=mJNCT_WALL_GRND;
    mJNCT CORR[2]=mJNCT WALL WALL;
    mJNCT CORR[3]=mJNCT WALL FLR;
    mJNCT_CORR[4]=mJNCT_LINTEL;
    mJNCT CORR[5]=mJNCT SILL;
    mJNCT CORR[6]=mJNCT JAMB;
    mJNCT_CORR[7]=mJNCT_OTHER;

    mJNCT CORR MC[0]=mJNCT ROOF WALL MC;
    mJNCT_CORR_MC[1]=mJNCT_WALL_GRND_MC;
    mJNCT CORR MC[2]=mJNCT WALL WALL MC;
    mJNCT CORR MC[3]=mJNCT WALL FLR MC;
    mJNCT_CORR_MC[4]=mJNCT_LINTEL_MC;
    mJNCT CORR MC[5]=mJNCT SILL MC;
    mJNCT CORR MC[6]=mJNCT JAMB MC;
    mJNCT CORR MC[7]=mJNCT OTHER MC;
  }

  mH TR TM=mJNCT L ROOF WALL MC*mJNCT CORR MC[0] +
           mJNCT L WALL GRND MC*mJNCT CORR MC[1] +
           mJNCT L WALL WALL MC*mJNCT CORR MC[2] +
           mJNCT L WALL FLR MC*mJNCT CORR MC[3] +
           mJNCT L LINTEL MC*mJNCT CORR MC[4] +
           mJNCT L SILL MC*mJNCT CORR MC[5] +
           mJNCT L JAMB MC*mJNCT CORR MC[6] +
           mJNCT L OTHER MC*mJNCT CORR MC[7] +
           mJNCT L ROOF WALL*mJNCT CORR[0] +
           mJNCT L WALL GRND*mJNCT CORR[1] +
           mJNCT L WALL WALL*mJNCT CORR[2] +
           mJNCT L WALL FLR*mJNCT CORR[3] +
           mJNCT L LINTEL*mJNCT CORR[4] +
           mJNCT_L_SILL*mJNCT_CORR[5] +
           mJNCT L JAMB*mJNCT CORR[6] +
           mJNCT L OTHER*mJNCT CORR[7];

 {
  ostringstream res; res << mH TR TM;
  P.ModKey(Id,"H-TR-TM",res.str());
 }
 {
  ostringstream res; res << mJNCT_L_ROOF_WALL_MC;
  P.ModKey(Id,"JNCT-L-ROOF-WALL-MC",res.str());
 }
 {
  ostringstream res; res << mJNCT L WALL GRND MC;
  P.ModKey(Id,"JNCT-L-WALL-GRND-MC",res.str());
 }
 {
  ostringstream res; res << mJNCT L WALL WALL MC;
  P.ModKey(Id,"JNCT-L-WALL-WALL-MC",res.str());
 }
 {
  ostringstream res; res << mJNCT_L_WALL_FLR_MC;
  P.ModKey(Id,"JNCT-L-WALL-FLR-MC",res.str());
 }
 {
  ostringstream res; res << mJNCT L LINTEL MC;
  P.ModKey(Id,"JNCT-L-LINTEL-MC",res.str());
 }
 {
  ostringstream res; res << mJNCT L SILL MC;
  P.ModKey(Id,"JNCT-L-SILL-MC",res.str());
```

```
  }
  {
   ostringstream res; res << mJNCT L JAMB MC;
   P.ModKey(Id,"JNCT-L-JAMB-MC",res.str());
  }
  {
   ostringstream res; res << mJNCT L OTHER MC;
   P.ModKey(Id,"JNCT-L-OTHER-MC",res.str());
  }
  {
   ostringstream res; res << mJNCT_L_ROOF_WALL;
   P.ModKey(Id,"JNCT-L-ROOF-WALL",res.str());
  }
  {
   ostringstream res; res << mJNCT L WALL GRND;
   P.ModKey(Id,"JNCT-L-WALL-GRND",res.str());
  }
  {
   ostringstream res; res << mJNCT L WALL WALL;
   P.ModKey(Id,"JNCT-L-WALL-WALL",res.str());
  }
  {
   ostringstream res; res << mJNCT L WALL FLR;
   P.ModKey(Id,"JNCT-L-WALL-FLR",res.str());
  }
  {
   ostringstream res; res << mJNCT L LINTEL;
   P.ModKey(Id,"JNCT-L-LINTEL",res.str());
  }
  {
   ostringstream res; res << mJNCT L SILL;
   P.ModKey(Id,"JNCT-L-SILL",res.str());
  }
  {
   ostringstream res; res << mJNCT L JAMB;
   P.ModKey(Id,"JNCT-L-JAMB",res.str());
  }
  {
   ostringstream res; res << mJNCT_L_OTHER;
   P.ModKey(Id,"JNCT-L-OTHER",res.str());
  }

}

void SPACE::fH_TR(PROJECTDefaults& D, BOC& P) {
  if(func report) cout << endl << "  " << "SPACE::fH TR";
//calculate heat transfer coefficient for zone

  double mH TR=0;
  for (int i=0;i<vW.size();i++)
    mH_TR+=P.Vnum(vW[i].Id,"H-TR")*P.Vnum(vW[i].Id,"MULTIPLIER");
  mH TR+=P.Vnum(Id,"H-TR-TM");

 {
  ostringstream res; res << mH TR;
  P.ModKey(Id,"H-TR",res.str());  //includes doors and smoke vents
 }

}

void SPACE::fCM(PROJECTDefaults& D, BOC& P) {
  if(func report) cout << endl << "  " << "SPACE::fCM";
//sums up (cm * net area) for all walls in zone. Units: kJ/m2K * m2=
kJ/K
// It also sums up the same for doors.

  deque<double> mCM WALL;
  deque<double> mCM DOOR;
  double mCM[2]={0, 0};
```

```
  for (int i=0;i<vW.size();i++)  {

mCM WALL=P.Vlnum Mul(vW[i].Id,"CM",P.Vnum(vW[i].Id,"MULTIPLIER"));
    for (int j=0;j<2;j++)
      mCM[j]+=mCM WALL[j];

    // AN (15/01/2010): adding the thermal mass of doors in the zone
to that of walls above
    for(int k=0;k<vW[i].vD.size();k++) {

mCM DOOR=P.Vlnum Mul(vW[i].vD[k].Id,"CM",(P.Vnum(vW[i].vD[k].Id,"MULT
IPLIER")*P.Vnum(vW[i].Id,"MULTIPLIER")));
      for (int j=0;j<2;j++)
        mCM[j]+=mCM DOOR[j];
    } // end for

  } // end for

 {
  ostringstream res; res << "{ ";
  res << mCM[0] << ", ";
  res << mCM[1] << " }";
  P.ModKey(Id,"CM",res.str());    //includes doors
 }
}

//call after fH TR and fCM and before fQ TR HEAT

void SPACE::fTAU HEAT(PROJECTDefaults& D, BOC& P) {
  if(func report) cout << endl << "  " << "SPACE::fTAU HEAT";
//calculatate time constant for heating TAU-HEAT, in hours

//  double mTAU=0;
  double mTAU[12]={0.0, 0.0 ,0.0 ,0.0 ,0.0 ,0.0 ,0.0 ,0.0 ,0.0 ,0.0
,0.0 ,0.0};
  deque<double> mCM=P.Vlnum(Id,"CM");
  double mH TR=P.Vnum(Id,"H-TR");
  deque<double> mH_VENT_HEAT_M=P.Vlnum(Id,"H-VENT-HEAT");
  deque<double> mH INF HEAT M=P.Vlnum(Id,"H-INF-HEAT");
//  double mH VENT HEAT=0.0;
//                                  for(int                i=0;i<12;i++)
mH VENT HEAT+=(mH VENT HEAT M[i]+mH INF HEAT M[i])/12;
  double  mH VENT HEAT[12]={0.0,  0.0  ,0.0  ,0.0  ,0.0  ,0.0  ,0.0  ,0.0
,0.0 ,0.0 ,0.0 ,0.0 ,0.0};
  for(int i=0;i<12;i++)
    mH VENT HEAT[i]=mH VENT HEAT M[i]+mH INF HEAT M[i];
//mTAU now set to 0 if mH TR and mH VENT HEAT are both 0 (internal
zones)
//check effect on calcs in fCOOL INTERMITTANCE, fHEAT INTERMITTANCE,
fR_B_COOL AND fR_B_HEAT
  for(int i=0;i<12;i++) {
    if (mH TR+mH VENT HEAT[i]==0)
      mTAU[i]=-1;
// Do check always later if mTAU is -1 before using it!!!
    else
      mTAU[i]=(mCM[1]/3.6)/(mH_TR+mH_VENT_HEAT[i]);    // hours
  } // end for

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mTAU[i] << ", ";
   res << mTAU[11] << " }";
   P.ModKey(Id,"TAU-HEAT",res.str());
  }

}
```

```
void SPACE::fTAU COOL(PROJECTDefaults& D, BOC& P) {
  if(func report) cout << endl << "  " << "SPACE::fTAU COOL";
//calculatate time constant for cooling TAU-COOL, in hours

//  double mTAU=0;
  double mTAU[12]={0.0, 0.0 ,0.0 ,0.0 ,0.0 ,0.0 ,0.0 ,0.0 ,0.0 ,0.0
,0.0 ,0.0};
  deque<double> mCM=P.Vlnum(Id,"CM");
  double mH TR=P.Vnum(Id,"H-TR");
  deque<double> mH VENT COOL M=P.Vlnum(Id,"H-VENT-COOL");
  deque<double> mH_INF_COOL_M=P.Vlnum(Id,"H-INF-COOL");
  deque<double>  mH VENT COOL NC M=P.Vlnum(Id,"H-VENT-COOL-NC");    //
night cooling if available
//  double mH_VENT_COOL=0;
//                                  for(int                i=0;i<12;i++)
mH VENT COOL+=(mH VENT COOL M[i]+mH INF COOL M[i])/12;
  double  mH_VENT_COOL[12]={0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,
0.0, 0.0, 0.0, 0.0};
  for(int i=0;i<12;i++)
    mH_VENT_COOL[i]=mH_VENT_COOL_M[i]+mH_INF_COOL_M[i];
  for(int i=0;i<12;i++) {
    if (mH TR+mH VENT COOL[i]+mH VENT COOL NC M[i]==0)
      mTAU[i]=-1;
// Do check always later if mTAU is -1 before using it!!!
    else

mTAU[i]=(mCM[1]/3.6)/(mH TR+mH VENT COOL[i]+mH VENT COOL NC M[i]);
  } // end for

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mTAU[i] << ", ";
   res << mTAU[11] << " }";
   P.ModKey(Id,"TAU-COOL",res.str());
  }
//cout  <<  endl  <<  endl  <<  P.Vtext(Id,"H-VENT-COOL-NC")  <<  endl;
system("PAUSE");
}

void SPACE::fHEAT QUASI CONTINUOUS(PROJECTDefaults& D, PROJECTTables&
T, BOC& P) {
  if(func report)    cout    <<    endl    <<    "         "    <<
"SPACE::fHEAT QUASI CONTINUOUS";
  //modifies heating set point to account for intermittent heating
(nights, weekends and holidays)
  //as discribed in CEN document pr EN wi 14:2004
  //the  adjusted  internal  temperature  is  the  constant  internal
temperature which would result in the same heat loss as that obtained
with intermittent heating during the period

  deque<double> mHEAT SET POINT=P.Vlnum(Id,"HEAT-SET-POINT");
  deque<double> mSB SET POINT=P.Vlnum(Id,"SB-SET-POINT");
  deque<double> mHEAT_SET_AVE=P.Vlnum(Id,"HEAT-SET-AVE");

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mHEAT SET POINT[i] << ", ";
  res << mHEAT_SET_POINT[11] << " }";
  P.ModKey(Id,"HEAT-SET-POINT-ORI",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mSB_SET_POINT[i] << ", ";
  res << mSB SET POINT[11] << " }";
  P.ModKey(Id,"SB-SET-POINT-ORI",res.str());
 }
```

```
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mHEAT SET AVE[i] << ", ";
  res << mHEAT_SET_AVE[11] << " }";
  P.ModKey(Id,"HEAT-SET-AVE-ORI",res.str());
 }

  deque<double> mOCC T START=P.Vlnum(Id,"OCC-T-START");
  deque<double> mOCC T END=P.Vlnum(Id,"OCC-T-END");
  deque<double> mMAX_TIME_REDUCED=P.Vlnum(Id,"MAX-TIME-REDUCED");
  deque<double> mMIN TIME REDUCED=P.Vlnum(Id,"MIN-TIME-REDUCED");
  deque<double>  mTAU=P.Vlnum(Id,"TAU-HEAT");   //mTAU may be -1 if
heat loss coeffs are zero (internal zones)
  double mOCC TIME;
  string mWEATHER;
  if(gId<0) mWEATHER="_LON"; else mWEATHER="_"+P.Vcod(gId,"WEATHER");

  // For CEN2008, flag to denote if neither Mode A nor B applies and
a separate calculation would be required
  int mHEAT INTERMIT FLAG[12]={0,0,0,0,0,0,0,0,0,0,0,0};

  for (int i=0;i<12;i++) {
    //case O (CEN p.58), use time average of set point temperatures:
    if(mSB SET POINT[i]==-7777) {
      mSB SET POINT[i]=T.Value("T003"+mWEATHER,1,i+1);
      mOCC TIME=mOCC T END[i]-mOCC T START[i];

mHEAT SET AVE[i]=(mHEAT SET POINT[i]*mOCC TIME+mSB SET POINT[i]*(24-
mOCC TIME))/24;
    }
    // If the set point temperature variations between normal and
reduced heating is less than 3K, use the time average
    // of set point temperatures in the calculation (Mode A - part
1):
    if  ((mHEAT SET POINT[i]-mSB SET POINT[i])<=D.Value("INT HEAT 1"))
{
      mHEAT SET POINT[i]=mHEAT SET AVE[i];
    }
    else  {
      // If the time constant of the zone is less than 0.2*the
duration of the shortest reduced heating period, use the
      // time average of set point temperatures in the calculation
(Mode A - part 2):
      if(mTAU[i]<(D.Value("INT HEAT 2")*mMIN TIME REDUCED[i])      &&
mTAU[i]!=-1) //if mTAU[i]=-1, it should be considered to have a high
value
        mHEAT_SET_POINT[i]=mHEAT_SET_AVE[i];

      // If the time constant of the zone is greater than 3*the
duration of the longest reduced heating period, use the
      // set point temperature for the normal heating mode in the
calculation (Mode B), i.e., there is no need to amend
      // mHEAT_SET_POINT which already contains the keyword HEAT-SET-
POINT.

      else { // note that the condition below is the inverse of Mode
B
        if(mTAU[i]<=(D.Value("INT HEAT 3")*mMAX TIME REDUCED[i])    &&
mTAU[i]!=-1) {   //if mTAU[i]=-1, it should be considered to have a
high value

          // Check which version of CEN 13790 (2004 or 2008) is
applied for calculating the intermittance
          if (P.Vcodid(gId,"INTERMIT-CALC")==1) {  // 1=CEN2004
            //for case A use normal set-point temperature (CEN p.58)
            //if case is not case O, A or B, interpolate:
            //interpolate  between  the  ave  and  set  point  temp
depending on the TAU value:
```

```
            double mHEAT SET POINT CALC=0.0;
            if         (((D.Value("INT HEAT 3")*mMAX TIME REDUCED[i])-
(D.Value("INT HEAT 2")*mMIN TIME REDUCED[i]))!=0.0)  {
              mHEAT SET POINT CALC=mHEAT SET POINT[i]-
((mHEAT_SET_POINT[i]-
mHEAT SET AVE[i])*((D.Value("INT HEAT 3")*mMAX TIME REDUCED[i])-
mTAU[i])/((D.Value("INT HEAT 3")*mMAX TIME REDUCED[i])-
(D.Value("INT_HEAT_2")*mMIN_TIME_REDUCED[i])));
              mHEAT SET POINT[i]=mHEAT SET POINT CALC;
            }
          } // end if
          else  // i.e., CEN2008 is applied
            // In the case where the intermittent heating does not
fulfil any of the above clauses, a different calculation is
            //  performed which requires the calculation of a
dimensionless reduction factor for intermittent heating which is
            // then multiplied by the energy demand calculated for
continuous heating. This is done in the function
            // fHEAT INTERMITTANCE. The check below sets a flag that
will trigger the fHEAT_INTERMITTANCE function later once
            // the heating demand has been calculated.
            mHEAT INTERMIT FLAG[i]=1;     // mark the flag for this
month for a different intermittency calculation later
        } // end if
      } // end else
    }
  }

//temperature gradient correction (if there are no destratification
fans in zone):
  if (P.Vcod(Id,"DESTRAT-FAN")=="NO") {
    deque<double> mHEAT SET POINT ORI=P.Vlnum(Id,"HEAT-SET-POINT");
    double mGRAD=P.Vnum(pId,"GRAD");

    // For the notional building, adjust the temperature gradient
coefficient based on the zone's glazing
    // classification and servicing strategy
    if (P.Vcod(gId,"ACT-NOT")=="NOT") {
      //SERVICE-STRATEGY categories are:
      // 1)Heated and Naturally Ventilated
      // 2)Heated and Mechanically Ventilated
      // 3)Air Conditioned
      // 4)No heating or cooling
      // 5)Mixed-mode and Naturally Ventilated
      // 6)Mixed-mode and Mechanically Ventilated
      if ( (P.Vnum(Id,"SERVICE-STRATEGY")==3) || (P.Vnum(Id,"SERVICE-
STRATEGY")==4) )
        mGRAD=T.Value("Not_TEMP_GRAD",2,P.Vint(Id,"NOT-CASE"));
      else // i.e., there is heating only or heating and mixed-mode
cooling
        mGRAD=T.Value("Not_TEMP_GRAD",1,P.Vint(Id,"NOT-CASE"));
    } // end if

    double mHEIGHT=P.Vnum(Id,"HEIGHT");

    for (int i=0;i<12;i++)  {
      mHEAT_SET_POINT[i]+=mGRAD*(mHEIGHT/D.Value("dTEMP_GRAD1")-
D.Value("dTEMP GRAD2"));

      //if adjusted set point is > outside air temp, set set point to
outside air temp
      if  ((mHEAT SET POINT ORI[i]<T.Value("T003"+mWEATHER,1,i+1))  &&
(mHEAT_SET_POINT[i]>T.Value("T003"+mWEATHER,1,i+1)))
mHEAT SET POINT[i]=T.Value("T003"+mWEATHER,1,i+1);
    }
  }

 {
  ostringstream res; res << "{ ";
```

```
  for (int i=0;i<11;i++)
    res << mHEAT SET POINT[i] << ", ";
  res << mHEAT SET POINT[11] << " }";
  P.ModKey(Id,"HEAT-SET-POINT",res.str());
 }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mHEAT INTERMIT FLAG[i] << ", ";
  res << mHEAT_INTERMIT_FLAG[11] << " }";
  P.ModKey(Id,"HEAT-INTERMIT-FLAG",res.str());
 }

}

void SPACE::fQ TR HEAT(PROJECTDefaults& D, PROJECTTables& T, BOC& P)
{
  if(func report) cout << endl << "  " << "SPACE::fQ TR HEAT";
//calculate transmission heat transfer for zone for heating

  double mH TR=P.Vnum(Id,"H-TR");
  double mQ TR HEAT[12];
  deque<double> mHEAT SET POINT;

  mHEAT SET POINT=P.Vlnum(Id,"HEAT-SET-POINT");
  string mWEATHER;
  if(gId<0) mWEATHER=" LON"; else mWEATHER=" "+P.Vcod(gId,"WEATHER");
  for (int i=0;i<12;i++) {
    if (mHEAT SET POINT[i]!=-9999)

mQ TR HEAT[i]=mH TR*D.Value("D008")*T.Value("NJ",1,i+1)*(mHEAT SET PO
INT[i]-T.Value("T003"+mWEATHER,1,i+1));
  }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ TR HEAT[i] << ", ";
  res << mQ_TR_HEAT[11] << " }";
  P.ModKey(Id,"Q-TR-HEAT",res.str());
 }
}

void SPACE::fQ TR COOL(PROJECTDefaults& D, PROJECTTables& T, BOC& P)
{
  if(func report) cout << endl << "  " << "SPACE::fQ TR COOL";
//calculate transmission heat transfer for zone for cooling

  double mH TR=P.Vnum(Id,"H-TR");
  double mQ TR COOL[12]={0.0, 0.0 ,0.0 ,0.0 ,0.0 ,0.0 ,0.0 ,0.0 ,0.0
,0.0 ,0.0 ,0.0};
  deque<double> mCOOL SET POINT;

  mCOOL_SET_POINT=P.Vlnum(Id,"COOL-SET-POINT");

  // JO 01Jan08. Modification of Set Point for Mixed-mode calculation
in both
  // Actual and Notional building. The correction below (23Sep07) is
for overheating (general)
  // and Mixed-mode in Reference building.
  if(P.Vcod(pId,"MIXED-MODE")=="YES")
    for (int i=0;i<12;i++) {
     if (mCOOL_SET_POINT[i]!=-9999)
      mCOOL SET POINT[i]=D.Value("DMMTemp");
    }

  // JO 23Sep07. Modification of Set Point for Overheating and Mixed-
mode calculation
  int mCOOL_SET_POINT_MODE=P.Vcodid(gId,"COOL-SET-POINT-MODE");
```

```
  switch (mCOOL SET POINT MODE) {
  case 2:
       {
            for (int i=0;i<12;i++) {
             if (mCOOL SET POINT[i]!=-9999)
              mCOOL SET POINT[i]=D.Value("DOHTemp");
            }
       } break;
  case 3:
       {
            for (int i=0;i<12;i++) {
             if (mCOOL SET POINT[i]!=-9999)
              mCOOL_SET_POINT[i]=D.Value("DMMTemp");
            }
       } break;
  }

    string mWEATHER;
    if(gId<0)                    mWEATHER=" LON";                    else
mWEATHER=" "+P.Vcod(gId,"WEATHER");
    for (int i=0;i<12;i++) {
     if (mCOOL SET POINT[i]!=-9999)

mQ TR COOL[i]=mH TR*D.Value("D008")*T.Value("NJ",1,i+1)*(mCOOL SET PO
INT[i]-T.Value("T003"+mWEATHER,1,i+1));
    }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ TR COOL[i] << ", ";
  res << mQ TR COOL[11] << " }";
  switch (mCOOL SET POINT MODE) {
  case 1:
       {
          P.ModKey(Id,"Q-TR-COOL",res.str());
       } break;
  case 2:
       {
          P.ModKey(Id,"Q-TR-COOL-OH",res.str());
       } break;
  case 3:
       {
          P.ModKey(Id,"Q-TR-COOL",res.str());
       } break;
  }
 }

}

//ventilation:
void  SPACE::fVENT ZONE HVAC(PROJECTDefaults&  D,  PROJECTTables&  T,
BOC& P) {
  if(func report) cout << endl << "  " << "SPACE::fVENT ZONE HVAC";
//sets heat recovery option for zone from parent HVAC and ventilation
type keyword (Q-V-TYPE)

  //if ventilation is at HVAC level, copy keywrds from HVAC to zone
  if (P.Vcodid(Id,"VENT-ZONE-HVAC")==2) {
    // Check if this is the actual building (notional building is set
in the Notional function of ZONE)
    if (P.Vcod(gId,"ACT-NOT")=="ACT")  {
      //heat recovery
      {
       ostringstream res; res << P.Vcod(pId,"HEAT-REC-SYSTEM");
       P.ModKey(Id,"HEAT-REC-SYSTEM",res.str());
      }
      //v2.0.a  -  now  using  heat  recovery  efficiency  from  iSBEM:
```

```
keyword HEAT-REC-SEFF
      {
       ostringstream res; res << P.Vnum(pId,"HEAT-REC-SEFF");
       P.ModKey(Id,"HEAT-REC-SEFF",res.str());
      }
      {
       ostringstream res; res << P.Vcod(pId,"HEAT-REC-VAR-EFF");
       P.ModKey(Id,"HEAT-REC-VAR-EFF",res.str());
      }
    } // end else
  }
}

void SPACE::fF_VENT_M(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "   " << "SPACE::fF VENT M";
//sets keyword "F-VENT-M"

  deque<double> mF VENT M;

  //store in mF VENT M the number of hours per month which system
operates:
  if  (T.Value("tSYS COOL",1,P.Vcodid(pId,"TYPE"))==1)  //system has
cooling: use sys-t-hours
    mF VENT M=P.Vlnum(Id,"SYS-T-HOURS");
  else
    mF VENT M=P.Vlnum(Id,"SYS-HEAT-T-HOURS");      //system   has   no
cooling: use sys-heat-t-hours

  for (int i=0;i<12;i++)
    mF VENT M[i]*=D.Value("dF Vent")/(T.Value("NJ",1,i+1)*24);


 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mF VENT M[i] << ", ";
  res << mF VENT M[11] << " }";
  P.ModKey(Id,"F-VENT-M",res.str());
 }

}

void SPACE::fU V M(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "   " << "SPACE::fU V M";
    //sets ventilation flow rate keywords for zone: "U-V-M" for mech
ventilation and "U-V-N" for natural ventilation
//these are air flow rates in l/s/m2

  double mAREA=P.Vnum(Id,"AREA");
  deque<double>              mOA FLOW PERS=P.Vlnum(Id,"OA-FLOW-PERS");
//lh/s/m2/month
  int i;


//divide mOA FLOW PERS by mOCC T HOURS to get l/s/m2:
  if(P.Vcod(Id,"ACT-DBS-ACU")=="YES") {

// New corrections Jan09
  deque<double>                mOCC T HOURS=P.Vlnum(Id,"OCC-T-HOURS");
//hours/month

  for (i=0;i<12;i++)
    if (mOCC_T_HOURS[i]!= 0.00)  mOA_FLOW_PERS[i]/=mOCC_T_HOURS[i];
    else mOA FLOW PERS[i]=0.00;

// End New corrections Jan09

// Correction and new code from 10 Sep 08 starts here:
// The correction implies to divide the Database OA-FLOW-PERS by the
```

```
total hours of the month
//     for (i=0;i<12;i++) mOA FLOW PERS[i]/=(T.Value("NJ",1,i+1)*24);
//mSYS T HOURS[i];
//   The intention is to reproduce eventually the following
calculation:
// Q-VENT-HEAT m=1.2*3600*OA-FLOW-PERS m*Area*DeltaT m/1e6 in MJ,  m
refers to month m. OA-FLOW-PERS m is the value coming from the
database (ACU). Q-VENT-HEAT m is the total losses due to ventilation
losses in month m.

  }
  deque<double> mU V M;         //mechanical air flow rate l/s/m2
  deque<double> mU V N;

  if (P.Vcod(Id,"Q-V-TYPE")=="Mechanical") {
    for (i=0;i<12;i++) {
      mU_V_M[i]=mOA_FLOW_PERS[i];
      mU V N[i]=0;
    }
  }
  else  {
    for (i=0;i<12;i++) {
      mU V M[i]=0.0;
      mU V N[i]=mOA FLOW PERS[i];
    }
  }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mU V M[i] << ", ";
  res << mU V M[11] << " }";
  P.ModKey(Id,"U-V-M",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mU V N[i] << ", ";
  res << mU_V_N[11] << " }";
  P.ModKey(Id,"U-V-N",res.str());
 }
}

void  SPACE::fH VENT HEAT(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func report) cout << endl << "  " << "SPACE::fH VENT HEAT";
//calculate ventilation heat transfer coefficient

  double mAREA=P.Vnum(Id,"AREA");
  int i;

  deque<double> U V M E=P.Vlnum(Id,"U-V-M");
  deque<double> U V N=P.Vlnum(Id,"U-V-N");

  double  F CTRL VENT=1;   //this is now always 1 as we are not now
including recirc or flow reduction
  double R_HR=0.0;

//efficiency of heat recovery from input:
  if (P.Vcodid(Id,"HEAT-REC-SYSTEM")!=1) { //system has heat recovery
    R HR=P.Vnum(Id,"HEAT-REC-SEFF");
  }
  else {
   R HR=0; //system has no heat recovery
  }

  deque<double> mBR LIGHT CHECK=P.Vlnum(Id,"BR-LIGHT-CHECK");

double mOUT_ENV_AREA=0;
```

```
 //calculation of the space outer envelope area:
for(int i=0;i<vW.size();i++)
//external envelope only:
  if (P.Vcodid(vW[i].Id,"TYPE")==1)

mOUT ENV AREA+=P.Vnum(vW[i].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER");
// Check multiplier

double mU INF WIND ZONE[12]={0,0,0,0,0,0,0,0,0,0,0,0};
double mU INF STACK ZONE[12]={0,0,0,0,0,0,0,0,0,0,0,0};
double mU_INF_TOT_ZONE[12]={0,0,0,0,0,0,0,0,0,0,0,0};
deque<double> mU INF WIND WALL;
deque<double> mU INF STACK WALL;

double mQ50 INF=P.Vnum(Id,"Q50-INF");
double mQ4 INF=mQ50 INF*T.Value("tn50Q4",1,(int)mBR LIGHT CHECK[7]);
if                                                    (mQ4 INF==0)
mQ4 INF=0.01*T.Value("tn50Q4",1,(int)mBR LIGHT CHECK[7]);
//deque<double>  mHEAT SET POINT=P.Vlnum(Id,"HEAT-SET-POINT");      //
not used in this function
//deque<double> mCOOL SET POINT=P.Vlnum(Id,"COOL-SET-POINT");

//Calculation of the infiltrations due to wind in a whole zone as the
addition of the contribution of each wall

string mWEATHER;
if (gId<0) mWEATHER=" LON"; else mWEATHER=" "+P.Vcod(gId,"WEATHER");

for(int i=0;i<vW.size();i++) {
   if (mOUT ENV AREA!=0)
     mU INF WIND WALL=P.Vlnum Mul(vW[i].Id,"U-INF-WIND-
WALL",P.Vnum(vW[i].Id,"MULTIPLIER")*P.Vnum(vW[i].Id,"AREA")/mOUT ENV
AREA);
   else {
     for(int j=0;j<vW.size();j++)
       mU INF WIND WALL[j]==0;
   }
   for(int month=0;month<12;month++)
     mU INF WIND ZONE[month]+=mU INF WIND WALL[month];   //m3/hm2   of
all zones outer envelopes
}

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mU INF WIND ZONE[i] << ", ";
  res << mU INF WIND ZONE[11] << " }";
  P.ModKey(Id,"U-INF-WIND-ZONE-H",res.str());
 }

//Calculation of the infiltrations due to stack in a whole zone as
the addition of the contribution of each wall

for(int i=0;i<vW.size();i++) {
   if (mOUT ENV AREA!=0)
     mU INF STACK WALL=P.Vlnum Mul(vW[i].Id,"U-INF-STACK-
WALL",P.Vnum(vW[i].Id,"MULTIPLIER")*P.Vnum(vW[i].Id,"AREA")/mOUT_ENV_
AREA);
   else {
     for(int j=0;j<vW.size();j++)
       mU INF STACK WALL[j]==0;
   }
   for(int month=0;month<12;month++)
     mU INF STACK ZONE[month]+=mU INF STACK WALL[month];  //m3/hm2  of
all zones outer envelopes
}

 {
  ostringstream res; res << "{ ";
```

```
  for (int i=0;i<11;i++)
    res << mU INF STACK ZONE[i] << ", ";
  res << mU INF STACK ZONE[11] << " }";
  P.ModKey(Id,"U-INF-STACK-ZONE-H",res.str());
 }

 for(int month=0;month<12;month++) {
   if(mAREA==0) mU_INF_WIND_ZONE[month]=-1;
   else {

mU_INF_TOT_ZONE[month]=max(mU_INF_STACK_ZONE[month],mU_INF_WIND_ZONE[
month])+D.Value("dU INF7")*((mU INF STACK ZONE[month]*mU INF WIND ZON
E[month])/mQ4 INF);
     mU_INF_TOT_ZONE[month]*=mOUT ENV AREA/mAREA;      //to   convert
m3/hm2 outer envelope to m3/hm2 floor area
   }
 }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mU INF TOT ZONE[i] << ", ";
  res << mU INF TOT ZONE[11] << " }";
  P.ModKey(Id,"U-INF-TOT-ZONE-H",res.str());
 }

  double U V INF=0;
  for(int i=0;i<12;i++) U V INF+=mU INF TOT ZONE[i]/12;

  double U V HEAT[12];
//   for(int  i=0;i<12;i++)  U V HEAT[i]=mU INF TOT ZONE[i]+F VENT*((1-
R HR)*U V M E[i]+U V N[i]);
//     for(int   i=0;i<12;i++)   U V HEAT[i]=mU INF TOT ZONE[i]/3.6+(1-
R HR)*U V M E[i]+U V N[i];
  for(int i=0;i<12;i++) U V HEAT[i]=(1-R HR)*U V M E[i]+U V N[i];

  double mH VENT HEAT[12];
  for(int                                                  i=0;i<12;i++)
mH_VENT_HEAT[i]=D.Value("D014")*mAREA*U_V_HEAT[i];

  double mH INF HEAT[12];
  for(int                                                  i=0;i<12;i++)
mH INF HEAT[i]=D.Value("D014")*mAREA*mU INF TOT ZONE[i]/3.6;
  // The division by 3.6 in the above line is to convert m3/h.m2 to
l/s.m2

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mH VENT HEAT[i] << ", ";
  res << mH_VENT_HEAT[11] << " }";
  P.ModKey(Id,"H-VENT-HEAT",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mH_INF_HEAT[i] << ", ";
  res << mH INF HEAT[11] << " }";
  P.ModKey(Id,"H-INF-HEAT",res.str());
 }
}

void  SPACE::fH VENT COOL(PROJECTDefaults&  D,  PROJECTTables&  T,  BOC&
P) {
  if(func report) cout << endl << "  " << "SPACE::fH VENT COOL";
//calculate ventilation heat transfer coefficient for cooling

  double mAREA=P.Vnum(Id,"AREA");
  deque<double> mU_V_M=P.Vlnum(Id,"U-V-M");
```

```
  deque<double> mBR LIGHT CHECK=P.Vlnum(Id,"BR-LIGHT-CHECK");
  //deque<double> mF VENT M=P.Vlnum(Id,"F-VENT-M");
  //double U V N COOL[12];
  //for(int                                                    i=0;i<12;i++)
U_V_N_COOL[i]=T.Value("T026",1,P.Vcodid(Id,"Q-V-
FACILITY"))*mF VENT M[i];
  // Now calculating the ventilation heat loss coefficient in cooling
in the same way as for heating - AN - 22/10/2010
  deque<double> U V N COOL=P.Vlnum(Id,"U-V-N");

  double R_HR=0.0;
//efficiency of heat recovery from input:
  if (P.Vcodid(Id,"HEAT-REC-SYSTEM")!=1) { //system has heat recovery
    if (P.Vcodid(Id,"HEAT-REC-VAR-EFF")==2) // 2=NO
      R HR=P.Vnum(Id,"HEAT-REC-SEFF");
    else // i.e., variable heat recovery efficency is applicable
      R_HR=0;
  }
  else {
    R_HR=0; //system has no heat recovery
  }


// Modified ID8 - now including heat recovery in cooling calc,
U V COOL and mH VENT COOL are now deques
// Modified ID8c - addition of infiltration in cooling, considering
wind effects only and not stack effects:

 double mOUT ENV AREA=0;
 //calculation of the space outer envelope area:
 for(int i=0;i<vW.size();i++)
   if (P.Vcodid(vW[i].Id,"TYPE")==1)

mOUT ENV AREA+=P.Vnum(vW[i].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER");

 double mU INF WIND ZONE[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 deque<double> mU INF WIND WALL;
// deque<double> mCOOL SET POINT=P.Vlnum(Id,"COOL-SET-POINT");

 //Calculation of the infiltrations due to wind in a whole zone as
the addition of the contribution of each wall
 for(int i=0;i<vW.size();i++) {
//                  mU INF WIND WALL=P.Vlnum Mul(vW[i].Id,"U-INF-WIND-
WALL",P.Vnum(vW[i].Id,"MULTIPLIER")); // m3/hm2 outer envelope
   if   (mOUT_ENV_AREA!=0)   mU_INF_WIND_WALL=P.Vlnum_Mul(vW[i].Id,"U-
INF-WIND-
WALL",P.Vnum(vW[i].Id,"MULTIPLIER")*P.Vnum(vW[i].Id,"AREA")/mOUT ENV
AREA);  // m3/hm2 outer envelope
   else {for(int j=0;j<vW.size();j++) mU INF WIND WALL[j]==0;}
   for(int                                        month=0;month<12;month++)
mU_INF_WIND_ZONE[month]+=mU_INF_WIND_WALL[month];
 }

 for(int month=0;month<12;month++) {
   if(mAREA==0) mU INF WIND ZONE[month]=-1;
   else  mU INF WIND ZONE[month]*=mOUT ENV AREA/mAREA;   //to convert
m3/hm2 outer envelope to m3/hm2 floor area
 }

  double U_V_COOL[12];
//  for(int i=0;i<12;i++) U V COOL[i]=mU INF WIND ZONE[i]+F VENT*((1-
R HR)*mU V M+U V N COOL);
//    for(int  i=0;i<12;i++)  U_V_COOL[i]=mU_INF_WIND_ZONE[i]/3.6+(1-
R HR)*mU V M[i]+U V N COOL[i];
  for(int i=0;i<12;i++) U V COOL[i]=(1-R HR)*mU V M[i]+U V N COOL[i];

  double mH VENT COOL[12];
  double mH INF COOL[12];
  for(int                                                  i=0;i<12;i++)
```

```
mH VENT COOL[i]=D.Value("D014")*mAREA*U V COOL[i];
  for(int                                                    i=0;i<12;i++)
mH INF COOL[i]=D.Value("D014")*mAREA*mU INF WIND ZONE[i]/3.6;
  // The division by 3.6 in the above line is to convert m3/h.m2 to
l/s.m2
// End Modified ID8 - now including heat recovery in cooling calc
//  End  Modified  ID8c  -  addition  of  infiltration  in  cooling,
considering wind effects only and not stack effects:

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mH VENT COOL[i] << ", ";
  res << mH_VENT_COOL[11] << " }";
  P.ModKey(Id,"H-VENT-COOL",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mH_INF_COOL[i] << ", ";
  res << mH INF COOL[11] << " }";
  P.ModKey(Id,"H-INF-COOL",res.str());
 }
}

void  SPACE::fQ VENT HEAT(PROJECTDefaults&  D,  PROJECTTables&  T,  BOC&
P) {
  if(func report) cout << endl << "  " << "SPACE::fQ VENT HEAT";
//calculate ventilation heat transfer for zone for heating

  deque<double> mH VENT HEAT=P.Vlnum(Id,"H-VENT-HEAT");
  deque<double> mH INF HEAT=P.Vlnum(Id,"H-INF-HEAT");
  double  mQ VENT HEAT[12]={0.0,  0.0  ,0.0  ,0.0  ,0.0  ,0.0  ,0.0  ,0.0
,0.0 ,0.0 ,0.0 ,0.0};
  double mQ INF HEAT[12]={0.0, 0.0 ,0.0 ,0.0 ,0.0 ,0.0 ,0.0 ,0.0 ,0.0
,0.0 ,0.0 ,0.0};
  deque<double> mHEAT SET POINT;

  mHEAT_SET_POINT=P.Vlnum(Id,"HEAT-SET-POINT");

  string mWEATHER;
  if(gId<0) mWEATHER="_LON"; else mWEATHER="_"+P.Vcod(gId,"WEATHER");

  deque<double>                mOCC T HOURS=P.Vlnum(Id,"OCC-T-HOURS");
//hours/month
  double mOCC CORR FACTOR[12];

  for (int i=0;i<12;i++) {
    mOCC CORR FACTOR[i]=mOCC T HOURS[i]/(T.Value("NJ",1,i+1)*24);
    if (mHEAT SET POINT[i]!=-9999) {

mQ INF HEAT[i]=mH INF HEAT[i]*D.Value("D008")*T.Value("NJ",1,i+1)*(mH
EAT SET POINT[i]-T.Value("T003"+mWEATHER,1,i+1));

mQ VENT HEAT[i]=mQ INF HEAT[i]+mOCC CORR FACTOR[i]*mH VENT HEAT[i]*D.
Value("D008")*T.Value("NJ",1,i+1)*(mHEAT SET POINT[i]-
T.Value("T003"+mWEATHER,1,i+1));
    }
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ_VENT_HEAT[i] << ", ";
  res << mQ VENT HEAT[11] << " }";
  P.ModKey(Id,"Q-VENT-HEAT",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
```

```
    res << mQ INF HEAT[i] << ", ";
  res << mQ INF HEAT[11] << " }";
  P.ModKey(Id,"Q-INF-HEAT",res.str());
 }
}

void SPACE::fQ VENT COOL(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func report) cout << endl << "  " << "SPACE::fQ VENT COOL";
//calculate ventilation heat transfer for zone for cooling

  deque<double> mH VENT COOL=P.Vlnum(Id,"H-VENT-COOL");
  deque<double> mH INF COOL=P.Vlnum(Id,"H-INF-COOL");

  double mQ VENT COOL[12]={0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };
  double mQ INF COOL[12]={0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };
  deque<double> mCOOL SET POINT;

  mCOOL SET POINT=P.Vlnum(Id,"COOL-SET-POINT");

  // JO 01Jan08. Modification of Set Point for Mixed-mode calculation
in both
  // Actual and Notional building. The correction below (23Sep07) is
for overheating (general)
  // and Mixed-mode in Reference building.
  if(P.Vcod(pId,"MIXED-MODE")=="YES")
    for (int i=0;i<12;i++) {
     if (mCOOL SET POINT[i]!=-9999)
      mCOOL SET POINT[i]=D.Value("DMMTemp");
    }

  // JO 23Sep07. Modification of Set Point for Overheating and Mixed-
mode calculation
  int mCOOL SET POINT MODE=P.Vcodid(gId,"COOL-SET-POINT-MODE");

  switch (mCOOL SET POINT MODE) {
  case 2:
       {
            for (int i=0;i<12;i++) {
             if (mCOOL_SET_POINT[i]!=-9999)
              mCOOL SET POINT[i]=D.Value("DOHTemp");
            }
       } break;
  case 3:
       {
            for (int i=0;i<12;i++) {
             if (mCOOL SET POINT[i]!=-9999)
              mCOOL SET POINT[i]=D.Value("DMMTemp");
            }
       } break;
  }


  deque<double>               mOCC T HOURS=P.Vlnum(Id,"OCC-T-HOURS");
//hours/month
  double mOCC CORR FACTOR[12];

    for (int i=0;i<12;i++) {
      mOCC CORR FACTOR[i]=mOCC T HOURS[i]/(T.Value("NJ",1,i+1)*24);
      if (mCOOL SET POINT[i]!=-9999) {

mQ INF COOL[i]=mH INF COOL[i]*D.Value("D008")*T.Value("NJ",1,i+1)*(mC
OOL SET POINT[i]-T.Value("T025",1,i+1));

mQ VENT COOL[i]=mQ INF COOL[i]+mOCC CORR FACTOR[i]*mH VENT COOL[i]*D.
Value("D008")*T.Value("NJ",1,i+1)*(mCOOL SET POINT[i]-
T.Value("T025",1,i+1));
```

```
      }
    }

  switch (mCOOL SET POINT MODE) {
  case 1:
       {
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mQ VENT COOL[i] << ", ";
          res << mQ_VENT_COOL[11] << " }";
          P.ModKey(Id,"Q-VENT-COOL",res.str());
         }
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mQ_INF_COOL[i] << ", ";
          res << mQ INF COOL[11] << " }";
          P.ModKey(Id,"Q-INF-COOL",res.str());
         }
       } break;
  case 2:
       {
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mQ VENT COOL[i] << ", ";
          res << mQ VENT COOL[11] << " }";
          P.ModKey(Id,"Q-VENT-COOL-OH",res.str());
         }
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mQ INF COOL[i] << ", ";
          res << mQ INF COOL[11] << " }";
          P.ModKey(Id,"Q-INF-COOL-OH",res.str());
         }
       } break;
  case 3:
       {
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mQ VENT COOL[i] << ", ";
          res << mQ_VENT_COOL[11] << " }";
          P.ModKey(Id,"Q-VENT-COOL",res.str());
         }
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mQ_INF_COOL[i] << ", ";
          res << mQ INF COOL[11] << " }";
          P.ModKey(Id,"Q-INF-COOL",res.str());
         }
       } break;
  }


  double mQ VENT COOL NC[12]={0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mQ VENT COOL NC[i] << ", ";
   res << mQ_VENT_COOL_NC[11] << " }";
   P.ModKey(Id,"Q-VENT-COOL-NC",res.str());
   P.ModKey(Id,"H-VENT-COOL-NC",res.str());
  }
```

```
  //calculate ventilation heat transfer for zone for cooling in the
actual building if there is night cooling
  if(P.Vcod(gId,"ACT-NOT")=="ACT") {
    // Check if there is night cooling
    if(P.Vcod(Id,"NIGHT-COOLING")=="YES")
      fQ VENT COOL NC(D, T, P);
  }

}


void  SPACE::fQ VENT COOL NC(PROJECTDefaults&  D,  PROJECTTables&  T,
BOC& P) {
  if(func report) cout << endl << "  " << "SPACE::fQ VENT COOL NC";
  //calculate ventilation heat transfer for zone for cooling if there
is night cooling

  double mH VENT COOL NC[12]={0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };
  double mQ VENT COOL NC[12]={0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };
  double  mOPT FLOW NC[12]={0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,
0.0, 0.0, 0.0, 0.0 };
  double mOPT TO MAX NC[12]={0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };  // ratio of best to max flow

  double mAREA=P.Vnum(Id,"AREA");
  double  mMAX FLOW NC=P.Vnum(Id,"MAX-FLOW-NC");   // maximum air flow
rate for night cooling input by user, in l/s.m2
  mMAX FLOW NC*=mAREA;  // maximum air flow rate for night cooling in
l/s.
  double mFLOW=0.0;
  double  mMAX HOURS NC=P.Vnum(Id,"MAX-HOURS-NC");        //  maximum
number of hours in the month for night cooling input by user, in
hours/month
  double mOCC CORR FACTOR;

  double R_HR=0.0;
  //efficiency of heat recovery from input:
  if (P.Vcodid(Id,"HEAT-REC-SYSTEM")!=1) { //system has heat recovery
    if (P.Vcodid(Id,"HEAT-REC-VAR-EFF")==2) // 2=NO
      R HR=P.Vnum(Id,"HEAT-REC-SEFF");
    else // i.e., variable heat recovery efficency is applicable
      R_HR=0;
  }
  else {
    R_HR=0; //system has no heat recovery
  }

  // set this to zero for now, which might be revised later
//  R HR=0;    // No longer always the case - see above


  deque<double> mH VENT COOL=P.Vlnum(Id,"H-VENT-COOL");
  deque<double> mH INF COOL=P.Vlnum(Id,"H-INF-COOL");
  deque<double> mQ_TR_COOL=P.Vlnum(Id,"Q-TR-COOL");
  deque<double> mQ VENT COOL=P.Vlnum(Id,"Q-VENT-COOL");
  deque<double> mCM=P.Vlnum(Id,"CM");
  double mH_TR=P.Vnum(Id,"H-TR");
  deque<double> mCOOL SET POINT=P.Vlnum(Id,"COOL-SET-POINT");

  // Modification of the cooling set point for mixed-mode cooling
  if(P.Vcod(pId,"MIXED-MODE")=="YES")
    for (int i=0;i<12;i++) {
     if (mCOOL_SET_POINT[i]!=-9999)
      mCOOL SET POINT[i]=D.Value("DMMTemp");
    }
```

```
  string mWEATHER;
  if(gId<0) mWEATHER=" LON"; else mWEATHER=" "+P.Vcod(gId,"WEATHER");
  double mTEMP EXT;  // external termperature during night cooling

  deque<double> mQ_I=P.Vlnum(Id,"Q-I");
  deque<double> mQ SUN=P.Vlnum(Id,"Q-SUN");
  deque<double> mQ SUN NT=P.Vlnum(Id,"Q-SUN-NT");
  deque<double> mQ_TSC=P.Vlnum(Id,"Q-TSC-COOL");

/*calculate k for this zone:
k = 2*a*r*d*0.33*N*height*EFFr/EFFt
where
a - absorption coefficient
r - surface heat loss resistance
d - proportion of radiant energy falling on the occupied area
N - fresh air change rate in air changes /hour (from the activity
database)
EFFr - radiant efficiency
EFFt - seasonal heat generator efficiency
a = 0.9 and r = 0.123
*/
  double k=0.0;
  double a=D.Value("dAbsorbCoeff");
  double r=D.Value("dSurfaceResistance");
  double rad factor1=D.Value("drad factor1"); //2
  double rad factor2=D.Value("drad factor2"); //0.33
  double d=0.0; //d depends on the system type serving zone
  double  Rad Tot Eff=0.0;  //radiant  to  total  efficiency  ratio,
depends on the system type

  d=T.Value("tRadiant Prop",1,P.Vcodid(pId,"TYPE"));  //d  depends  on
the system type serving zone
  Rad Tot Eff=T.Value("tRad Tot Eff",1,P.Vcodid(pId,"TYPE"));
//radiant to total efficiency ratio, depends on the system type

  double N=0.0;
  double height=P.Vnum(Id,"HEIGHT"); //zone height, needed to convert
l/m/m2 to ACH

  if      (Rad Tot Eff==-6666)      Rad Tot Eff=(P.Vnum(pId,"RADIANT-
EFF"))/(P.Vnum(pId,"HEAT-GEN-SEFF"));

  deque<double> lU V MIN = P.Vlnum(Id,"OA-FLOW-PERS"); //fresh air -
lh/month/m2  (We believe it is actually lh/s/m2/month
  if(P.Vcod(Id,"ACT-DBS-ACU")=="YES") {
    for (int i=0;i<12;i++)
      lU V MIN[i]/=24*T.Value("NJ",1,i+1); //l/s/m2 monthly figures
  }
  for (int i=0;i<12;i++) {
    N+=lU V MIN[i];
  }

  N/=12; //average of all monthly figures

  //convert N from l/s/m2 to ACH:
  N*=3.6/height; //ACH

  //calculate k: k = 2*a*r*d*0.33*N*height*EFFr/EFFt
  k = rad factor1*a*r*d*rad factor2*N*height*Rad Tot Eff;

  double mDEM COOL=0.0;     // cooling demand without night cooling
  double mDEM COOL NC=0.0;  // cooling demand with night cooling

  int row;

  for (int i=0;i<12;i++) {
    // Check if this month has night cooling
    if ((int)T.Value("tNIGHT COOL MONTH",1,i+1)==1) {
```

```
      row=P.V1lint(Id,"COOL-MODE",i);
      mTEMP EXT=T.Value("T025nc"+mWEATHER,1,i+1);
      mOCC CORR FACTOR=mMAX HOURS NC/(T.Value("NJ",1,i+1)*24);

      // Calculate cooling demand without night cooling using 0 air
flow
      mFLOW=0.0;
      mDEM COOL=fCALC COOL DEM NC(D,T,P,          R HR,          mFLOW,
mOCC CORR FACTOR, T.Value("NJ",1,i+1), mCOOL SET POINT[i], mTEMP EXT,

mQ_I[i]+mQ_SUN[i]+mQ_SUN_NT[i]+mQ_TSC[i],
mQ TR COOL[i]+mQ VENT COOL[i],
                                    mH TR,               mH INF COOL[i],
mH_VENT_COOL[i], mCM[1], k, row );

      // If the cooling demand without night cooling is 0, then there
is no need for night cooling
      if (mDEM COOL==0.0) {
        mOPT FLOW NC[i]=0.0;
        mOPT_TO_MAX_NC[i]=0.0;
        mH VENT COOL NC[i]=0.0;
        mQ VENT COOL NC[i]=0.0;
      }
      else { //i.e., there is a cooling demand without night cooling
        mFLOW=mMAX FLOW NC;
        mDEM COOL NC=fCALC COOL DEM NC(D,T,P,       R HR,       mFLOW,
mOCC CORR FACTOR, T.Value("NJ",1,i+1), mCOOL SET POINT[i],
                                       mTEMP EXT,
mQ I[i]+mQ SUN[i]+mQ SUN NT[i]+mQ TSC[i],
mQ TR COOL[i]+mQ VENT COOL[i],
                                       mH TR,          mH INF COOL[i],
mH VENT COOL[i], mCM[1], k, row );
        // If there is a cooling demand with night cooling
        if (mDEM COOL NC>0.0) {
          mOPT FLOW NC[i]=mFLOW;
          if (mMAX FLOW NC>0.0)
            mOPT TO MAX NC[i]=mOPT FLOW NC[i]/mMAX FLOW NC;
        } // end if
        else { // i.e., night cooling with this air flow causes the
cooling demand to be <=0.0

          mFLOW=mMAX_FLOW_NC;
          int max iteration=MAX ITERATIONS DO LOOPS;
          do {
               max_iteration--;
               mFLOW/=2.0;
               // Calculate the cooling demand with night cooling
using 1/2 the flow
               mDEM COOL NC=fCALC COOL DEM NC(D,T,P,   R HR,   mFLOW,
mOCC CORR FACTOR, T.Value("NJ",1,i+1),
                                          mCOOL_SET_POINT[i],
mTEMP EXT, mQ I[i]+mQ SUN[i]+mQ SUN NT[i]+mQ TSC[i],

mQ TR COOL[i]+mQ VENT COOL[i],          mH TR,          mH INF COOL[i],
mH VENT COOL[i], mCM[1], k, row );

          } while (mDEM_COOL_NC<=0.0 && max_iteration>0);

          if (mMAX FLOW NC>0.0)
              mOPT_TO_MAX_NC[i]=mOPT_FLOW_NC[i]/mMAX_FLOW_NC;

          mOPT FLOW NC[i]=(mFLOW*mDEM COOL)/(mDEM COOL-mDEM COOL NC);
          if (mMAX_FLOW_NC>0.0)
            mOPT TO MAX NC[i]=mOPT FLOW NC[i]/mMAX FLOW NC;

        } // end else

        mH VENT COOL NC[i]=D.Value("D014")*(1-R HR)*mOPT FLOW NC[i];
        if (mCOOL_SET_POINT[i]!=-9999) {
```

```
          if((mCOOL SET POINT[i]-mTEMP EXT)!=0.0)
            mH VENT COOL NC[i]*=(mCOOL SET POINT[i]-
mTEMP EXT)/(mCOOL SET POINT[i]-T.Value("T025",1,i+1));
          else
            mH_VENT_COOL_NC[i]=0.0;


mQ_VENT_COOL_NC[i]=mOCC_CORR_FACTOR*mH_VENT_COOL_NC[i]*D.Value("D008"
)*T.Value("NJ",1,i+1)*(mCOOL SET POINT[i]-T.Value("T025",1,i+1));
        }

      } // end else
    } // end if
  } // end for

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mH VENT COOL NC[i] << ", ";
   res << mH_VENT_COOL_NC[11] << " }";
   P.ModKey(Id,"H-VENT-COOL-NC",res.str());
  }

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mQ VENT COOL NC[i] << ", ";
   res << mQ VENT COOL NC[11] << " }";
   P.ModKey(Id,"Q-VENT-COOL-NC",res.str());
  }

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mOPT FLOW NC[i]/mAREA << ", ";
   res << mOPT FLOW NC[11]/mAREA << " }";
   P.ModKey(Id,"OPT-FLOW-NC",res.str());       // l/s.m2
  }

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mOPT TO MAX NC[i] << ", ";
   res << mOPT TO MAX NC[11] << " }";
   P.ModKey(Id,"RATIO-OPT-MAX-NC",res.str());
  }

} // end fQ_VENT_COOL_NC


double SPACE::fCALC COOL DEM NC(PROJECTDefaults& D, PROJECTTables& T,
BOC &P, double R HR, double mFLOW, double mOCC CORR FACTOR, double
mNJ,
                                double      mCOOL SET POINT,      double
mTEMP EXT, double mGAINS, double mLOSSES, double mH TR,
                                double mH INF COOL, double mH VENT COOL,
double mCM, double k, int row ) {
  if(func report) cout << endl << "  " << "SPACE::fCALC COOL DEM NC";
  //calculate cooling demand in a month using night cooling given a
particular air flow

  double mH VENT COOL NC=D.Value("D014")*(1-R HR)*mFLOW;

  double mQ VENT COOL NC=0.0;


  if (mCOOL SET POINT!=-9999)

mQ_VENT_COOL_NC=mOCC_CORR_FACTOR*mH_VENT_COOL_NC*D.Value("D008")*mNJ*
```

```
(mCOOL SET POINT-mTEMP EXT);

  double mGAIN LOSS RATIO NC;
  double mR B COOL NC;
  double mQ_LOSS_COOL_NC;

  if ( (mGAINS==0) || ((mLOSSES+mQ VENT COOL NC)<=0) )
    mGAIN_LOSS_RATIO_NC=-1;
  else
    mGAIN LOSS RATIO NC=(mLOSSES+mQ VENT COOL NC)/mGAINS;

  double mTAU NC=0.0;

  if(mH_TR+mH_VENT_COOL+mH_VENT_COOL_NC==0)
    mTAU NC=-1;
  else
    mTAU_NC=(mCM/3.6)/(mH_TR+mH_VENT_COOL+mH_VENT_COOL_NC);

  double a=-1;

  if(mTAU NC!=-1)
    a=T.Value("tTCcool",row,1)+mTAU NC/T.Value("tTCcool",row,2);

  if (mGAIN LOSS RATIO NC==-1 || a==-1)
    mR B COOL NC=1;
  else {
    if (mGAIN LOSS RATIO NC==1)
      mR B COOL NC=a/(a+1);
    else
      mR B COOL NC=(1-pow(mGAIN LOSS RATIO NC,a))/(1-
pow(mGAIN LOSS RATIO NC,a+1));
  }

  //for zones where H-TR and H-VENT-COOL are 0, TAU-COOL was set to -
1 to avoid dividing by zero
  //for these zones, prEN wi14 section 4.2.1.2 equation 31-32 would
lead to mR B COOL NC=1
  if (mTAU NC==-1)
    mR_B_COOL_NC=1.0;

  mQ LOSS COOL NC=mR B COOL NC*(mLOSSES+mQ VENT COOL NC);


  double mQ DEM COOL NC=0.0;

  if (mGAIN LOSS RATIO NC<=D.Value("D013"))
    mQ DEM COOL NC=(mGAINS-mQ LOSS COOL NC)*D.Value("D038");


  mQ DEM COOL NC/=(1+k);

  return (mQ DEM COOL NC);

} // end fCALC_COOL_DEM_NC



void SPACE::fQ I(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "SPACE::fQ I";
//internal heat gain from occupants, equipment and lighting

  double mAREA=P.Vnum(Id,"AREA");
//  double R_EL=D.Value("D018"); //not used. = 1
//   double  F I VENT=T.Value("T052",1,P.Vcodid(Id,  "F-I-VENT-TYPE"));
//not used. F-I-VENT-TYPE is always type 5 => F I VENT=0.5
  double mQ_I_VENT[12];

  for (int i=0;i<12;i++)
    mQ_I_VENT[i]=0.0;
```

```
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ_I_VENT[i] << ", ";
  res << mQ I VENT[11] << " }";
  P.ModKey(Id,"Q-I-VENT",res.str());
 }

  deque<double> mQ I PERS=P.Vlnum(Id,"PERS-HG-SEN");
  deque<double> mQ_I_APP=P.Vlnum(Id,"EQUP-HG-SEN");
  deque<double> mBR LIGHT CHECK;
  mBR LIGHT CHECK=P.Vlnum(Id,"BR-LIGHT-CHECK");

  double mQ I[12];
  deque<double> mQ I LI=P.Vlnum(Id,"Q-I-LI");
  for (int i=0;i<12;i++) {
    if(P.Vcod(Id,"ACT-DBS-ACU")=="YES")

mQ_I[i]=(mQ_I_PERS[i]+mQ_I_APP[i])*mAREA*D.Value("D008ACU")+mQ_I_LI[i
]+mQ I VENT[i];
    else

mQ I[i]=(mQ I PERS[i]+mQ I APP[i])*mAREA*D.Value("D008")*T.Value("NJ"
,1,i+1)+mQ I LI[i]+mQ I VENT[i];
  }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ I[i] << ", ";
  res << mQ I[11] << " }";
  P.ModKey(Id,"Q-I",res.str());
 }
}

void SPACE::fQ SUN(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "SPACE::fQ SUN";
//calculate solar gain, including from solar energy systems

  double mAREA=P.Vnum(Id,"AREA");
  double mAREA_WIN_ORI[9]={0, 0, 0, 0, 0, 0, 0, 0, 0 };

  //calculate solar gain through transparent surfaces (only windows
are considered as transparent in SBEM):
  double mQ SUN T[12]={0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  for (int i=0;i<vW.size();i++)
    for (int j=0;j<vW[i].vW.size();j++) {
      deque<double> mQ SUN T W;
      mQ SUN T W=P.Vlnum Mul(vW[i].vW[j].Id,"Q-SUN-
T",P.Vnum(vW[i].Id,"MULTIPLIER")*P.Vnum(vW[i].vW[j].Id,"MULTIPLIER"))
; // Check multiplier
      for (int k=0;k<12;k++)
        mQ_SUN_T[k]+=mQ_SUN_T_W[k];

      mAREA WIN ORI[P.Vcodid(vW[i].Id,"ORIENTATION")-
1]+=P.Vnum(vW[i].vW[j].Id,"AREA-
G")*P.Vnum(vW[i].vW[j].Id,"MULTIPLIER");
    }

 //write calculated value to keyword for solar gain through
transparent surfaces, Q-SUN-T
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ_SUN_T[i] << ", ";
  res << mQ SUN T[11] << " }";
  P.ModKey(Id,"Q-SUN-T",res.str());
 }
```

```
  // JO, 27 May 09. Additional calculation for Solar Gain Limit
  double mSOLAR GAIN=0;
  for(int i=3;i<9;i++) mSOLAR GAIN+=mQ SUN T[i];
  mSOLAR_GAIN/=3.6;  // kWh
 {
  ostringstream res; res << mSOLAR GAIN;
  P.ModKey(Id,"SOLAR-GAIN",res.str());
 }


  double mQ SUN[12];
  for (int i=0;i<12;i++)
    mQ_SUN[i]=mQ_SUN_T[i];
{
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ SUN[i] << ", ";
  res << mQ SUN[11] << " }";
  P.ModKey(Id,"Q-SUN",res.str());
 }

 // Q SUN NT - solar gain through opaque surfaces

 deque<double> mQ SUN NT, mQ SUN NT W;
 for (int i=0;i<12;i++) mQ SUN NT.push back(0.0);
 for (int i=0;i<vW.size();i++) {
   mQ SUN NT W=P.Vlnum Mul(vW[i].Id,"Q-SUN-
NT",P.Vnum(vW[i].Id,"MULTIPLIER"));
   for (int j=0;j<12;j++) mQ SUN NT[j]+=mQ SUN NT W[j];
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ SUN NT[i] << ", ";
  res << mQ SUN NT[11] << " }";
  P.ModKey(Id,"Q-SUN-NT",res.str());
 }
 }

  // JO, 31 May 09. Additional calculation for Solar Peak Gain
  double hSOL[13]={0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  for(int h=0;h<13;h++)
    for(int i=0;i<9;i++) {
      hSOL[h]+=mAREA WIN ORI[i]*T.Value("tCIBSEA230",i+1,h+1);
    }
  double mPSG=hSOL[0];
  for(int h=1;h<13;h++)
    if(hSOL[h]>mPSG) mPSG=hSOL[h];
 {
  ostringstream res; res << mPSG/mAREA;
  P.ModKey(Id,"AE-PSG",res.str());
 }


}

void SPACE::fQ_TSC(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "SPACE::fQ TSC";
//calculate TSC contributions

  // Update Solar collector's inputs

  deque<double> mQ_TOT, mQ_TOT_DIRECT, mKWH_AUX;
  double mAux;
  double mCoolCtrl;
  double mQ_TSC_HEAT[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  double mQ TSC HEAT DIRECT[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
};
  double mQ_TSC_COOL[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
```

```
  double mTSC KWH AUX[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  double mQ TSCuse[12];
  double mUse;
  int mFan;
  for(int m=0;m<12;m++) mQ_TSCuse[m]=T.Value("tTSC_use",1,m+1);
  if(P.Vcodid(Id,"TSC-SYSTEM")==1) {
    deque<double> mTSC SUPPLY=P.Vlnum(Id,"TSC-SUPPLY");
    deque<int> mTSC=P.Vlrefid(Id,"TSC");
    int nTsc=mTSC.size();
    for(int i=0;i<nTsc;i++) {
      mQ_TOT=P.Vlnum(mTSC[i],"Q-TOT");
      mQ TOT DIRECT=P.Vlnum(mTSC[i],"Q-TOT-DIRECT");
      mKWH AUX=P.Vlnum(mTSC[i],"KWH-AUX");
      mCoolCtrl=T.Value("tTSC_CoolCtrl",1,P.Vcodid(mTSC[i],"CONTROL-
TYPE"));
      double zMul=P.Vnum(Id,"MULTIPLIER");
      if(zMul==0) zMul=1.0;
      double mSUPPLY=P.Vnum(mTSC[i],"ZONE-SUPPLY");
      mFan=P.Vcodid(mTSC[i],"FAN-INDEPENDENT");
      if(mSUPPLY>0)
        for(int m=0;m<12;m++) {
          if(mFan==1) mUse=mQ TSCuse[m]; else mUse=1.0;
          mAux=mUse*mTSC SUPPLY[i]/mSUPPLY/zMul;
          mQ TSC HEAT[m]+=mQ TOT[m]*mAux;
          mQ TSC HEAT DIRECT[m]+=mQ TOT DIRECT[m]*mAux;

mQ TSC COOL[m]+=mCoolCtrl*(mQ TOT[m]+mQ TOT DIRECT[m])*mAux;
          mTSC KWH AUX[m]+=mKWH AUX[m]*mAux;
        }
    }
  }
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mQ TSC HEAT[i] << ", ";
   res << mQ TSC HEAT[11] << " }";
   P.ModKey(Id,"Q-TSC-HEAT",res.str());    // MJ per month
  }
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mQ_TSC_HEAT_DIRECT[i] << ", ";
   res << mQ TSC HEAT DIRECT[11] << " }";
   P.ModKey(Id,"Q-TSC-HEAT-DIRECT",res.str());    // MJ per month
  }
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mQ TSC COOL[i] << ", ";
   res << mQ TSC COOL[11] << " }";
   P.ModKey(Id,"Q-TSC-COOL",res.str());    // MJ per month
  }
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mTSC KWH AUX[i] << ", ";
   res << mTSC_KWH_AUX[11] << " }";
   P.ModKey(Id,"TSC-KWH-AUX",res.str());    // KWH per month
  }

}
void SPACE::fR B HEAT(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "  " << "SPACE::fR_B_HEAT";
//calculate utilisation factor for heat gain
//This is the fraction of the heat from the sun and internal sources
released in the zone
// that results in a decrease of the heat demand for space heating

  double mAREA=P.Vnum(Id,"AREA");
```

```
  // GAIN LOSS RATIO H
  double mGAIN LOSS RATIO H[12];
  deque<double> mQ I=P.Vlnum(Id,"Q-I");
  deque<double> mQ_SUN=P.Vlnum(Id,"Q-SUN");
  deque<double> mQ TSC=P.Vlnum(Id,"Q-TSC-HEAT");
  deque<double> mQ SUN NT=P.Vlnum(Id,"Q-SUN-NT");
  deque<double> mQ_TR_HEAT=P.Vlnum(Id,"Q-TR-HEAT");
  deque<double> mQ VENT HEAT=P.Vlnum(Id,"Q-VENT-HEAT");
  for (int i=0;i<12;i++)
    if (mQ_TR_HEAT[i]+mQ_VENT_HEAT[i]==0) mGAIN_LOSS_RATIO_H[i]=-1;
    else
mGAIN LOSS RATIO H[i]=(mQ I[i]+mQ SUN[i]+mQ SUN NT[i]+mQ TSC[i])/(mQ
TR_HEAT[i]+mQ_VENT_HEAT[i]);
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mGAIN LOSS RATIO H[i] << ", ";
  res << mGAIN LOSS RATIO H[11] << " }";
  P.ModKey(Id,"GAIN-LOSS-RATIO-H",res.str());
 }

  deque<double>  mTAU=P.Vlnum(Id,"TAU-HEAT");   //mTAU  may  be  -1  if
heat loss coeffs are zero (internal zones)

    int row;
    double a[12]={ -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1 };
    for (int i=0;i<12;i++)
      if(mTAU[i]==-1) a[i]=-1;
      else {
        // Check which version of CEN 13790 (2004 or 2008) is applied
for calculating the intermittance
        if (P.Vcodid(gId,"INTERMIT-CALC")==1)   // 1=CEN2004
          row=P.V1lint(Id,"HEAT-MODE",i);
        else // i.e., CEN2008
          row=1;   // always use row 1 in the table for CEN2008 or
until it is decided to change the values in the table so that row 1
and row 2 are identical

a[i]=T.Value("tTCheat",row,1)+mTAU[i]/T.Value("tTCheat",row,2);
      }
//    else a=T.Value("tTCheat",1,1)+mTAU[i]/T.Value("tTCheat",1,2);


  // R_B_HEAT
  double mR B HEAT[12];
//  for (int i=0;i<12;i++) {
//    if (mGAIN_LOSS_RATIO_H[i]==-1 || a[i]==-1) mR_B_HEAT[i]=1;
//    else if (mGAIN LOSS RATIO H[i]==1) mR B HEAT[i]=a[i]/(a[i]+1);
//         else  mR B HEAT[i]=(1-pow(mGAIN LOSS RATIO H[i],a[i]))/(1-
pow(mGAIN_LOSS_RATIO_H[i],a[i]+1));
//  }

  for (int i=0;i<12;i++) {
    if (mGAIN LOSS RATIO H[i]==-1 || a[i]==-1)
      mR B HEAT[i]=1;
    else {
      if (mGAIN LOSS RATIO H[i]==1)
        mR B HEAT[i]=a[i]/(a[i]+1);
      else {
        if (mGAIN LOSS RATIO H[i]<0.0)
          mR B HEAT[i]=1/mGAIN LOSS RATIO H[i];
        else
          mR B HEAT[i]=(1-pow(mGAIN LOSS RATIO H[i],a[i]))/(1-
pow(mGAIN LOSS RATIO H[i],a[i]+1));
      } // end else
    } // end else
  } // end for
```

```
  //for zones where H-TR and H-VENT-HEAT are 0, TAU-HEAT is set to -1
to avoid dividing by zero
  //for these zones, prEN wi14 section 4.2.1.1 equation 31-32 would
lead to mR B HEAT=1
  for (int i=0;i<12;i++) {
    if (mTAU[i]==-1) mR B HEAT[i]=1.0;
  }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mR B HEAT[i] << ", ";
  res << mR B HEAT[11] << " }";
  P.ModKey(Id,"R-B-HEAT",res.str());
 }
}

void SPACE::fR B COOL(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "SPACE::fR B COOL";
//calculate utilisation factor for heat loss
//This is the fraction of the maximum amount of heat to be released
to its surroundings, through
//transmission and ventilation during the period of cooling demand
that results in a decrease of the cooling demand.


  double mGAIN LOSS RATIO C[12];
  deque<double> mQ I=P.Vlnum(Id,"Q-I");
  deque<double> mQ SUN=P.Vlnum(Id,"Q-SUN");
  deque<double> mQ SUN NT=P.Vlnum(Id,"Q-SUN-NT");
  deque<double> mQ TSC=P.Vlnum(Id,"Q-TSC-COOL");
  deque<double> mQ TR COOL;
  deque<double> mQ VENT COOL;
  deque<double> mQ VENT COOL NC;  // night cooling

  // JO 23Sep07. Modification of Set Point for Overheating and Mixed-
mode calculation
  int mCOOL SET POINT MODE=P.Vcodid(gId,"COOL-SET-POINT-MODE");

  switch (mCOOL SET POINT MODE) {
  case 1:
       {
         mQ TR COOL=P.Vlnum(Id,"Q-TR-COOL");
         mQ VENT COOL=P.Vlnum(Id,"Q-VENT-COOL");
         mQ_VENT_COOL_NC=P.Vlnum(Id,"Q-VENT-COOL-NC");
       } break;
  case 2:
       {
         mQ TR COOL=P.Vlnum(Id,"Q-TR-COOL-OH");
         mQ VENT COOL=P.Vlnum(Id,"Q-VENT-COOL-OH");
       } break;
  case 3:
       {
         mQ_TR_COOL=P.Vlnum(Id,"Q-TR-COOL");
         mQ VENT COOL=P.Vlnum(Id,"Q-VENT-COOL");
         mQ VENT COOL NC=P.Vlnum(Id,"Q-VENT-COOL-NC");
       } break;
  }

  for (int i=0;i<12;i++)
//     if (mQ I[i]+mQ SUN[i]+mQ SUN NT[i]==0) mGAIN LOSS RATIO C[i]=-
1;
    if     (     (mQ_I[i]+mQ_SUN[i]+mQ_SUN_NT[i]+mQ_TSC[i]==0)     ||
((mQ TR COOL[i]+mQ VENT COOL[i]+mQ VENT COOL NC[i])<=0) )
      mGAIN LOSS RATIO C[i]=-1;
    else
mGAIN LOSS RATIO C[i]=(mQ TR COOL[i]+mQ VENT COOL[i]+mQ VENT COOL NC[
i])/(mQ I[i]+mQ SUN[i]+mQ SUN NT[i]+mQ TSC[i]);
// modified ID8
```

```
  switch (mCOOL SET POINT MODE) {
  case 1:
       {
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mGAIN_LOSS_RATIO_C[i] << ", ";
          res << mGAIN LOSS RATIO C[11] << " }";
          P.ModKey(Id,"GAIN-LOSS-RATIO-C",res.str());
         }
       } break;
  case 2:
       {
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mGAIN LOSS RATIO C[i] << ", ";
          res << mGAIN LOSS RATIO C[11] << " }";
          P.ModKey(Id,"GAIN-LOSS-RATIO-C-OH",res.str());
         }
       } break;
  case 3:
       {
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mGAIN LOSS RATIO C[i] << ", ";
          res << mGAIN LOSS RATIO C[11] << " }";
          P.ModKey(Id,"GAIN-LOSS-RATIO-C",res.str());
         }
       } break;
  }

  deque<double> mTAU=P.Vlnum(Id,"TAU-COOL");

    int row;
    double a[12]={ -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1 };
    for (int i=0;i<12;i++)
      if(mTAU[i]==-1) a[i]=-1;
      else {
        row=P.V1lint(Id,"COOL-MODE",i);

a[i]=T.Value("tTCcool",row,1)+mTAU[i]/T.Value("tTCcool",row,2);
      }

  double mR B COOL[12];
//  for (int i=0;i<12;i++)
//    if (mGAIN LOSS RATIO C[i]==-1 || a[i]==-1) mR B COOL[i]=1;
//    else if (mGAIN LOSS RATIO C[i]==1) mR B COOL[i]=a[i]/(a[i]+1);
//         else  mR_B_COOL[i]=(1-pow(mGAIN_LOSS_RATIO_C[i],a[i]))/(1-
pow(mGAIN LOSS RATIO C[i],a[i]+1));

  for (int i=0;i<12;i++) {
    if (mGAIN LOSS RATIO C[i]==-1 || a[i]==-1)
      mR B COOL[i]=1;
    else {
      if (mGAIN LOSS RATIO C[i]==1)
        mR B COOL[i]=a[i]/(a[i]+1);
      else {
        if (mGAIN LOSS RATIO C[i]<0.0)
          mR B COOL[i]=1;
        else     mR_B_COOL[i]=(1-pow(mGAIN_LOSS_RATIO_C[i],a[i]))/(1-
pow(mGAIN LOSS RATIO C[i],a[i]+1));
            // Note that the difference between the line above and
the equation in the standard ISO 13790:2008 is due
            // to the fact that the mGAIN LOSS RATIO C in SBEM is
calculated as losses/gains for cooling while in the
            //  standard  ISO  13790:2008,  it  is  calculated  as
```

```
gains/losses (similar to the one in heating)
      } // end else
    } // end else
  } // end for

  //for zones where H-TR and H-VENT-COOL are 0, TAU-COOL is set to -1
to avoid dividing by zero
  //for these zones, prEN wi14 section 4.2.1.2 equation 31-32 would
lead to mR B COOL=1
  for (int i=0;i<12;i++) {
    if (mTAU[i]==-1) mR_B_COOL[i]=1.0;
  }

  switch (mCOOL_SET_POINT_MODE) {
  case 1:
       {
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mR_B_COOL[i] << ", ";
          res << mR B COOL[11] << " }";
          P.ModKey(Id,"R-B-COOL",res.str());
         }

         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << a[i] << ", ";
          res << a[11] << " }";
          P.ModKey(Id,"A-FACTOR-COOL",res.str());
         }
       } break;
  case 2:
       {
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mR B COOL[i] << ", ";
          res << mR_B_COOL[11] << " }";
          P.ModKey(Id,"R-B-COOL-OH",res.str());
         }

         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << a[i] << ", ";
          res << a[11] << " }";
          P.ModKey(Id,"A-FACTOR-COOL-OH",res.str());
         }
       } break;
  case 3:
       {
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mR B COOL[i] << ", ";
          res << mR_B_COOL[11] << " }";
          P.ModKey(Id,"R-B-COOL",res.str());
         }

         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << a[i] << ", ";
          res << a[11] << " }";
          P.ModKey(Id,"A-FACTOR-COOL",res.str());
         }
       } break;
  }
```

```
}

void SPACE::fQ GAIN HEAT(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func report) cout << endl << "  " << "SPACE::fQ GAIN HEAT";
//calculate useful gains from sun and internal sources

  double mQ GAIN HEAT[12];
  deque<double> mQ I=P.Vlnum(Id,"Q-I");
  deque<double> mQ_SUN=P.Vlnum(Id,"Q-SUN");
  deque<double> mQ TSC=P.Vlnum(Id,"Q-TSC-HEAT");
  deque<double> mQ TSC DIRECT=P.Vlnum(Id,"Q-TSC-HEAT-DIRECT");
  deque<double> mQ_SUN_NT=P.Vlnum(Id,"Q-SUN-NT");
  deque<double> mR B HEAT=P.Vlnum(Id,"R-B-HEAT");
// ID8 Solar gain through opaque constructions Q-SUN-NT added to calc
  for (int i=0;i<12;i++)

mQ GAIN HEAT[i]=(mQ I[i]+mQ SUN[i]+mQ SUN NT[i]+mQ TSC[i])*mR B HEAT[
i]+mQ_TSC_DIRECT[i];
// End ID8 Solar gain through opaque constructions Q-SUN-NT added to
calc
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ GAIN HEAT[i] << ", ";
  res << mQ GAIN HEAT[11] << " }";
  P.ModKey(Id,"Q-GAIN-HEAT",res.str());
 }
}

void SPACE::fQ GAIN COOL(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func report) cout << endl << "  " << "SPACE::fQ GAIN COOL";
//calculate gains from sun and internal sources

  double mQ GAIN COOL[12];
  deque<double> mQ I=P.Vlnum(Id,"Q-I");
  deque<double> mQ_SUN=P.Vlnum(Id,"Q-SUN");
  deque<double> mQ SUN NT=P.Vlnum(Id,"Q-SUN-NT");
  deque<double> mQ TSC=P.Vlnum(Id,"Q-TSC-COOL");
  for (int i=0;i<12;i++)
    mQ GAIN COOL[i]=mQ I[i]+mQ SUN[i]+mQ SUN NT[i]+mQ TSC[i];
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ GAIN COOL[i] << ", ";
  res << mQ_GAIN_COOL[11] << " }";
  P.ModKey(Id,"Q-GAIN-COOL",res.str());
 }
}

void SPACE::fQ LOSS COOL(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func report) cout << endl << "  " << "SPACE::fQ LOSS COOL";
//calculate useful losses through transmission and ventilation during
the period
//of cooling demand that results in a decrease of the cooling demand

  double mQ_LOSS_COOL[12];
  deque<double> mQ TR COOL;
  deque<double> mQ VENT COOL;
  deque<double> mQ_VENT_COOL_NC;  // night cooling
  deque<double> mR B COOL;


  // JO 23Sep07. Modification of Set Point for Overheating and Mixed-
mode calculation
  int mCOOL_SET_POINT_MODE=P.Vcodid(gId,"COOL-SET-POINT-MODE");
```

```
  switch (mCOOL SET POINT MODE) {
  case 1:
       {
         mQ_TR_COOL=P.Vlnum(Id,"Q-TR-COOL");
         mQ VENT COOL=P.Vlnum(Id,"Q-VENT-COOL");
         mQ VENT COOL NC=P.Vlnum(Id,"Q-VENT-COOL-NC");
         mR_B_COOL=P.Vlnum(Id,"R-B-COOL");
       } break;
  case 2:
       {
         mQ TR COOL=P.Vlnum(Id,"Q-TR-COOL-OH");
         mQ VENT COOL=P.Vlnum(Id,"Q-VENT-COOL-OH");
         mR_B_COOL=P.Vlnum(Id,"R-B-COOL-OH");
       } break;
  case 3:
       {
         mQ TR COOL=P.Vlnum(Id,"Q-TR-COOL");
         mQ VENT COOL=P.Vlnum(Id,"Q-VENT-COOL");
         mQ_VENT_COOL_NC=P.Vlnum(Id,"Q-VENT-COOL-NC");
         mR B COOL=P.Vlnum(Id,"R-B-COOL");
       } break;
  }

  for (int i=0;i<12;i++)

mQ LOSS COOL[i]=mR B COOL[i]*(mQ TR COOL[i]+mQ VENT COOL[i]+mQ VENT C
OOL NC[i]);

  switch (mCOOL SET POINT MODE) {
  case 1:
       {
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mQ LOSS COOL[i] << ", ";
          res << mQ LOSS COOL[11] << " }";
          P.ModKey(Id,"Q-LOSS-COOL",res.str());
         }
       } break;
  case 2:
       {
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mQ LOSS COOL[i] << ", ";
          res << mQ LOSS COOL[11] << " }";
          P.ModKey(Id,"Q-LOSS-COOL-OH",res.str());
         }
       } break;
  case 3:
       {
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mQ LOSS COOL[i] << ", ";
          res << mQ_LOSS_COOL[11] << " }";
          P.ModKey(Id,"Q-LOSS-COOL",res.str());
         }
       } break;
  }

}

void  SPACE::fQ DEM HEAT ROOM(PROJECTDefaults&  D,  PROJECTTables&  T,
BOC& P) {
  if(func report) cout << endl << "  " << "SPACE::fQ DEM HEAT ROOM";
//calculate space heating demand for zone
```

```
  deque<double> mQ TR HEAT=P.Vlnum(Id,"Q-TR-HEAT");
  deque<double> mQ VENT HEAT=P.Vlnum(Id,"Q-VENT-HEAT");
  deque<double> mQ GAIN HEAT=P.Vlnum(Id,"Q-GAIN-HEAT");
  deque<double> mGAIN LOSS RATIO H=P.Vlnum(Id,"GAIN-LOSS-RATIO-H");
  double mQ DEM HEAT M ROOM[12]={0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0};
  deque<double> mHEAT SET POINT;

  //if there is heating in system
  if (T.Value("tSYS HEAT",1,P.Vcodid(pId,"TYPE"))==1)  {

    mHEAT SET POINT=P.Vlnum(Id,"HEAT-SET-POINT");
    string mWEATHER;
    if(gId<0)                    mWEATHER=" LON";                    else
mWEATHER=" "+P.Vcod(gId,"WEATHER");
    for (int i=0;i<12;i++) {
      if (mGAIN_LOSS_RATIO_H[i]>D.Value("D013") ||
         mHEAT SET POINT[i]<T.Value("T003"+mWEATHER,1,i+1) )
         mQ DEM HEAT M ROOM[i]=0;
      else
         mQ DEM HEAT M ROOM[i]=mQ TR HEAT[i]+mQ VENT HEAT[i]-
mQ GAIN HEAT[i];
     if (mQ DEM HEAT M ROOM[i]<0) mQ DEM HEAT M ROOM[i]=0;
    }

//radiant correction:
//                        double       mK=P.Vnum(pId,"K");       //
mK=T.Value("tK",1,P.Vcodid(pId,"TYPE"));

/*calculate k for this zone:
k = 2*a*r*d*0.33*N*height*EFFr/EFFt
where
a - absorption coefficient
r - surface heat loss resistance
d - proportion of radiant energy falling on the occupied area
N - fresh air change rate in air changes /hour (from the activity
database)
EFFr - radiant efficiency
EFFt - seasonal heat generator efficiency
a = 0.9 and r = 0.123
*/
    double k=0.0;
    double a=D.Value("dAbsorbCoeff");
    double r=D.Value("dSurfaceResistance");
    double rad_factor1=D.Value("drad_factor1"); //2
    double rad_factor2=D.Value("drad_factor2"); //0.33
    double   d=T.Value("tRadiant Prop",1,P.Vcodid(pId,"TYPE"));   //d
depends on the system type serving zone
    double
Rad Tot Eff=T.Value("tRad Tot Eff",1,P.Vcodid(pId,"TYPE")); //radiant
to total efficiency ratio, depends on the system type
    double N=0.0;
    double  mAREA=P.Vnum(Id,"AREA"); //zone area, needed to convert
l/m/m2 to ACH
    double height=P.Vnum(Id,"HEIGHT"); //zone height

    // If this is the 2010 notional building for E&W and this is a
top lit zone
    if    (    (P.Vcod(gId,"ACT-NOT")=="NOT")    &&    (P.Vint(Id,"NOT-
CASE")==11) ) {  // 11=EW_NOT_2010_Top_Lit
      // Divide the radiant generator efficiency by the thermal
generator efficiency

Rad Tot Eff=T.Value("Not Radiant Heat Seff",1,11)/T.Value("Not Radian
t Heat Seff",2,11);
      // Set "d" to correspond to the HVAC system type "Multiburner
radiant heaters" in order to follow the specs
      // in the 2010 NCM Modelling Guide for the notional building in
this case, which is the only case where the
```

```
      // radiant correction is applied to the heating demand of the
notional building (otherwise, k is set to 0 below).
      d=T.Value("tRadiant Prop",1,9);    //  column  9  =  Multiburner
radiant heaters
    }
    else {
      if     (Rad Tot Eff==-6666)     Rad Tot Eff=(P.Vnum(pId,"RADIANT-
EFF"))/(P.Vnum(pId,"HEAT-GEN-SEFF"));
    } // end else

    deque<double> lU_V_MIN = P.Vlnum(Id,"OA-FLOW-PERS");
    if(P.Vcod(Id,"ACT-DBS-ACU")=="YES") {
      for (int i=0;i<12;i++)
        lU_V_MIN[i]/=24*T.Value("NJ",1,i+1); //l/s/m2 monthly figures
    }
    for (int i=0;i<12;i++) {
      N+=lU_V_MIN[i]/12; //average of all monthly figures
    }
    //convert N from l/s/m2 to ACH:
    N*=3.6/height; //ACH

    //calculate k: k = 2*a*r*d*0.33*N*height*EFFr/EFFt
    k = rad factor1*a*r*d*rad factor2*N*height*Rad Tot Eff;

    if (P.Vcod(gId,"ACT-NOT")=="NOT") {
      // Check if the following conditions are NOT fulfilled:
      // This is a top-lit space and the parent HVAC system provides
heating only (i.e., no cooling),
      // the heat source of the parent HVAC system is NOT a heat pump
and is NOT district heating, and its
      // fuel is NOT biogas, biomass, or waste heat
      // 11=EW NOT 2010 Top Lit , 0=no cooling
      //  18=Heat  pump  (electric):  air  source  ,  19=Heat  pump
(electric): ground or water source
      // 12=Heat pump (gas/oil): air source , 13=Heat pump (gas/oil):
ground or water source
      // 15=District Heating ,  3=Biogas, 9=Biomass, 12=Waste heat
      if       (       !(       (P.Vint(Id,"NOT-CASE")==11)       &&
(T.Value("tSYS_COOL",1,P.Vcodid(pId,"TYPE"))==0)
            &&          (P.Vcodid(pId,"HEAT-SOURCE")!=18)          &&
(P.Vcodid(pId,"HEAT-SOURCE")!=19)
            &&          (P.Vcodid(pId,"HEAT-SOURCE")!=12)          &&
(P.Vcodid(pId,"HEAT-SOURCE")!=13)
            &&          (P.Vcodid(pId,"HEAT-SOURCE")!=15)          &&
(P.Vcodid(pId,"FUEL-TYPE")!=3)
            && (P.Vcodid(pId,"FUEL-TYPE")!=9) && (P.Vcodid(pId,"FUEL-
TYPE")!=12) ) )
        k=0; // in this case don't correct for notional building:
    }
    for (int i=0;i<12;i++)
      mQ_DEM_HEAT_M_ROOM[i]/=(1+k);
  }

  // Scottish regulation gives 5% credits to only heated building
  if (P.Vcod(gId,"ACT-NOT")=="ACT") {
      bool S6 RED FLAG=0;
//SERVICE-STRATEGY categories are:
          // 1)Heated and Naturally Ventilated
          // 2)Heated and Mechanically Ventilated
          // 3)Air Conditioned
          // 4)No heating or cooling
          // From 01 Jan 08
          // 5)Mixed-mode and Naturally Ventilated
          // 6)Mixed-mode and Mechanically Ventilated
//Heated and Mechanically Ventilated category includes heating only
systems with any of:
          //- HVAC level ventilation
          //- zone level mechanical supply
          //- zones with extract systems serving more than one space
```

```
      int system=(int)P.Vnum(Id,"SERVICE-STRATEGY");
        if (system==1) S6 RED FLAG=1; //doesn't include zones with no
heating or cooling
      double mHEAT ONLY RED = 1.0;
      if    (((P.Vcodid(cId,"BR-TYPE")==3)    ||    (P.Vcodid(cId,"EPC-
TYPE")==4))&&  S6 RED FLAG  &&  (P.Vcodid(cId,"BR-S6-NV-CREDIT")==1)  )
mHEAT ONLY RED=D.Value("dHEAT ONLY RED");

      for (int i=0;i<12;i++)
        mQ DEM HEAT M ROOM[i]*=mHEAT ONLY RED;
  }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ DEM HEAT M ROOM[i] << ", ";
  res << mQ_DEM_HEAT_M_ROOM[11] << " }";
  P.ModKey(Id,"Q-DEM-HEAT-M-ROOM",res.str());
 }

 double mQ DEM HEAT ROOM=0;
 for (int i=0;i<12;i++) mQ DEM HEAT ROOM+=mQ DEM HEAT M ROOM[i];
 {
  ostringstream res; res << mQ DEM HEAT ROOM;
  P.ModKey(Id,"Q-DEM-HEAT-ROOM",res.str());
 }
}


void SPACE::fHEAT INTERMITTANCE(PROJECTDefaults& D, PROJECTTables& T,
BOC& P) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"SPACE::fHEAT INTERMITTANCE";
  //adjust values of "Q-DEM-HEAT-M-ROOM" and "Q-DEM-HEAT-ROOM" for
intermittency for conditions which were not covered
  //in the function fHEAT QUASI CONTINUOUS, if the 2008 version of
CEN 13790 is applied

  deque<double> a_red;          // dimensionless reduction factor for
intermittent heating
  deque<double> f N H;            // fraction of days in month with
normal heating mode
  deque<double> mSYS T HOURS;

  if (T.Value("tSYS COOL",1,P.Vcodid(pId,"TYPE"))==1)    // if system
has cooling: use sys-t-hours
    mSYS T HOURS=P.Vlnum(Id,"SYS-T-HOURS");
  else                                                    // if system
has no cooling: use sys-heat-t-hours
     mSYS T HOURS=P.Vlnum(Id,"SYS-HEAT-T-HOURS");

  deque<double> mGAIN LOSS RATIO H=P.Vlnum(Id,"GAIN-LOSS-RATIO-H");
  deque<double>    mQ DEM HEAT M ROOM=P.Vlnum(Id,"Q-DEM-HEAT-M-ROOM");
// continuous heating demand monthly

  deque<double>  mHEAT INTERMIT FLAG=P.Vlnum(Id,"HEAT-INTERMIT-FLAG");
// flag to indicate this intermittency calculation is applicable

  double mQ DEM HEAT ROOM=0;
  deque<double> mTAU_HEAT=P.Vlnum(Id,"TAU-HEAT");

  int row;

  for (int i=0;i<12;i++) {
    if (mHEAT INTERMIT FLAG[i]=1) { // if this type of intermittance
calculation is applicable to this month

      // Fraction of the number of hours in the month with a normal
heating set-point (i.e., no reduced setpoint or switch-off)
```

```
      f N H[i]=mSYS T HOURS[i]/(24*T.Value("NJ",1,i+1));
      //row=P.V1lint(Id,"HEAT-MODE",i);
      row=1;  // always use row 1 in the table for CEN2008 (when this
function is called) or until it is decided to change the values in
the table

      if (mTAU HEAT[i]!=0.0 && mTAU HEAT[i]!=-1) {
        // The figure 3 is an empirical correlation factor, bH,red
        a red[i]=1-
(3*(T.Value("tTCheat",row,2)/mTAU HEAT[i])*mGAIN LOSS RATIO H[i]*(1-
f_N_H[i]));

        if (a red[i]<f N H[i])   // set minimum value
          a_red[i]=f_N_H[i];
        else if (a red[i]>1.0)   // set maximum value
                a red[i]=1.0;
      } // end if
      else
        a red[i]=1.0;

      //monthly values:
      mQ DEM HEAT M ROOM[i]*=a red[i];
    } // end if
  } // end for

  // Update annual value
  for (int i=0;i<12;i++)
   mQ DEM HEAT ROOM+=mQ DEM HEAT M ROOM[i];

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mQ DEM HEAT M ROOM[i] << ", ";
   res << mQ DEM HEAT M ROOM[11] << " }";
   P.ModKey(Id,"Q-DEM-HEAT-M-ROOM",res.str());
  }
  {
   ostringstream res; res << mQ DEM HEAT ROOM;
   P.ModKey(Id,"Q-DEM-HEAT-ROOM",res.str());
  }

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << a_red[i] << ", ";
   res << a red[11] << " }";
   P.ModKey(Id,"HEAT-INTERMIT-RED",res.str());   // reduction factor
due to intermittence for heating
  }

} // end fHEAT_INTERMITTANCE


void SPACE::fQ AUX FAN HEATERS(PROJECTDefaults& D, PROJECTTables& T,
BOC& P) {
  if(func report)     cout     <<     endl     <<     "          "     <<
"SPACE::fQ_AUX_FAN_HEATERS";
//calculate auxiliary energy for fanned warm air heater systems
(based on heating demand)

  deque<double> mQ DEM HEAT M ROOM=P.Vlnum(Id,"Q-DEM-HEAT-M-ROOM");
  double mAUX M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
  double mAUX_ENERGY_KWH=P.Vnum(pId,"AUX-ENERGY/KWH");
  double mAREA=P.Vnum(Id,"AREA");

  if (mAREA>0.0)
    for (int i=0;i<12;i++) {
      mAUX M[i]=(mQ DEM HEAT M ROOM[i]*mAUX ENERGY KWH)/mAREA;      //
AN(18/06/09): added division by area so units are kWh/m2 to enable
```

```
the addition of the other elements below
    }

  //add mech vent auxiliary energy
  deque<double> mOA_FLOW_PERS=P.Vlnum(Id,"OA-FLOW-PERS");
  double mSFP=0;
  if (P.Vcod(Id,"Q-V-TYPE")=="Mechanical") {
    //check whether ventilation is at zone or HVAC level:
    if (P.Vcod(Id,"VENT-ZONE-HVAC")=="HVAC") {
      mSFP=P.Vnum(pId,"SFP");
    }
    else if (P.Vcod(Id,"VENT-ZONE-HVAC")=="ZONE") {
      mSFP=P.Vnum(Id,"VENT-SFP");
    }
  }

  for(int i=0;i<12;i++)
    mAUX M[i]+=mOA FLOW PERS[i]*mSFP/1000;

  // add mechanical exhaust auxiliary energy
  deque<double> mSYS T HOURS;
  double mQ V M EXH;
  double mSFP EXH;

  if (T.Value("tSYS COOL",1,P.Vcodid(pId,"TYPE"))==1)
    mSYS T HOURS=P.Vlnum(Id,"SYS-T-HOURS");
  else
    mSYS T HOURS=P.Vlnum(Id,"SYS-HEAT-T-HOURS");

 //if there is mechanical exhaust - add exhaust auxiliary energy to
monthly auxiliary energy
 //use the value of SFP entered by user for exhaust ventilation
 //use the exhaust flow rate entered by user, Q-V-M-EXH
  if (P.Vcodid(Id,"VENT-MECH-EXH")==1) {
    mQ V M EXH=P.Vnum(Id,"Q-V-M-EXH");  //l/s/m2
    mSFP EXH=P.Vnum(Id,"VENT-SFP-EXH"); //W/l/s
    for(int i=0;i<12;i++)
      mAUX M[i]+=mQ V M EXH*mSFP EXH*mSYS T HOURS[i]/1000; //kWh/m2
  }

  double mDesAux;
  if(P.Vcod(gId,"ACT-NOT")=="ACT")
    if (P.Vcod(Id,"DESTRAT-FAN")=="YES") {
      deque<double> mTSC KWH AUX=P.Vlnum(Id,"TSC-KWH-AUX");
      for(int i=0;i<12;i++) {
        mDesAux=D.Value("dDESTRAT AUX")/3.6-mTSC KWH AUX[i];
        if(mDesAux<0.0) mDesAux=0.0;
        mAUX_M[i]+=mDesAux;  //KWH/m2
      }
    }

  double mKWH AUX=0.0;
  double mQ AUX=0.0;

  for(int i=0;i<12;i++)
    mKWH AUX+=mAUX M[i];

  mQ AUX=mKWH AUX*3.6*mAREA;  //MJ
  mKWH AUX*=mAREA;    //KWH

  double mQ AUX M[12]= {0,0,0,0,0,0,0,0,0,0,0,0};
  double mKWH AUX M[12]= {0,0,0,0,0,0,0,0,0,0,0,0};
  for(int i=0;i<12;i++)  {
    mQ AUX M[i]=mAUX M[i]*3.6*mAREA; //MJ
    mKWH AUX M[i]=mAUX M[i]*mAREA;   //KWH
  }

  // Applying the correction for electric power factor is now done on
ZONE calculated energy
```

```
  // rather than CO2 so would now be included in mQ AUX M, mQ AUX,
mKWH AUX M, and mKWH AUX - AN 18/01/08
// If this is the actual building (fuel for auxiliary is always grid-
electricity)
    for(int m=0;m<12;m++) {
      mQ AUX M[m]*=(1-T.Value("tPOWER FACTOR",1,P.Vcodid(gId,"ELEC-
POWER-FACTOR")));
      mKWH_AUX_M[m]*=(1-T.Value("tPOWER_FACTOR",1,P.Vcodid(gId,"ELEC-
POWER-FACTOR")));
    }
    mQ_AUX*=(1-T.Value("tPOWER_FACTOR",1,P.Vcodid(gId,"ELEC-POWER-
FACTOR")));
    mKWH AUX*=(1-T.Value("tPOWER FACTOR",1,P.Vcodid(gId,"ELEC-POWER-
FACTOR")));


 {
  ostringstream res; res << mQ AUX;
  P.ModKey(Id,"Q-AUX",res.str());      //MJ
 }
 {
  ostringstream res; res << mQ AUX/3.6;
  P.ModKey(Id,"KWH-AUX",res.str());   //KWh
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ AUX M[i]/3.6 << ", ";
  res << mQ AUX M[11]/3.6 << " }";
  P.ModKey(Id,"KWH-AUX-M",res.str());  //KWh
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ AUX M[i] << ", ";
  res << mQ AUX M[11] << " }";
  P.ModKey(Id,"Q-AUX-M",res.str());    //MJ
 }

}

void  SPACE::fQ DEM COOL ROOM(PROJECTDefaults&  D,  PROJECTTables&  T,
BOC& P) {
  if(func report) cout << endl << "  " << "SPACE::fQ DEM COOL ROOM";
//calculate comfort cooling demand for zone

  deque<double> mQ GAIN COOL=P.Vlnum(Id,"Q-GAIN-COOL");
  deque<double> mQ_LOSS_COOL;
  deque<double> mGAIN LOSS RATIO C;

  // JO 23Sep07. Modification of Set Point for Overheating and Mixed-
mode calculation
  int mCOOL SET POINT MODE=P.Vcodid(gId,"COOL-SET-POINT-MODE");

  switch (mCOOL SET POINT MODE) {
  case 1:
       {
         mQ LOSS COOL=P.Vlnum(Id,"Q-LOSS-COOL");
         mGAIN LOSS RATIO C=P.Vlnum(Id,"GAIN-LOSS-RATIO-C");
       } break;
  case 2:
       {
         mQ_LOSS_COOL=P.Vlnum(Id,"Q-LOSS-COOL-OH");
         mGAIN LOSS RATIO C=P.Vlnum(Id,"GAIN-LOSS-RATIO-C-OH");
       } break;
  case 3:
       {
         mQ LOSS COOL=P.Vlnum(Id,"Q-LOSS-COOL");
         mGAIN_LOSS_RATIO_C=P.Vlnum(Id,"GAIN-LOSS-RATIO-C");
```

```
        } break;
  }

  double mQ DEM COOL M ROOM[12]={0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0};

//now calculating the cooling demand "Q-DEM-COOL-M-ROOM" when the
system does not have cooling,
//where there is no cooling the primary energy is set to zero in
zone::fQ PRIM COOL
//  if (T.Value("tSYS_COOL",1,P.Vcodid(pId,"TYPE"))==1)  {
    for (int i=0;i<12;i++) {
      if (mGAIN LOSS RATIO C[i]<=D.Value("D013"))
        mQ_DEM_COOL_M_ROOM[i]=(mQ_GAIN_COOL[i]-
mQ LOSS COOL[i])*D.Value("D038");
    }

/*calculate k for this zone:
k = 2*a*r*d*0.33*N*height*EFFr/EFFt
where
a - absorption coefficient
r - surface heat loss resistance
d - proportion of radiant energy falling on the occupied area
N - fresh air change rate in air changes /hour (from the activity
database)
EFFr - radiant efficiency
EFFt - seasonal heat generator efficiency
a = 0.9 and r = 0.123
*/
    double k=0.0;
    double a=D.Value("dAbsorbCoeff");
    double r=D.Value("dSurfaceResistance");
    double rad factor1=D.Value("drad factor1"); //2
    double rad factor2=D.Value("drad factor2"); //0.33
    // !!!!! AN (18/12/08): Modification below introduced so that the
Cooling demand in the Reference building is independent of the actual
building
    // Since the following 2 parameters depend on the HVAC system
type serving the zone, the if condition below ensures that
    // the system type for the Reference building is constant and
independent from the system type used in the actual building
    // The codid of 26 corresponds to "Split or multi-split system"
which provides heating and cooling with natural vent.
    double d=0.0; //d depends on the system type serving zone
    double  Rad_Tot_Eff=0.0;  //radiant  to  total  efficiency  ratio,
depends on the system type
    if (P.Vcod(gId,"NOT-TYPE")=="REF")  {
      d=T.Value("tRadiant Prop",1,26); //d depends on the system type
serving zone
      Rad Tot Eff=T.Value("tRad Tot Eff",1,26);  //radiant  to  total
efficiency ratio, depends on the system type
    }
    else {
      d=T.Value("tRadiant Prop",1,P.Vcodid(pId,"TYPE"));  //d  depends
on the system type serving zone
      Rad Tot Eff=T.Value("tRad Tot Eff",1,P.Vcodid(pId,"TYPE"));
//radiant to total efficiency ratio, depends on the system type
    }

    double N=0.0;
//     double mAREA=P.Vnum(Id,"AREA"); //zone area, needed to convert
l/m/m2 to ACH
    double  height=P.Vnum(Id,"HEIGHT");  //zone  height,  needed  to
convert l/m/m2 to ACH

//following  line  had  incorrect  (non-existent)  keyword,  correction
below
//        if  (Rad Tot Eff==-6666)  Rad Tot Eff=(P.Vnum(pId,"RADIANT-
EFF"))/(P.Vnum(Id,"GEN-SEFF"));
```

```
    if     (Rad Tot Eff==-6666)     Rad Tot Eff=(P.Vnum(pId,"RADIANT-
EFF"))/(P.Vnum(pId,"HEAT-GEN-SEFF"));

    deque<double> lU V MIN = P.Vlnum(Id,"OA-FLOW-PERS"); //fresh air
- lh/month/m2  (We believe it is actually lh/s/m2/month
    if(P.Vcod(Id,"ACT-DBS-ACU")=="YES") {
      for (int i=0;i<12;i++)
        lU_V_MIN[i]/=24*T.Value("NJ",1,i+1); //l/s/m2 monthly figures
    }
    for (int i=0;i<12;i++) {
      N+=lU_V_MIN[i];
    }
    N/=12; //average of all monthly figures

    //convert N from l/s/m2 to ACH:
    N*=3.6/height; //ACH

    //calculate k: k = 2*a*r*d*0.33*N*height*EFFr/EFFt
    k = rad factor1*a*r*d*rad factor2*N*height*Rad Tot Eff;

    // No radiant correction for the notional building:
    if (P.Vcod(gId,"ACT-NOT")=="NOT")
      k=0.0;

//radiant correction:
//  double mK=P.Vnum(pId,"K");
    for (int i=0;i<12;i++)
      mQ DEM COOL M ROOM[i]/=(1+k);
//  }

 double mQ DEM COOL ROOM=0;
 for (int i=0;i<12;i++) mQ DEM COOL ROOM+=mQ DEM COOL M ROOM[i];
 mQ DEM COOL ROOM*=D.Value("D038");

  switch (mCOOL SET POINT MODE) {
  case 1:
       {
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mQ DEM COOL M ROOM[i] << ", ";
          res << mQ_DEM_COOL_M_ROOM[11] << " }";
          P.ModKey(Id,"Q-DEM-COOL-M-ROOM",res.str());
         }
         {
          ostringstream res; res << mQ DEM COOL ROOM;
          P.ModKey(Id,"Q-DEM-COOL-ROOM",res.str());
         }
       } break;
  case 2:
       {
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mQ DEM COOL M ROOM[i] << ", ";
          res << mQ DEM COOL M ROOM[11] << " }";
          P.ModKey(Id,"Q-DEM-COOL-M-ROOM-OH",res.str());
         }
         {
          ostringstream res; res << mQ_DEM_COOL_ROOM;
          P.ModKey(Id,"Q-DEM-COOL-ROOM-OH",res.str());
         }
       } break;
  case 3:
       {
         {
          ostringstream res; res << "{ ";
          for (int i=0;i<11;i++)
            res << mQ_DEM_COOL_M_ROOM[i] << ", ";
```

```
          res << mQ_DEM_COOL_M_ROOM[11] << " }";
          P.ModKey(Id,"Q-DEM-COOL-M-ROOM",res.str());
         }
         {
          ostringstream res; res << mQ_DEM_COOL_ROOM;
          P.ModKey(Id,"Q-DEM-COOL-ROOM",res.str());
         }
       } break;
  }

}

// Added ID8 - for intermittent cooling:
void SPACE::fCOOL INTERMITTANCE(PROJECTDefaults& D, PROJECTTables& T,
BOC& P) {
  if(func report)    cout    <<    endl    <<    "          "    <<
"SPACE::fCOOL_INTERMITTANCE";
  //adjust values of "Q-DEM-COOL-M-ROOM" and "Q-DEM-COOL-ROOM" for
intermittency

  deque<double> a red;          // dimensionless reduction factor for
intermittent cooling
  deque<double> f N C;           // fraction of days in month with
normal cooling mode
  deque<double> mSYS COOL DAYS=P.Vlnum(Id,"SYS-COOL-DAYS");   // days
in the month where normal cooling mode operates

  deque<double> mGAIN LOSS RATIO C;
  deque<double> mQ DEM COOL M ROOM;

  double mQ DEM COOL ROOM=0;
  deque<double> mTAU COOL=P.Vlnum(Id,"TAU-COOL");

  int mCOOL SET POINT MODE=P.Vcodid(gId,"COOL-SET-POINT-MODE");

  switch (mCOOL SET POINT MODE) {
    case 1:
           {
            mGAIN_LOSS_RATIO_C=P.Vlnum(Id,"GAIN-LOSS-RATIO-C");
            mQ DEM COOL M ROOM=P.Vlnum(Id,"Q-DEM-COOL-M-ROOM");
           }
           break;
    case 2:
           {
            mGAIN_LOSS_RATIO_C=P.Vlnum(Id,"GAIN-LOSS-RATIO-C-OH");
            mQ DEM COOL M ROOM=P.Vlnum(Id,"Q-DEM-COOL-M-ROOM-OH");
           }
           break;
    case 3:
           {
            mGAIN_LOSS_RATIO_C=P.Vlnum(Id,"GAIN-LOSS-RATIO-C");
            mQ DEM COOL M ROOM=P.Vlnum(Id,"Q-DEM-COOL-M-ROOM");
           }
           break;
  } // end switch

  int row;

  for (int i=0;i<12;i++) {
    // Due to the diurnal pattern of the weather, and the effect of
the building thermal inertia, an evening/night
    // thermostat set-back or switch-off has, in general, a much
smaller effect on the energy need for cooling than a
    // thermostat set-back or switch-off has on the heating energy
need. This implies that a thermostat set-back or
    // switch-off during evening/night will result in only a small or
no decrease in energy need for cooling, except
    // during very warm months or in the case of high internal heat
gains, in combination with small heat losses.
```

```
    // Therefore, the time fraction for intermittency in the cooling
mode is based on the number of days per month
    // with cooling instead of number of hours per month as for the
heating mode.
    // Fraction of the number of days in the month with a normal
cooling set-point (i.e., no reduced setpoint or switch-off)
    f N C[i]=mSYS COOL DAYS[i]/T.Value("NJ",1,i+1);
    row=P.V1lint(Id,"COOL-MODE",i);

    if    (mTAU COOL[i]!=0.0    &&    mGAIN LOSS RATIO C[i]!=0.0    &&
mTAU_COOL[i]!=-1) {
      // The figure 3 is an empirical correlation factor, bC,red
      // Note that "1/mGAIN LOSS RATIO C" is used in the line below
instead of "mGAIN_LOSS_RATIO_C" because this ratio in
      // SBEM is calculated as losses/gains for cooling while in the
standard ISO 13790:2008, it is calculated as gains/losses
      // (i.e., similar to the one in heating)
      a red[i]=1-
(3*(T.Value("tTCheat",row,2)/mTAU COOL[i])*(1/mGAIN LOSS RATIO C[i])*
(1-f_N_C[i]));

      if (a red[i]<f N C[i])   // set minimum value
        a red[i]=f N C[i];
      else if (a red[i]>1.0)   // set maximum value
             a red[i]=1.0;
    } // end if
    else
      a red[i]=1.0;

    //monthly values:
    mQ DEM COOL M ROOM[i]*=a red[i];
  } // end for

  //annual value:
  for (int i=0;i<12;i++)
    mQ DEM COOL ROOM+=mQ DEM COOL M ROOM[i];

  switch (mCOOL SET POINT MODE) {
    case 1:
           {
            {
             ostringstream res; res << "{ ";
             for (int i=0;i<11;i++)
               res << mQ DEM COOL M ROOM[i] << ", ";
             res << mQ_DEM_COOL_M_ROOM[11] << " }";
             P.ModKey(Id,"Q-DEM-COOL-M-ROOM",res.str());
            }
            {
             ostringstream res; res << mQ DEM COOL ROOM;
             P.ModKey(Id,"Q-DEM-COOL-ROOM",res.str());
            }
           } break;
    case 2:
           {
            {
             ostringstream res; res << "{ ";
             for (int i=0;i<11;i++)
               res << mQ DEM COOL M ROOM[i] << ", ";
             res << mQ DEM COOL M ROOM[11] << " }";
             P.ModKey(Id,"Q-DEM-COOL-M-ROOM-OH",res.str());
            }
            {
             ostringstream res; res << mQ_DEM_COOL_ROOM;
             P.ModKey(Id,"Q-DEM-COOL-ROOM-OH",res.str());
            }
           } break;
    case 3:
           {
            {
```

```
             ostringstream res; res << "{ ";
             for (int i=0;i<11;i++)
               res << mQ DEM COOL M ROOM[i] << ", ";
             res << mQ DEM COOL M ROOM[11] << " }";
             P.ModKey(Id,"Q-DEM-COOL-M-ROOM",res.str());
            }
            {
             ostringstream res; res << mQ_DEM_COOL_ROOM;
             P.ModKey(Id,"Q-DEM-COOL-ROOM",res.str());
            }
           } break;
  } // end switch

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << a_red[i] << ", ";
   res << a red[11] << " }";
   P.ModKey(Id,"COOL-INTERMIT-RED",res.str());   // reduction factor
due to intermittence for cooling
  }

} // end fCOOL INTERMITTANCE


// End Added ID8 - for intermittent cooling



// End Object: SPACE



//: zone.cpp

#include "headers/PROJECTObjects.h"
#define PRINT(STR, VAR) cout << STR " = " << VAR << endl

using namespace std;

void ZONE::TEST() {
// This function is designed to test different types of objects
// Go on developing it
//  cout << "Inicio de Test ZONE" << endl;
//  cout << "Fin de Test ZONE" << endl;
  SPACE::TEST();
}

//  ZONE::Notional(PROJECTDefaults&  D,  PROJECTTables&  T,  BOC  &P,
BOC&N, int i)
void ZONE::Notional(PROJECTDefaults& D, PROJECTTables& T, BOC &P, BOC
&N, int i) {
  if(func report) cout << endl << "  " << "ZONE::Notional";
  char C=34;
  BObject Temp;
  Temp.Ntype=9;
  Temp.Type="ZONE";
  Temp.Name=P.vINP[i].Name;
  KeywordValue KV;

  int Temp Not Case=0;

  if (P.Vnum(i,"ACTIVITY")!=0) {
    {
     ostringstream res; res << P.Vnum(i,"ACTIVITY");
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make pair("ACTIVITY",KV));
    }
    { // Copy the activity name onto the notional building so that it
```

```
appears in its data reflection report
     ostringstream res; res << P.Vtext(i,"ACTIVITY-NAME");
     KV.Reset(); KV.Init(2,res.str());
     Temp.Prop.insert(make pair("ACTIVITY-NAME",KV));
    }
  }
  else {
    {
     ostringstream res; res << P.Vtext(i,"HEAT-SET-POINT-ORI");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make_pair("HEAT-SET-POINT",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"SB-SET-POINT-ORI");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("SB-SET-POINT",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"HEAT-SET-AVE-ORI");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("HEAT-SET-AVE",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"MAX-TIME-REDUCED");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("MAX-TIME-REDUCED",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"MIN-TIME-REDUCED");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("MIN-TIME-REDUCED",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"HEAT-MODE");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("HEAT-MODE",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"COOL-MODE");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("COOL-MODE",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"COOL-SET-POINT");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("COOL-SET-POINT",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"LIGHT-T-START");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make_pair("LIGHT-T-START",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"LIGHT-T-END");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("LIGHT-T-END",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"OCC-T-START");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("OCC-T-START",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"OCC-T-END");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make_pair("OCC-T-END",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"OCC-T-HOURS");
```

```
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("OCC-T-HOURS",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"SYS-T-START");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("SYS-T-START",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"SYS-T-END");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("SYS-T-END",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"SYS-T-HOURS");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make_pair("SYS-T-HOURS",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"SYS-HEAT-T-START");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("SYS-HEAT-T-START",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"SYS-HEAT-T-END");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("SYS-HEAT-T-END",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"SYS-HEAT-T-HOURS");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("SYS-HEAT-T-HOURS",KV));
    }
    {
     ostringstream res; res << P.Vnum(i,"LIGHT-ILLUM");
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make pair("LIGHT-ILLUM",KV));
    }
    {
     ostringstream res; res << P.Vnum(i,"LIGHT-DISP-PW-ORI");
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make_pair("LIGHT-DISP-PW",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"PERS-HG-SEN");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("PERS-HG-SEN",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"PERS-HG-LAT");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("PERS-HG-LAT",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"EQUP-HG-SEN");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make_pair("EQUP-HG-SEN",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"EQUP-HG-LAT");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("EQUP-HG-LAT",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"OA-FLOW-PERS-MIN");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("OA-FLOW-PERS-MIN",KV));
    }
```

```
    {
     ostringstream res; res << P.Vtext(i,"OA-FLOW-PERS-DES");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("OA-FLOW-PERS-DES",KV));
    }
//    {
//     ostringstream res; res << P.Vtext(i,"F-CTRL-VENT-TYPE");
//     KV.Reset(); KV.Init(4,res.str());
//     Temp.Prop.insert(make pair("F-CTRL-VENT-TYPE",KV));
//    }
    {
     ostringstream res; res << "no demand controlled ventilation";
     KV.Reset(); KV.Init(4,res.str());
     Temp.Prop.insert(make_pair("F-CTRL-VENT-DEM",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"DHW");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("DHW",KV));
    }
    {
     ostringstream res; res << P.Vtext(i,"BR-LIGHT-CHECK");
     KV.Reset(); KV.Init(5,res.str());
     Temp.Prop.insert(make pair("BR-LIGHT-CHECK",KV));
    }
    {
     ostringstream res; res << P.Vnum(i,"SERVICE-TYPE");
     KV.Reset(); KV.Init(4,res.str());
     Temp.Prop.insert(make pair("SERVICE-TYPE",KV));
    }
  }

 // Save NOT-CASE in ZONE after modifying it to reflect the impact of
DRIVER2A on column of tables to retrieve values from
 if (P.Vint(gId,"NOT-CASE")==9)  // 9=EW NOT 2010
   {
    Temp Not Case=P.Vint(gId,"NOT-CASE")+P.Vint(i,"DRIVER2A");      //
temporary parameter used below in this function
    ostringstream res; res << Temp_Not_Case;
    KV.Reset(); KV.Init(1,res.str());
    Temp.Prop.insert(make pair("NOT-CASE",KV));
   }
 else
   {
    Temp_Not_Case=P.Vint(gId,"NOT-CASE");     //  temporary  parameter
used below in this function
    ostringstream res; res << Temp Not Case;
    KV.Reset(); KV.Init(1,res.str());
    Temp.Prop.insert(make pair("NOT-CASE",KV));
   }
 {
  ostringstream res; res << P.Vnum(i,"AREA");
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make_pair("AREA",KV));
 }
 {
  ostringstream res; res << P.Vnum(i,"MULTIPLIER");
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("MULTIPLIER",KV));
 }
 {
  ostringstream res; res << P.Vnum(i,"HEIGHT");
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("HEIGHT",KV));
 }
//remove Q-V-M-MAX, use OA-FLOW-PERS instead:
// {
//  ostringstream res; res << P.Vnum(i,"Q-V-M-MAX");
//  KV.Reset(); KV.Init(1,res.str());
```

```
//  Temp.Prop.insert(make pair("Q-V-M-MAX",KV));
// }
//whether you have mechanical supply in the zone, either at zone or
HVAC level:
 {
  ostringstream res; res << P.Vcod(i,"Q-V-TYPE");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make_pair("Q-V-TYPE",KV));
 }
 {
  ostringstream res; res << P.Vcod(i,"Q-V-FACILITY");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("Q-V-FACILITY",KV));
 }
 {
  ostringstream res; res << P.Vnum(i,"VENT-FRACT-ON");
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("VENT-FRACT-ON",KV));
 }
 {
  ostringstream res; res << P.Vcod(i,"Q-PRIM-LI-MET");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("Q-PRIM-LI-MET",KV));
 }

 if(P.Vref(i,"DHW-GENERATOR")!=" ") {
   {
    ostringstream res; res << C << P.Vref(i,"DHW-GENERATOR") << C;
    KV.Reset(); KV.Init(3,res.str());
    Temp.Prop.insert(make pair("DHW-GENERATOR",KV));
   }
   {
    ostringstream res; res << P.Vcod(i,"DHW-SYS-DIST");
    KV.Reset(); KV.Init(4,res.str());
    Temp.Prop.insert(make pair("DHW-SYS-DIST",KV));
   }
//!!!!!! Set the deadleg length in the notional building to zero.
   {
    //ostringstream res; res << P.Vnum(i,"DHW-PIPE-LEN");
    ostringstream res; res << 0.0;
    KV.Reset(); KV.Init(1,res.str());
    Temp.Prop.insert(make_pair("DHW-PIPE-LEN",KV));
   }
 }

  // If this is the aggregate notional and this is an activity where
the notional building should have light dimming
  if  ( (P.Vint(gId,"NOT-CASE")==9)  &&  (P.Vcodid(i,"DIMMING-NOT")==1)
) {   // // 9=EW NOT 2010 ,  1=YES
    {
    ostringstream  res;  res  <<  "BOTH";      //  i.e.,  manual  plus
photoelectric control
    KV.Reset(); KV.Init(4,res.str());
    Temp.Prop.insert(make_pair("LIGHT-CONTROL",KV));
    }
    {
    ostringstream res; res << "PH-DIMMING";
    KV.Reset(); KV.Init(4,res.str());
    Temp.Prop.insert(make pair("LIGHT-PHOTO-TYPE",KV));
    }
    {
    ostringstream res; res << "NO";
    KV.Reset(); KV.Init(4,res.str());
    Temp.Prop.insert(make pair("LIGHT-PH-BACK",KV));
    }


    { // The value of the parasitic power will be modified later to
the lesser of the current value and 3% of the
```

```
      // installed lighting power
     double mAwindow=0;
     ostringstream res;
     for(int i=0;i<vW.size();i++)
       for(int j=0;j<vW[i].vW.size();j++)
         if(P.Vcodid(vW[i].vW[j].Id,"DISP-WIND")==2)   //   Not   for
Display windows
           mAwindow+=P.Vnum(vW[i].vW[j].Id,"AREA");
     if(P.Vint(Id,"DRIVER2A")>0 && mAwindow>0)
       res << D.Value("dPARAS PWR NOT1");
     else
       res << 0.0;
     KV.Reset(); KV.Init(1,res.str());
     Temp.Prop.insert(make_pair("LIGHT-PARAS-PW",KV));
    }

  } // end if
  else
    {
    ostringstream res; res << "MANUAL";
    KV.Reset(); KV.Init(4,res.str());
    Temp.Prop.insert(make pair("LIGHT-CONTROL",KV));
    }
  {
  ostringstream res; res << T.Value("Not Inf",1,Temp Not Case);   //
JO 24 Feb 09
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("Q50-INF",KV));
  }
  {
  ostringstream res; res << "UNKNOWN";
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("LIGHT-CASE",KV));
  }
  deque<double> mBR LIGHT CHECK=P.Vlnum(i,"BR-LIGHT-CHECK");
  if(mBR LIGHT CHECK[0]==1) {
    {
    ostringstream res; res << "NOT2";
    KV.Reset(); KV.Init(4,res.str());
    Temp.Prop.insert(make pair("LIGHT-TYPE",KV));
    }
  }
  else {
    {
    ostringstream res; res << "NOT1";
    KV.Reset(); KV.Init(4,res.str());
    Temp.Prop.insert(make pair("LIGHT-TYPE",KV));
    }
  }

 { // Whether display lighting uses efficiency lamps
  ostringstream               res;              res               <<
T.TValue("Not Eff Disp Light",1,Temp Not Case);
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("LIGHT-DISPLAY-EFF",KV));
 }
 { // display lighting lamp luminous efficacy
  ostringstream               res;              res               <<
T.Value("Not Disp Light Lum Eff",1,Temp Not Case);
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("LAMP-BALLAST-EFF-DL",KV));
 }

  // Based on communication with Aecom on 14/01/2010, if the actual
building has time control for display
  // lighting (whether switching or dimming), then the notional
building will have time switching with the same
  // hours off and fraction off as the actual building. Otherwise,
both the actual and notional buildings will have the
```

```
  // display lighting on all the time as before - AN (15/01/2010)
//     if   (   (P.Vcodid(cId,"EPC-TYPE")==6)   ||   (P.Vcodid(cId,"BR-
TYPE")==5) )  { // for RoI
//    if (P.Vcodid(i,"LIGHT-TIME-SWITCH")!=3) {   // 3=NONE
//      {
//       ostringstream res; res << "SWITCHING";
//       KV.Reset(); KV.Init(4,res.str());
//       Temp.Prop.insert(make_pair("LIGHT-TIME-SWITCH",KV));
//      }
//      {
//       ostringstream res; res << P.Vnum(i,"LIGHT-DISP-HOFF");
//       KV.Reset(); KV.Init(1,res.str());
//       Temp.Prop.insert(make pair("LIGHT-DISP-HOFF",KV));
//      }
//      {
//       ostringstream res; res << P.Vnum(i,"LIGHT-DISP-F");
//       KV.Reset(); KV.Init(1,res.str());
//       Temp.Prop.insert(make pair("LIGHT-DISP-F",KV));
//      }
//    } // end if
//  } // end if

  // !!! The above was modified again according to the new NCM
Modelling Guide so that the 2010 notional building no
  // !!! longer has time control of display lighting.
  { // The notional building has no time switching for the display
lighting
   ostringstream res; res << "NO";
   KV.Reset(); KV.Init(4,res.str());
   Temp.Prop.insert(make pair("LIGHT-TIME-SWITCH",KV));
  }

  // Heat recovery in the notional building zone will depend on the
ventilation type in the actual building zone
  if(P.Vcodid(i,"Q-V-TYPE")==2) { // 2=Mechanical   (whether at zone
level or HVAC level)
   {
    ostringstream              res;              res              <<
T.TValue("Not_Heat_Recovery_Sys",1,Temp_Not_Case);
    KV.Reset(); KV.Init(4,res.str());
    Temp.Prop.insert(make pair("HEAT-REC-SYSTEM",KV));
   }
   {
    ostringstream              res;              res              <<
T.Value("Not_Heat_Recovery_Seff",1,Temp_Not_Case);
    KV.Reset(); KV.Init(1,res.str());
    Temp.Prop.insert(make pair("HEAT-REC-SEFF",KV));
   }
   {
    ostringstream              res;              res              <<
T.TValue("Not_Var_Heat_Recovery",1,Temp_Not_Case);
    KV.Reset(); KV.Init(4,res.str());
    Temp.Prop.insert(make pair("HEAT-REC-VAR-EFF",KV));
   }
  } // end if
  else { //i.e., ventilation is natural in the actual building zone
   {
    ostringstream res; res << "No heat recovery";
    KV.Reset(); KV.Init(4,res.str());
    Temp.Prop.insert(make_pair("HEAT-REC-SYSTEM",KV));
   }
   {
    ostringstream res; res << 0.0;
    KV.Reset(); KV.Init(1,res.str());
    Temp.Prop.insert(make pair("HEAT-REC-SEFF",KV));
   }
   {
    ostringstream res; res << "NO";
    KV.Reset(); KV.Init(4,res.str());
```

```
    Temp.Prop.insert(make pair("HEAT-REC-VAR-EFF",KV));
   }
  } // end else
 {
  ostringstream res; res << P.Vcod(i,"VENT-MECH-EXH");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("VENT-MECH-EXH",KV));
 }
//keyword below specifies whether the extract system serves more than
one room:
 {
  ostringstream res; res << P.Vcod(i,"VENT-MECH-EXH-NCM");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make_pair("VENT-MECH-EXH-NCM",KV));
 }

 if (P.Vcod(Id,"VENT-MECH-EXH")=="YES") {
   //set correct extract SFP for notional building
   double NOT SFP EXH;
   //check which type of exhaust system you have:
   if (P.Vcodid(i,"VENT-MECH-EXH-NCM")==1)
     NOT SFP EXH=T.Value("tNOT SFP",3,Temp Not Case);    //Have    an
extract system serving multiple spaces // JO 24 Feb 09
   else
     NOT SFP EXH=T.Value("tNOT SFP",4,Temp Not Case);      //Have   an
extract system serving a single room  // JO 24 Feb 09
   {
    ostringstream res; res << NOT SFP EXH;
    KV.Reset(); KV.Init(1,res.str());
    Temp.Prop.insert(make pair("VENT-SFP-EXH",KV));
   }
   {
    ostringstream res; res << P.Vnum(i,"Q-V-M-EXH"); //new keyword
added for exhaust capacity in notional building (same as actual) - AN
- 12/03/07
    KV.Reset(); KV.Init(1,res.str());
    Temp.Prop.insert(make pair("Q-V-M-EXH",KV));
   }
 }
 {
  ostringstream res; res << P.Vcod(i,"VENT-ZONE-HVAC");
  KV.Reset(); KV.Init(4,res.str());
  Temp.Prop.insert(make pair("VENT-ZONE-HVAC",KV));
 }

 //set correct supply SFP for notional building
   double NOT SFP=0.0;
   //check whether ventilation is at zone or HVAC level:
   if (P.Vcod(i,"VENT-ZONE-HVAC")=="HVAC") {
     NOT SFP=T.Value("tNOT SFP",1,Temp Not Case);     //HVAC     level
ventilation
   }
   else if ((P.Vcod(i,"VENT-ZONE-HVAC")=="ZONE") && (P.Vcod(i,"Q-V-
TYPE")=="Mechanical")) {
     NOT SFP=T.Value("tNOT SFP",2,Temp Not Case);      //zone   level
mechanical ventilation
   }
   //check for HEPA filters in zone, if zone has these, add 1 to the
supply SFP
   if     ((P.Vcod(i,"HEPA-FILTER")=="YES")     &&     (P.Vcod(i,"Q-V-
TYPE")=="Mechanical"))
     NOT SFP+=1.0;

 {
  ostringstream res; res << NOT SFP;
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("VENT-SFP",KV));
 }
```

```
  // For auxiliary energy calc, JO 01 Jun 09
   double NOT SFP TU=0.0;
   NOT SFP TU=T.Value("tNOT SFP",5,Temp Not Case);

  //check for HEPA filters in zone, if zone has these, add 1 to the
supply SFP
  // Do we keep this for the SFP TU?
//      if   ((P.Vcod(i,"HEPA-FILTER")=="YES")   &&   (P.Vcod(i,"Q-V-
TYPE")=="Mechanical"))
//    NOT SFP TU+=1.0;
{
  ostringstream res; res << NOT SFP TU;
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make_pair("SFP-TU",KV));
 }

// Psi values for thermal bridges of notional building (the values
are zeros except for notional 2010 and Scottish notional)
 {
  ostringstream res; res << T.Value("Not_Psi",1,Temp_Not_Case);
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("JNCT-ROOF-WALL",KV));
 }
 {
  ostringstream res; res << T.Value("Not Psi",2,Temp Not Case);
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("JNCT-WALL-GRND",KV));
 }
 {
  ostringstream res; res << T.Value("Not Psi",3,Temp Not Case);
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("JNCT-WALL-WALL",KV));
 }
 {
  ostringstream res; res << T.Value("Not Psi",4,Temp Not Case);
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("JNCT-WALL-FLR",KV));
 }
 {
  ostringstream res; res << T.Value("Not Psi",5,Temp Not Case);
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make_pair("JNCT-LINTEL",KV));
 }
 {
  ostringstream res; res << T.Value("Not_Psi",6,Temp_Not_Case);
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("JNCT-SILL",KV));
 }
 {
  ostringstream res; res << T.Value("Not Psi",7,Temp Not Case);
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("JNCT-JAMB",KV));
 }

 {
  ostringstream res; res << T.Value("Not Psi MC",1,Temp Not Case);
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("JNCT-ROOF-WALL-MC",KV));
 }
 {
  ostringstream res; res << T.Value("Not Psi MC",2,Temp Not Case);
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make_pair("JNCT-WALL-GRND-MC",KV));
 }
 {
  ostringstream res; res << T.Value("Not_Psi_MC",3,Temp_Not_Case);
  KV.Reset(); KV.Init(1,res.str());
  Temp.Prop.insert(make pair("JNCT-WALL-WALL-MC",KV));
 }
```

```
  {
   ostringstream res; res << T.Value("Not Psi MC",4,Temp Not Case);
   KV.Reset(); KV.Init(1,res.str());
   Temp.Prop.insert(make pair("JNCT-WALL-FLR-MC",KV));
  }
  {
   ostringstream res; res << T.Value("Not Psi MC",5,Temp Not Case);
   KV.Reset(); KV.Init(1,res.str());
   Temp.Prop.insert(make pair("JNCT-LINTEL-MC",KV));
  }
  {
   ostringstream res; res << T.Value("Not Psi MC",6,Temp Not Case);
   KV.Reset(); KV.Init(1,res.str());
   Temp.Prop.insert(make_pair("JNCT-SILL-MC",KV));
  }
  {
   ostringstream res; res << T.Value("Not_Psi_MC",7,Temp_Not_Case);
   KV.Reset(); KV.Init(1,res.str());
   Temp.Prop.insert(make pair("JNCT-JAMB-MC",KV));
  }

   // Set the indicator for the E&W 2010 notional building such that
NO degradation is applied to the psi values, i.e.,
   // the values from the tables are used as they are.
   if(Temp Not Case>=9) {   // if this is the 2010 notional for E, W,
and NI

     { // Save the keyword for non-metal cladding as a deque of 8
values all equal to 1.0
      ostringstream res; res << "{ ";
      for (int i=0;i<7;i++)
        res << 1.0 << ", ";
      res << 1.0 << " }";
      KV.Reset(); KV.Init(5,res.str());
      Temp.Prop.insert(make pair("JNCT-ACCR-DETAIL",KV));
     }

     { // Save the keyword for metal cladding as a deque of 8 values
all equal to 1.0
      ostringstream res; res << "{ ";
      for (int i=0;i<7;i++)
        res << 1.0 << ", ";
      res << 1.0 << " }";
      KV.Reset(); KV.Init(5,res.str());
      Temp.Prop.insert(make_pair("JNCT-ACCR-DETAIL-MC",KV));
     }

   } // end if

// 18 April 2010 JO: Shell and core
    if (P.Vcod(i,"SHELL-ASSUMED")=="YES")
      {
       ostringstream res; res << "YES";
       KV.Reset(); KV.Init(4,res.str());
       Temp.Prop.insert(make pair("SHELL-ASSUMED",KV));
      }


   N.vINP.push back(Temp);

}
//  ZONE::Notional(PROJECTDefaults&  D,  PROJECTTables&  T,  BOC  &P,
BOC&N, int i)

// v4.3.0 Added
void ZONE::Init(PROJECTDefaults& D, PROJECTTables& T, PROJECTTables&
A, BOC& P) {
   if(func report) cout << endl << "   " << "ZONE::Init";
  // Do not attempt to do any calculation here for any object down the
```

```
hierarchy (walls, windows, ...)
 // They are not yet created.

//v2.0.a: Heat recovery systems types have been updated in matrix - if old system types have
//been used, convert these to new types and add a warning in the SBEM log file
  if ((P.Vcodid(Id,"HEAT-REC-SYSTEM")==3) || (P.Vcodid(Id,"HEAT-REC-SYSTEM")==4)) {
  //add warning to log:
    {ostringstream os; os << endl << endl << "WARNING: Heat Recovery system type [" << P.Vcod(Id,"HEAT-REC-SYSTEM") << "] has been chosen for ZONE [" << P.vINP[Id].Name << "]" << endl << "This type is no longer used and has been replaced by type 'Run around coil'" << endl;
       SendLog(os.str()); WriteLog();}
  //change system type:
    {
    ostringstream res; res << "Run around coil";
    P.ModKey(Id,"HEAT-REC-SYSTEM",res.str());
    }
  }

 // !! AN (04/08/2008): The following lines to set the keyword Q-V-TYPE have been moved to here from the function
 // !! SPACE::fVENT ZONE HVAC because it would be too late to set Q-V-TYPE at that stage as the function ZONE::Init which
 // !! occurs before fVENT ZONE HVAC refers to this keyword in order to determine the settings for the keywords
 // !! SERVICE-STRATEGY and OA-FLOW-PERS as seen below. This can result in wrong settings for these keywords in some cases,
 // !! and consequently also a wrong setting for the keyword BUILDING-ENVIRONMENT. Hence, I have moved the lines below to here.
 //if ventilation is at HVAC level, copy keywrds from HVAC to zone
  if (P.Vcodid(Id,"VENT-ZONE-HVAC")==2)
  //note: if you have mech supply at zone level Q-V-TYPE is already set to "Mechanical" in the input file
    {
     ostringstream res; res << "Mechanical";
     P.ModKey(Id,"Q-V-TYPE",res.str());
    }
  //Reference building always have natural ventilation. JO 17 Dec 08
  if (P.Vcodid(gId,"NOT-TYPE")==2)
    {
     ostringstream res; res << "Natural Supply";
     P.ModKey(Id,"Q-V-TYPE",res.str());
    }

 // Load Activity information from the database if it hasn't been defined in the input file
 fCHECK ACTIVITY(D,T,A,P);
 // Correct ...-T-START, ...-T-END in activities with BR-LIGHT-CHECK[9, 10 or 12] different to -1
 double mSTART[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
 double mEND[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
 deque<double> mHOURS;
 deque<double> mSTART ORI;
 deque<double> mEND_ORI;
 double mHOURSDAY;
 // Correction for lighting
 double mBR_LIGHT_CHECK=P.V1lnum(Id,"BR-LIGHT-CHECK",9);
 if (mBR LIGHT CHECK!=-1) {
   mHOURS=P.Vlnum(Id,"SYS-T-HOURS"); // TO BE CORRECTED TO LIGHT-T-HOURS when available
   mSTART ORI=P.Vlnum(Id,"LIGHT-T-START");
   mEND ORI=P.Vlnum(Id,"LIGHT-T-END");
   for (int m=0;m<12;m++) {
       mHOURSDAY=mHOURS[m]/T.Value("NJ",1,m+1)-1; // When correction is done, delete -1
       mSTART[m]=mBR_LIGHT_CHECK;
```

```
        mEND[m]=mBR_LIGHT_CHECK+mHOURSDAY;
        if (mEND[m]>23) {
           mEND[m]=23;
           mSTART[m]=23-mHOURSDAY;
           if (mSTART[m]<0) mSTART[m]=0;
        }
        if ((mEND[m]-mSTART[m])>(mEND_ORI[m]-mSTART_ORI[m])) {
           mSTART[m]=mSTART_ORI[m];
           mEND[m]=mEND_ORI[m];

        }
    }
    {
     ostringstream res; res << "{ ";
     for (int i=0;i<11;i++)
         res << mSTART[i] << ", ";
     res << mSTART[11] << " }";
     P.ModKey(Id,"LIGHT-T-START",res.str());
    }
    {
     ostringstream res; res << "{ ";
     for (int i=0;i<11;i++)
         res << mEND[i] << ", ";
     res << mEND[11] << " }";
     P.ModKey(Id,"LIGHT-T-END",res.str());
    }
  }
  // Correction for occupancy
  mBR_LIGHT_CHECK=P.V1lnum(Id,"BR-LIGHT-CHECK",10);
  if (mBR_LIGHT_CHECK!=-1) {
    mHOURS=P.Vlnum(Id,"SYS-T-HOURS");  //  TO  BE  CORRECTED  TO  OCC-T-
HOURS when available
    mSTART_ORI=P.Vlnum(Id,"OCC-T-START");
    mEND_ORI=P.Vlnum(Id,"OCC-T-END");
    for (int m=0;m<12;m++) {
        mHOURSDAY=mHOURS[m]/T.Value("NJ",1,m+1)-1;  //  When  correction
is done, delete -1
        mSTART[m]=mBR_LIGHT_CHECK;
        mEND[m]=mBR_LIGHT_CHECK+mHOURSDAY;
        if (mEND[m]>23) {
           mEND[m]=23;
           mSTART[m]=23-mHOURSDAY;
           if (mSTART[m]<0) mSTART[m]=0;
        }
        if ((mEND[m]-mSTART[m])>(mEND_ORI[m]-mSTART_ORI[m])) {
           mSTART[m]=mSTART_ORI[m];
           mEND[m]=mEND_ORI[m];
        }
    }
    {
     ostringstream res; res << "{ ";
     for (int i=0;i<11;i++)
         res << mSTART[i] << ", ";
     res << mSTART[11] << " }";
     P.ModKey(Id,"OCC-T-START",res.str());
    }
    {
     ostringstream res; res << "{ ";
     for (int i=0;i<11;i++)
         res << mEND[i] << ", ";
     res << mEND[11] << " }";
     P.ModKey(Id,"OCC-T-END",res.str());
    }
  }
  // Correction for system
  mBR_LIGHT_CHECK=P.V1lnum(Id,"BR-LIGHT-CHECK",11);
  if (mBR_LIGHT_CHECK!=-1) {
    mHOURS=P.Vlnum(Id,"SYS-T-HOURS");
    mSTART_ORI=P.Vlnum(Id,"SYS-T-START");
```

```
   mEND_ORI=P.Vlnum(Id,"SYS-T-END");
   for (int m=0;m<12;m++) {
       mHOURSDAY=mHOURS[m]/T.Value("NJ",1,m+1);
       mSTART[m]=mBR_LIGHT_CHECK;
       mEND[m]=mBR_LIGHT_CHECK+mHOURSDAY;
       if (mEND[m]>23) {
          mEND[m]=23;
          mSTART[m]=23-mHOURSDAY;
          if (mSTART[m]<0) mSTART[m]=0;
       }
       if ((mEND[m]-mSTART[m])>(mEND_ORI[m]-mSTART_ORI[m])) {
          mSTART[m]=mSTART_ORI[m];
          mEND[m]=mEND_ORI[m];
       }
   }
   {
    ostringstream res; res << "{ ";
    for (int i=0;i<11;i++)
        res << mSTART[i] << ", ";
    res << mSTART[11] << " }";
    P.ModKey(Id,"SYS-T-START",res.str());
   }
   {
    ostringstream res; res << "{ ";
    for (int i=0;i<11;i++)
        res << mEND[i] << ", ";
    res << mEND[11] << " }";
    P.ModKey(Id,"SYS-T-END",res.str());
   }
 }

 // Initialize zone lighting calculations: Eext and FD deques to 0.0
for each month
 mLIGHT_T_START=P.Vlnum(Id,"LIGHT-T-START");
 mLIGHT_T_END=P.Vlnum(Id,"LIGHT-T-END");
 mOCC_T_START=P.Vlnum(Id,"OCC-T-START");
 mOCC_T_END=P.Vlnum(Id,"OCC-T-END");
 double mEExt_max=D.Value("LIGHT_EExt_T3max");
 for (int j=0;j<12;j++) {
    mFD=1.0;
    mFO[j]=1.0;
    mPJF[j]=0.0;
    mLIGHT_ENERGY[j]=0.0;
    mLIGHT_GAIN[j]=0.0;
 }

 if(P.Vrefid(Id,"DHW-GENERATOR")!=0) {
    {
     ostringstream res; res << "YES";
     P.ModKey(P.Vrefid(Id,"DHW-GENERATOR"),"ACTIVATED",res.str());
    }
  }
  else {
      {ostringstream os; os << endl << endl << "WARNING: No/wrong HWS
assigned to ZONE [" << P.vINP[Id].Name << "]" << endl;
       SendLog(os.str()); WriteLog();}
  }


 // Initialize OA-FLOW-PERS depending on F-CTRL-VENT-DEM and values
of OA-FLOW-PERS-MIN and OA-FLOW-PERS-DES
 // Now F-CTRL-VENT-DEM is only defined at zone level. Check in the
future if is also added at HVAC level.

 deque<double> mOA_FLOW_PERS;
 double mCONT_COEFF;
 double mFLOW_RATE;

 if(P.Vcod(gId,"ACT-NOT")=="NOT") {
```

```
    // design flow rate applies whether the ventilation is natural or
mechanical
    mOA FLOW PERS=P.Vlnum(Id,"OA-FLOW-PERS-DES");
 } // end if
 else { // i.e., actual building
   // A control coefficient which depends on the air flow control of
demand-controlled ventilation according to the
   // selected option
   mCONT COEFF=T.Value("tDem Cont Vent Coeff",1,P.Vcodid(Id,"F-CTRL-
VENT-DEM"));
   // Calculate the air flow rate as modified by the demand-
controlled ventilation system defined
   for (int i=0;i<12;i++) {
     mFLOW_RATE=mCONT_COEFF*max(P.V1lnum(Id,"OA-FLOW-PERS-
MIN",i),(D.Value("dDEM CONT VENT1")*P.V1lnum(Id,"OA-FLOW-PERS-
DES",i)))+(1-mCONT COEFF)*P.V1lnum(Id,"OA-FLOW-PERS-DES",i);
     mOA_FLOW_PERS.push_back(mFLOW_RATE);
   } // end for
 } // end else

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mOA FLOW PERS[i] << ", ";
  res << mOA FLOW PERS[11] << " }";
  P.ModKey(Id,"OA-FLOW-PERS",res.str());
 }
  //set value of keyword SERVICE-STRATEGY for zone:
  //categories are 1)Heated and Nat. Ventilated, 2)Heated and Mech.
Ventilated, 3)Air Conditioned (heating and cooling), 4)No Heating or
Cooling
  // 01 Jan 08, Now added 5)Mixed-mode and Nat. Ventilated, 6)Mixed-
mode and Mech. Ventilated

  //first divide systems into heating only (set type to 1), or
heating and cooling (type 3)
  int system=T.Vint("tSERVICE STRGY",1,P.Vcodid(pId,"TYPE"));

  //for heating only systems, if you have mechanical supply, set type
to 2
  //note: Q-V-TYPE: can be Natural Supply or Mechanical
  //note: Q-V-TYPE: if you have mech supply at zone level, Q-V-TYPE
is written as "Mechanical" in the input file
  //note: Q-V-TYPE: if ventilation is at HVAC level: Q-V-TYPE has
been set to "Mechanical" in function SPACE::fVENT_ZONE_HVAC
  if(system==1)       system=T.Vint("tVENT STRGY",1,P.Vcodid(Id,"Q-V-
TYPE"));
  //for systems with no heating or cooling set type to 1 - v2.0.a
changed this so that zones have system=4 -> no auxiliary energy in
notional building
//  else if (system==4) system=1;

  //for heating only systems, if you have either mechanical supply
(this should already have been considered above with Q-V-TYPE),
  //OR a local extract system serving more than one room, set type to
2
//    if  (system==1  &&  ((P.Vcod(Id,"Q-V-TYPE")=="Mechanical")  ||
((P.Vcodid(Id,"VENT-MECH-EXH")==1)  &&  (P.Vcodid(Id,"VENT-MECH-EXH-
NCM")==1)))) system=2;
  if (system==1 && ((P.Vcod(Id,"Q-V-TYPE")=="Mechanical"))) system=2;
//      if   ((system==3   &&   P.Vcod(pId,"MIXED-MODE")=="YES")   ||
P.Vcod(gId,"COOL-SET-POINT-MODE")=="MIXED-MODE") system=5;
  if (system==3 && P.Vcod(pId,"MIXED-MODE")=="YES") system=5;
  if (P.Vcodid(cId,"EPC-TYPE")!=6)
    {    if    (P.Vcod(gId,"COOL-SET-POINT-MODE")=="MIXED-MODE"    &&
system!=4) system=5;}
  else
    { if  (P.Vcod(gId,"COOL-SET-POINT-MODE")=="MIXED-MODE")  system=5;
}
```

```
//     if   (system==5   &&   ((P.Vcod(Id,"Q-V-TYPE")=="Mechanical")   ||
((P.Vcodid(Id,"VENT-MECH-EXH")==1)   &&   (P.Vcodid(Id,"VENT-MECH-EXH-
NCM")==1)))) system=6;
  if (system==5 && ((P.Vcod(Id,"Q-V-TYPE")=="Mechanical"))) system=6;

  // Ensuring the ventilation strategy for the Reference building is
always set to be natural
//  if (P.Vcod(gId,"NOT-TYPE")=="REF")
//    system=5;


//SERVICE-STRATEGY categories are:
          // 1)Heated and Naturally Ventilated
          // 2)Heated and Mechanically Ventilated
          // 3)Air Conditioned
          // 4)No heating or cooling
          // From 01 Jan 08
          // 5)Mixed-mode and Naturally Ventilated
          // 6)Mixed-mode and Mechanically Ventilated
//Heated and Mechanically Ventilated category includes heating only
systems with any of:
          //- HVAC level ventilation
          //- zone level mechanical supply
          //- zones with extract systems serving more than one space
  {
    ostringstream res; res << system;
    P.ModKey(Id,"SERVICE-STRATEGY",res.str());
  }

  // Warning for Service Strategy
  if(P.Vcodid(Id,"SERVICE-TYPE")==1       &&       P.Vnum(Id,"SERVICE-
STRATEGY")==4)
    { ostringstream logtext; logtext << endl << "WARNING: ZONE " <<
P.vINP[Id].Name << " is unconditioned but the NCM Activity database
identifies    this    Activity    as    Conditioned"    <<    endl;
SendLog(logtext.str()); }

}  //End of function

void  ZONE::fCHECK ACTIVITY(PROJECTDefaults&  D,  PROJECTTables&  T,
PROJECTTables& A, BOC& P) {
  if(func_report) cout << endl << "  " << "ZONE::fCHECK_ACTIVITY";
 if(P.Vnum(Id,"ACTIVITY")==0.0) {
   //Check there are all the required keywords for activity
   int flag=0;
   if(P.Vtext(Id,"BR-LIGHT-CHECK")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword BR-
LIGHT-CHECK has not been defined in ZONE [" << P.vINP[Id].Name << "]"
<< endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"PERS-HG-SEN")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword PERS-
HG-SEN has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"PERS-HG-LAT")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword PERS-
HG-LAT has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"EQUP-HG-SEN")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword EQUP-
HG-SEN has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
```

```
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"EQUP-HG-LAT")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword EQUP-
HG-LAT has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"HEAT-SET-POINT")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword HEAT-
SET-POINT has not been defined in ZONE [" << P.vINP[Id].Name << "]"
<< endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"SB-SET-POINT")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword SB-
SET-POINT has not been defined in ZONE [" << P.vINP[Id].Name << "]"
<< endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"HEAT-SET-AVE")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword HEAT-
SET-AVE has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"MAX-TIME-REDUCED")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword MAX-
TIME-REDUCED has not been defined in ZONE [" << P.vINP[Id].Name <<
"]" << endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"MIN-TIME-REDUCED")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword MIN-
TIME-REDUCED has not been defined in ZONE [" << P.vINP[Id].Name <<
"]" << endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"COOL-SET-POINT")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword COOL-
SET-POINT has not been defined in ZONE [" << P.vINP[Id].Name << "]"
<< endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"OA-FLOW-PERS-MIN")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword OA-
FLOW-PERS-MIN has not been defined in ZONE [" << P.vINP[Id].Name <<
"]" << endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"OA-FLOW-PERS-DES")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword OA-
FLOW-PERS-DES has not been defined in ZONE [" << P.vINP[Id].Name <<
"]" << endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"DHW")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword DHW
```

```
has not been defined in ZONE [" << P.vINP[Id].Name << "]" << endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"LIGHT-T-START")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword LIGHT-
T-START has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"LIGHT-T-END")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword LIGHT-
T-END has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"OCC-T-START")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword OCC-T-
START has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"OCC-T-END")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword OCC-T-
END has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"LIGHT-ILLUM")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword LIGHT-
ILLUM has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"LIGHT-DISP-PW")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword LIGHT-
DISP-PW has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"SYS-T-START")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword SYS-T-
START has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"SYS-T-END")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword SYS-T-
END has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"SYS-T-HOURS")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword SYS-T-
HOURS has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"SYS-HEAT-T-START")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword SYS-
```

```
HEAT-T-START has not been defined in ZONE [" << P.vINP[Id].Name <<
"]" << endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"SYS-HEAT-T-END")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword SYS-
HEAT-T-END has not been defined in ZONE [" << P.vINP[Id].Name << "]"
<< endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"SYS-HEAT-T-HOURS")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword SYS-
HEAT-T-HOURS has not been defined in ZONE [" << P.vINP[Id].Name <<
"]" << endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"HEAT-MODE")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword HEAT-
MODE has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"COOL-MODE")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword COOL-
MODE has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"OCC-T-HOURS")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword OCC-T-
HOURS has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"SERVICE-TYPE")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword
SERVICE has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"DRIVER2A")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword
DRIVER2A has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"DRIVER2B")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword
DRIVER2B has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"SOLAR-GAIN-LIMIT")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword SOLAR-
GAIN-LIMIT has not been defined in ZONE [" << P.vINP[Id].Name << "]"
<< endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"PEAK-INTERNAL-GAINS")==" ") {
```

```
      {ostringstream os; os << endl << endl << "ERROR: Keyword PEAK-
INTERNAL-GAINS has not been defined in ZONE [" << P.vINP[Id].Name <<
"]" << endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"DIMMING-NOT")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword
DIMMING has not been defined in ZONE [" << P.vINP[Id].Name << "]" <<
endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }
   if(P.Vtext(Id,"SYS-COOL-DAYS")==" ") {
      {ostringstream os; os << endl << endl << "ERROR: Keyword SYS-
COOL-DAYS has not been defined in ZONE [" << P.vINP[Id].Name << "]"
<< endl;
       SendLog(os.str()); WriteLog();}
      assert(1==0);
   }

 }
 else {
   //Take values from database

   // First Check Activities that were removed:
   // In version 1.2.a
   if(P.Vnum(Id,"ACTIVITY")==509) {
      {ostringstream os; os << endl << endl << "WARNING: Outdated
Activity (509->70)has been found and changed in ZONE [" <<
P.vINP[Id].Name << "]" << endl;
       SendLog(os.str()); WriteLog();}
      {
       ostringstream res; res << 70;
       P.ModKey(Id,"ACTIVITY",res.str());
      }
   }
   if(P.Vnum(Id,"ACTIVITY")==510) {
      {ostringstream os; os << endl << endl << "WARNING: Outdated
Activity (510->470)has been found and changed in ZONE [" <<
P.vINP[Id].Name << "]" << endl;
       SendLog(os.str()); WriteLog();}
      {
       ostringstream res; res << 470;
       P.ModKey(Id,"ACTIVITY",res.str());
      }
   }


   int ini;
   int num;
// CONF order
  int cLIGHT ILLUM=1;
  int cLIGHT_DIS=2;
  int cBR LIGHT CHECK=3;
  int cPERS HG SEN=4;
  int cPERS_HG_LAT=5;
  int cEQUP HG SEN=6;
  int cEQUP HG LAT=7;
  int cCOOL_SET_POINT=8;
  int cHEAT SET POINT=9;
  int cHEAT SET AVE=10;
  int cSB_SET_POINT=11;
  int cOA FLOW PERSON=12;
  int cOA FLOW PERSON DES=13;
  int cDHW=14;
  int cMAX TIME REDUCED=15;
  int cMIN TIME REDUCED=16;
  int cOCC_T_START=17;
```

```
  int cOCC T END=18;
  int cLIGHT T START=19;
  int cLIGHT T END=20;
  int cSYS T START=21;
  int cSYS_T_END=22;
  int cSYS T HOURS=23;
  int cSYS HEAT T START=24;
  int cSYS_HEAT_T_END=25;
  int cSYS HEAT T HOURS=26;
  int cHEAT MODE=27;
  int cCOOL_MODE=28;
  int cOCC T HOURS=29;
  int cSERVICE=30;
  int cDRIVER2A=31;
  int cDRIVER2B=32;
  int cSOLAR GAIN CHECK=33;
  int cPEAK_INTERNAL_GAINS=34;
  int cDIMMING=35;
  int cSYS COOL DAYS=36;

   ini=A.Vint("CONF",cBR LIGHT CHECK,1);
num=A.Vint("CONF",cBR LIGHT CHECK,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"BR-LIGHT-CHECK",res.str());
  }
   ini=A.Vint("CONF",cPERS HG SEN,1);
num=A.Vint("CONF",cPERS HG SEN,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"PERS-HG-SEN",res.str());
  }
   ini=A.Vint("CONF",cPERS_HG_LAT,1);
num=A.Vint("CONF",cPERS HG LAT,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"PERS-HG-LAT",res.str());
  }
   ini=A.Vint("CONF",cEQUP_HG_SEN,1);
num=A.Vint("CONF",cEQUP HG SEN,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"EQUP-HG-SEN",res.str());
  }
   ini=A.Vint("CONF",cEQUP_HG_LAT,1);
num=A.Vint("CONF",cEQUP HG LAT,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"EQUP-HG-LAT",res.str());
  }
   ini=A.Vint("CONF",cHEAT_SET_POINT,1);
num=A.Vint("CONF",cHEAT SET POINT,2);
  {
   ostringstream res; res << "{ ";
```

```
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"HEAT-SET-POINT",res.str());
  }
   ini=A.Vint("CONF",cSB SET POINT,1);
num=A.Vint("CONF",cSB SET POINT,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"SB-SET-POINT",res.str());
  }
   ini=A.Vint("CONF",cHEAT SET AVE,1);
num=A.Vint("CONF",cHEAT SET AVE,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"HEAT-SET-AVE",res.str());
  }
   ini=A.Vint("CONF",cMAX TIME REDUCED,1);
num=A.Vint("CONF",cMAX TIME REDUCED,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"MAX-TIME-REDUCED",res.str());
  }
   ini=A.Vint("CONF",cMIN TIME REDUCED,1);
num=A.Vint("CONF",cMIN TIME REDUCED,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"MIN-TIME-REDUCED",res.str());
  }
   ini=A.Vint("CONF",cCOOL_SET_POINT,1);
num=A.Vint("CONF",cCOOL SET POINT,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"COOL-SET-POINT",res.str());
  }
   ini=A.Vint("CONF",cOA_FLOW_PERSON,1);
num=A.Vint("CONF",cOA FLOW PERSON,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"OA-FLOW-PERS-MIN",res.str());
  }
//!!!NEW KEYWORD  --  CHECK THIS:
  ini=A.Vint("CONF",cOA FLOW PERSON DES,1);
num=A.Vint("CONF",cOA FLOW PERSON DES,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"OA-FLOW-PERS-DES",res.str());
  }
```

```
   ini=A.Vint("CONF",cDHW,1);   num=A.Vint("CONF",cDHW,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"DHW",res.str());
  }
   ini=A.Vint("CONF",cLIGHT T START,1);
num=A.Vint("CONF",cLIGHT T START,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"LIGHT-T-START",res.str());
  }
   ini=A.Vint("CONF",cLIGHT T END,1);
num=A.Vint("CONF",cLIGHT T END,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"LIGHT-T-END",res.str());
  }
   ini=A.Vint("CONF",cOCC T START,1);
num=A.Vint("CONF",cOCC T START,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"OCC-T-START",res.str());
  }
   ini=A.Vint("CONF",cOCC T END,1);
num=A.Vint("CONF",cOCC T END,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"OCC-T-END",res.str());
  }
   ini=A.Vint("CONF",cLIGHT_ILLUM,1);
num=A.Vint("CONF",cLIGHT ILLUM,2);
  {
   ostringstream res;
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1);
   P.ModKey(Id,"LIGHT-ILLUM",res.str());
  }
   ini=A.Vint("CONF",cLIGHT DIS,1);
num=A.Vint("CONF",cLIGHT DIS,2);
  {
   ostringstream res;
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1);
   P.ModKey(Id,"LIGHT-DISP-PW",res.str());
  }
   ini=A.Vint("CONF",cSYS T START,1);
num=A.Vint("CONF",cSYS_T_START,2);
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<num-1;i++)
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
   res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
   P.ModKey(Id,"SYS-T-START",res.str());
  }
   ini=A.Vint("CONF",cSYS T END,1);
num=A.Vint("CONF",cSYS_T_END,2);
```

```
   {
    ostringstream res; res << "{ ";
    for (int i=0;i<num-1;i++)
     res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
    P.ModKey(Id,"SYS-T-END",res.str());
   }
    ini=A.Vint("CONF",cSYS_T_HOURS,1);
num=A.Vint("CONF",cSYS T HOURS,2);
   {
    ostringstream res; res << "{ ";
    for (int i=0;i<num-1;i++)
     res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
    P.ModKey(Id,"SYS-T-HOURS",res.str());
   }
    ini=A.Vint("CONF",cSYS_HEAT_T_START,1);
num=A.Vint("CONF",cSYS HEAT T START,2);
   {
    ostringstream res; res << "{ ";
    for (int i=0;i<num-1;i++)
     res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
    P.ModKey(Id,"SYS-HEAT-T-START",res.str());
   }
    ini=A.Vint("CONF",cSYS HEAT T END,1);
num=A.Vint("CONF",cSYS HEAT T END,2);
   {
    ostringstream res; res << "{ ";
    for (int i=0;i<num-1;i++)
     res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
    P.ModKey(Id,"SYS-HEAT-T-END",res.str());
   }
    ini=A.Vint("CONF",cSYS HEAT T HOURS,1);
num=A.Vint("CONF",cSYS HEAT T HOURS,2);
   {
    ostringstream res; res << "{ ";
    for (int i=0;i<num-1;i++)
     res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
    P.ModKey(Id,"SYS-HEAT-T-HOURS",res.str());
   }
    ini=A.Vint("CONF",cHEAT MODE,1);
num=A.Vint("CONF",cHEAT_MODE,2);
   {
    ostringstream res; res << "{ ";
    for (int i=0;i<num-1;i++)
     res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
    P.ModKey(Id,"HEAT-MODE",res.str());
   }
    ini=A.Vint("CONF",cCOOL MODE,1);
num=A.Vint("CONF",cCOOL_MODE,2);
   {
    ostringstream res; res << "{ ";
    for (int i=0;i<num-1;i++)
     res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
    P.ModKey(Id,"COOL-MODE",res.str());
   }
    ini=A.Vint("CONF",cOCC T HOURS,1);
num=A.Vint("CONF",cOCC_T_HOURS,2);
   {
    ostringstream res; res << "{ ";
    for (int i=0;i<num-1;i++)
     res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
    P.ModKey(Id,"OCC-T-HOURS",res.str());
```

```
   }
    ini=A.Vint("CONF",cSERVICE,1);   num=A.Vint("CONF",cSERVICE,2);
   {
    ostringstream res;
    if(A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1)==1)
      res << "CONDITIONED";
    else if(A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1)==2)
      res << "UNCONDITIONED";
    else
      res << "EITHER";
    P.ModKey(Id,"SERVICE-TYPE",res.str());
   }
    ini=A.Vint("CONF",cDRIVER2A,1);   num=A.Vint("CONF",cDRIVER2A,2);
   { // DRIVER2A: 0 = not lit , 1 = side-lit ,  2 = top-lit
    ostringstream               res;               res               <<
A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1);
    P.ModKey(Id,"DRIVER2A",res.str());
   } // DRIVER2B: 0 = Unobstructed spaces , 1 = Spaces with fixed
partitions ,  2 = Spaces with moveable obstructions
    ini=A.Vint("CONF",cDRIVER2B,1);   num=A.Vint("CONF",cDRIVER2B,2);
   {
    ostringstream               res;               res               <<
A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1);
    P.ModKey(Id,"DRIVER2B",res.str());
   }
    ini=A.Vint("CONF",cSOLAR GAIN CHECK,1);
num=A.Vint("CONF",cSOLAR GAIN CHECK,2);
   {
    ostringstream res;
    if(A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1)==1)
      res << "YES";
    else
      res << "NO";
    P.ModKey(Id,"SOLAR-GAIN-CHECK",res.str());
   }
    ini=A.Vint("CONF",cPEAK INTERNAL GAINS,1);
num=A.Vint("CONF",cPEAK INTERNAL GAINS,2);
   {
    double mAREA=P.Vnum(Id,"AREA");
    double mPIG=A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1);
    //if(mAREA>0) mPIG/=mAREA;
    ostringstream res; res << mPIG;
    P.ModKey(Id,"AE-PIG",res.str());
   }
    ini=A.Vint("CONF",cDIMMING,1);   num=A.Vint("CONF",cDIMMING,2);
   {
    ostringstream res;
    if(A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1)==-1)    // -1 =
ticked box = yes
      res << "YES";
    else
      res << "NO";
    P.ModKey(Id,"DIMMING-NOT",res.str());
   }
    ini=A.Vint("CONF",cSYS COOL DAYS,1);
num=A.Vint("CONF",cSYS COOL DAYS,2);
   {
    ostringstream res; res << "{ ";
    for (int i=0;i<num-1;i++)
     res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+i) << ", ";
    res << A.Value("A"+P.Vtext(Id,"ACTIVITY"),1,ini+num-1) << " }";
    P.ModKey(Id,"SYS-COOL-DAYS",res.str());
   }

  }

}

void ZONE::Calc(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
```

```
  if(func report) cout << endl << "   " << "ZONE::Calc";
  if(calc report) cout << endl << P.vINP[Id].Name;

  // Calc Walls first
  for (int i=0;i<vW.size();i++)
    vW[i].Calc(D, T, P);
//  for (int i=0;i<vF.size();i++)
//    vF[i].Calc(D, T, P);
// its own calcs below...
// Added in v1.2.a calculation of external surface area of the zone.
// Modified JO 27 May 09 to get Wall, Roof and Floor areas for Solar
Gain Limit calculation
  double mAREA SUR EXT=0.0; double mWALL=0.0;   double mWALL ALL=0.0;
double mROOF=0.0;  double mFLOOR=0.0;
  double mLIGHT WALL FLOOR AREA=0.0;
  double mAREA=P.Vnum(Id,"AREA");
  double mEXPOSED FACADE WIDTH=0.0;     // total exposed facde width
of the zone

  for (int i=0;i<vW.size();i++) {
    if (P.Vcodid(vW[i].Id,"TYPE")==1) { // 1=exterior

mAREA SUR EXT+=P.Vnum(vW[i].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER");
      switch(P.Vcodid(vW[i].Id,"TYPE-ENV")) {
        case 1: { // wall

mWALL+=P.Vnum(vW[i].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER");

mWALL ALL+=P.Vnum(vW[i].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER");

mEXPOSED FACADE WIDTH+=P.Vnum(vW[i].Id,"LENGTH")*P.Vnum(vW[i].Id,"MUL
TIPLIER");
                } break;
        case                                                        2:
mROOF+=P.Vnum(vW[i].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER"); break;
        // In the line below, the floor area is multiplied by cosine
of the floor pitch angle (in radians) from the
        // horizontal in order to calculate the "footprint" of the
floor area rather than the sloping floor area.
        case                                                        3:
mFLOOR+=P.Vnum(vW[i].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER")*cos((3.
14159265/180.0)*P.Vnum(vW[i].Id,"PITCH")); break;
        default: break;
      } // end switch
    } // end if
    else { // not exterior
      switch(P.Vcodid(vW[i].Id,"TYPE-ENV")) {
        case                                                        1:
mWALL ALL+=P.Vnum(vW[i].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER");
break;
        // In the line below, the floor area is multiplied by cosine
of the floor pitch angle (in radians) from the
        // horizontal in order to calculate the "footprint" of the
floor area rather than the sloping floor area.
        case                                                        3:
mFLOOR+=P.Vnum(vW[i].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER")*cos((3.
14159265/180.0)*P.Vnum(vW[i].Id,"PITCH")); break;
        default: break;
      } // end switch
    } // end else
  } // end for

 {
  ostringstream res; res << mAREA SUR EXT;
  P.ModKey(Id,"AREA-SUR-EXT",res.str());         //includes doors
 }

 if(P.Vcod(gId,"ACT-NOT")=="ACT") {
   {
```

```
    ostringstream res; res << mWALL;
    P.ModKey(Id,"AREA-WALL",res.str());
   }
   {
    ostringstream res; res << mROOF;
    P.ModKey(Id,"AREA-ROOF",res.str());
   }
   {
    ostringstream res; res << mFLOOR;
    P.ModKey(Id,"AREA-FLOOR",res.str());
   }
   {
    ostringstream res; res << mEXPOSED FACADE WIDTH;
    P.ModKey(Id,"EXPOSED-FACADE-WIDTH",res.str());  // in m
   }
 } // end if

 {
  mFLOOR=mAREA;  // JO 17 June 09, We believe mFLOOR should be alwasy
equal to mAREA. If not, there is a weird discrepancy in the
description of the zone.
  if (mFLOOR>0) {
    mLIGHT WALL FLOOR AREA=mWALL ALL/mFLOOR;
    if(mLIGHT WALL FLOOR AREA>D.Value("LIGHT WALL FLOOR AREA LIMIT"))
      mLIGHT WALL FLOOR AREA=D.Value("LIGHT WALL FLOOR AREA LIMIT");
  }
  else mLIGHT WALL FLOOR AREA=-5555;
  ostringstream res; res << mLIGHT WALL FLOOR AREA;
  P.ModKey(Id,"LIGHT-WALL/FLOOR-AREA",res.str());          //includes
doors
 }

// function below sets values for NOT-EFF-MECH-CORR
//  in order to use correct table value for NOTIONAL building HEAT-
SSEEF
//  fNOT EFF MECH CORR(D, T, P);
  //for actual building only: calculate heat transfer coefficient for
thermal bridges by multiplying Length of juction by psi value for
that type of junction
  //for notional building only: heat transfer coefficient = 0
  //NOTE: notional thermal bridge heat loss is instead accounted for
by    use    of    'alpha'    value    multiplying    u-values    in
CONSTRUCTION::Notional()
  //NOTE: In Scottish Building Regs, they still use Psi values.
Modified to account for that: JO_05Apr07
  if (P.Vcod(gId,"ACT-NOT")=="ACT")
    fH TR TM(D, T, P);
  else // i.e., notional building
    if(P.Vcodid(cId,"BR-TYPE")==3   ||   P.Vcodid(cId,"EPC-TYPE")==4)
//Scottish
        fH_TR_TM_NOT_S6(D, T, P);
    else {
      if (P.Vint(Id,"NOT-CASE")>=9)  // 9=EW NOT 2010
        fH_TR_TM(D, T, P);
      else
        fH TR TM NOT(D, T, P);
    } // end else

  //calculate heat transfer coefficient for zone:
  fH_TR(D, P);
  //sums up (cm * net area) for all walls in zone. Units: kJ/m2K *
m2= kJ/K
  fCM(D, P);
  //calculate ventilation heat transfer coefficient for heating:
  fH VENT HEAT(D, T, P);
  //calculatate time constant for heating TAU-HEAT, in hours
  fTAU HEAT(D, P);       //must be called after fH VENT HEAT
  //modifies heating set point to account for intermittent heating
(nights, weekends and holidays)
```

```
  fHEAT QUASI CONTINUOUS(D, T, P); //call after fTAU HEAT
  //calculate ventilation heat transfer coefficient for cooling:
  fH VENT COOL(D, T, P);
  //calculate time constant for cooling TAU-COOL, in hours - moved
further down so that it would include night cooling if available
  //calculate transmission heat transfer for zone for heating:
  fQ TR HEAT(D, T, P);
  //calculate transmission heat transfer for zone for cooling:
  fQ TR COOL(D, T, P);
  //calculate ventilation heat transfer for zone for heating:
  fQ_VENT_HEAT(D, T, P);
  //calculate lighting energy use for the zone
  fLIGHT ENERGY(D, T, P);
  //calculate internal heat gain from occupants, equipment and
lighting
  fQ I(D, T, P);
  //calculate solar gain, including from solar energy systems
  fQ SUN(D, T, P);
  //calculate TSC contributions
  fQ_TSC(D, T, P);
  //calculate ventilation heat transfer for zone for cooling
  fQ VENT COOL(D, T, P);
  //calculate auxiliary energy use for the zone
  fQ AUX(D,T,P);
  //  Functions: fLIGHT ENERGY, fQ I, fQ SUN, and fQ TSC have been
moved up so the gains would be available for the night cooling
calculation
  //calculate utilisation factor for heat gain
  fR B HEAT(D, T, P);
  //calculate time constant for cooling TAU-COOL, in hours
  fTAU COOL(D, P);     //must be called after fH VENT COOL
  //calculate utilisation factor for heat loss
  fR B COOL(D, T, P);
  //calculate useful gains from sun and internal sources
  fQ GAIN HEAT(D, T, P);
  //calculate gains from sun and internal sources
  fQ GAIN COOL(D, T, P);
  //calculate useful losses through transmission and ventilation
during the period
  //of cooling demand that results in a decrease of the cooling
demand
  fQ_LOSS_COOL(D, T, P);
  //calculate space heating demand for zone
  fQ DEM HEAT ROOM(D, T, P);
  // Check which version of CEN 13790 (2004 or 2008) is applied for
calculating the intermittance
  if (P.Vcodid(gId,"INTERMIT-CALC")==2)  // 2=CEN2008
    // calculate the impact of intermittance for conditions which
were not covered in the function fHEAT QUASI CONTINUOUS
    fHEAT INTERMITTANCE(D, T, P);
  //calculate comfort cooling demand for zone
  fQ DEM COOL ROOM(D, T, P);
  // 04Oct07 JO: v2.0.d dev R13 Intermittance cooling suppressed due
to Bug found in CEN Standard. Claire is investigating this.
  // Check which version of CEN 13790 (2004 or 2008) is applied for
calculating the intermittance
  if (P.Vcodid(gId,"INTERMIT-CALC")==2)  // 2=CEN2008
    fCOOL INTERMITTANCE(D, T, P);
  //calculate energy required for Domestic Hot Water heating for zone
  fQ_DHW(D, T, P);


  // JO 31 May 09 New funcs for AE calculation
  fAUX ENERGY(D, T, P);            // This will complement previous
calculation on Q-AUX

}
void ZONE::fSolarGainCheck(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
```

```
  if(func report) cout << endl << "   " << "ZONE::fSolarGainCheck";
  if(calc report) cout << endl << P.vINP[Id].Name;

  //If limit is reached, set compliance keyword SOLAR-GAIN-OVER-LIMIT
to YES
  string mWEATHER;
  if(gId<0) mWEATHER=" LON"; else mWEATHER=" "+P.Vcod(gId,"WEATHER");

  double mSOLAR GAIN LIMIT WALLS=0.0;
  double mSOLAR GAIN LIMIT ROOFS=0.0;
  double mSOLAR_GAIN_LIMIT_FLOORS=0.0;
  double mGreater Area=0.0;

  if(P.Vint(Id,"DRIVER2A")<2) { // 1=side-lit , 0=not lit
    // if the exposed facade area is >0
    if((P.Vnum(Id,"AREA-WALL"))>0.0)
      // Column 1 of TSOL tables contains the vertical solar flux in
kWh/m2 on the east facing facade summed from Apr-Sep
      // multiplied by g-value of 0.68 and 0.9 glazing (10% frame).
In the line below, it is then multiplied by the
      // total exposed facade width of the zone and the height of 1
m, in order to calculate the reference vertical
      // solar flux.

mSOLAR GAIN LIMIT WALLS=T.Value("TSOL"+mWEATHER,1,1)*P.Vnum(Id,"EXPOS
ED-FACADE-WIDTH")*D.Value("dSOL WIN HEIGHT");
    else {// i.e., exposed facade area is = 0
      // Determine the greater of the total exposed roof area and the
footprint floor area
      mGreater Area=P.Vnum(Id,"AREA-ROOF");
      if(P.Vnum(Id,"AREA-FLOOR")>mGreater Area)
        mGreater Area=P.Vnum(Id,"AREA-FLOOR");
      // Column 2 of TSOL tables contains the horizontal solar flux
in kWh/m2 summed from Apr-Sep multiplied by g-value
      // of 0.68 and 0.75 for glazing (25% frame). In the line below,
it is then multiplied by the greater of the
      // total exposed roof area and the footprint floor area of the
zone and 10%, in order to calculate the reference
      // horizontal solar flux.

mSOLAR GAIN LIMIT ROOFS=T.Value("TSOL"+mWEATHER,1,2)*mGreater Area*D.
Value("dSOL_ROOF_AREA1");
    } // end else
  } // end if
  else if (P.Vint(Id,"DRIVER2A")==2) { // 2=top-lit
        // Determine the greater of the total exposed roof area and
the footprint floor area
        mGreater_Area=P.Vnum(Id,"AREA-ROOF");
        if(P.Vnum(Id,"AREA-FLOOR")>mGreater Area)
          mGreater Area=P.Vnum(Id,"AREA-FLOOR");
        // If the zone height is < 6m
        if((P.Vnum(Id,"HEIGHT"))<D.Value("dSOL ZONE HEIGHT"))
          // Column 2 of TSOL tables contains the horizontal solar
flux in kWh/m2 summed from Apr-Sep multiplied by
          // g-value of 0.68 and 0.75 for glazing (25% frame). In
the line below, it is then multiplied by the greater
          // of the total exposed roof area and the footprint floor
area of the zone and 10%, in order to calculate
          // the reference horizontal solar flux.

mSOLAR GAIN LIMIT ROOFS=T.Value("TSOL"+mWEATHER,1,2)*mGreater Area*D.
Value("dSOL ROOF AREA1");
        else // i.e., the zone height is >= 6m
          // Column 3 of TSOL tables contains the horizontal solar
flux in kWh/m2 summed from Apr-Sep multiplied by
          // g-value of 0.46 and 0.85 for glazing (15% frame). In
the line below, it is then multiplied by the the greater
          // of the total exposed roof area and the footprint floor
area of the zone and 20%, in order to calculate
```

```
           // the reference horizontal solar flux.

mSOLAR GAIN LIMIT ROOFS=T.Value("TSOL"+mWEATHER,1,3)*mGreater Area*D.
Value("dSOL ROOF AREA2");
       } // end if

  double mSOLAR GAIN LIMIT=mSOLAR GAIN LIMIT WALLS;
  if(mSOLAR_GAIN_LIMIT_ROOFS>mSOLAR_GAIN_LIMIT)
mSOLAR GAIN LIMIT=mSOLAR GAIN LIMIT ROOFS;
  if(mSOLAR GAIN LIMIT FLOORS>mSOLAR GAIN LIMIT)
mSOLAR_GAIN_LIMIT=mSOLAR_GAIN_LIMIT_FLOORS;

  double mSOLAR GAIN LIMIT OVER=0.0;

 {
  ostringstream res;

  if(P.Vcodid(Id,"SOLAR-GAIN-CHECK")==1)
//    if(P.Vnum(Id,"SOLAR-GAIN")>mSOLAR GAIN LIMIT) {
    if(redondea(P.Vnum(Id,"SOLAR-
GAIN"),2)>redondea(mSOLAR GAIN LIMIT,2)) {
      res << "YES"; P.ModKey(cId,"SOLAR-GAIN-OVER-LIMIT",res.str());
      if(mSOLAR GAIN LIMIT>0)
        mSOLAR GAIN LIMIT OVER=redondea((P.Vnum(Id,"SOLAR-GAIN")-
mSOLAR GAIN LIMIT)/mSOLAR GAIN LIMIT*100,1);
    }
    else {
      res << "NO";
      if(mSOLAR GAIN LIMIT>0)
        mSOLAR GAIN LIMIT OVER=redondea((P.Vnum(Id,"SOLAR-GAIN")-
mSOLAR GAIN LIMIT)/mSOLAR GAIN LIMIT*100,1);
    }
  else
    res << "NO";
  P.ModKey(Id,"SOLAR-GAIN-OVER-LIMIT",res.str());
 }

 {
  ostringstream res; res << "{ ";
  res << mSOLAR GAIN LIMIT WALLS << ", ";
  res << mSOLAR GAIN LIMIT ROOFS << ", ";
  res << mSOLAR_GAIN_LIMIT_FLOORS << ", ";
  res << mSOLAR GAIN LIMIT << ", ";
  res << mSOLAR GAIN LIMIT OVER << " }";
  P.ModKey(Id,"SOLAR-GAIN-LIMITS",res.str());
 }

}

void ZONE::fQ DHW(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "  " << "ZONE::fQ_DHW";
//calculate energy required for Domestic Hot Water heating for zone

  deque<double> mQ_DHW;

  string mDHW GENERATOR=P.Vref(Id,"DHW-GENERATOR");        // This is
not used
//  if (mDHW GENERATOR==" ")
//    mQ DHW=P.Vlnum Mul(Id,"DHW",0.0);
//  else

mQ DHW=P.Vlnum Mul(Id,"DHW",4.18/1000*P.Vnum(Id,"AREA")*(T.Value("tDH
W_TEMP",1,1)-T.Value("tDHW_TEMP",1,2)));    // MJ per month

// Supersided in version 1.1.a
//  for(int m=0;m<12;m++)
//    mQ DHW[m]*=T.Value("NJ",1,m+1);

 {
```

```
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ DHW[i] << ", ";
  res << mQ DHW[11] << " }";
  P.ModKey(Id,"Q-DHW",res.str());    // MJ per month
 }

  //!!! If this is the 2010 notional and this is NOT a not-daylit
zone (column 9 in notional tables),
  //!!! then adjust the hot water demand accordingly
  // The following adjustment is needed in order to adjust the hot
water demand of top-lit zones so that the
  // hot water energy consumption calculated later on in
DHW_GENERATOR::fDHW_ENERGY is correct. This is because
  // according to the 2010 NCM Modelling Guide, the efficiency of the
hot water system depends on whether the
  // zone is side-lit or top-lit but in SBEM, the DHW-GENERATOR
object can only have one efficiency, which in
  // DHW GENERATOR::fHEAT GEN EFF BH is set to that corresponding to
not daylit zones (according to P.Vint(gId,"NOT-CASE")=9)
  // from the tables. - AN (15/06/2010). The adjustment below relates
the demand to the correct efficiency from
  // the tables according to P.Vint(Id,"NOT-CASE") of the zone, i.e.,
according to DRIVER2A.
  if ( (P.Vcod(gId,"ACT-NOT")=="NOT") && (P.Vint(Id,"NOT-CASE")>9) )
{  //9=EW NOT 2010 Not Lit
    double mADJUST FAC=1.0;
    mADJUST FAC=fHW DEMAND ADJUST NOT(D, T, P);
    for (int i=0;i<12;i++)
      mQ DHW[i]*=mADJUST FAC;
  } // end if

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ DHW[i] << ", ";
  res << mQ DHW[11] << " }";
  P.ModKey(Id,"Q-DHW-CORR",res.str());     // MJ per month - adjusted
for the notional building
 }


//calculate distribution losses:
  double mDHW DIST LOSS[12]={ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };

  if (P.Vnum(Id,"DHW-PIPE-LEN")>D.Value("DHW 1"))
    for (int i=0;i<12;i++)
      mDHW DIST LOSS[i]=D.Value("DHW 2")*mQ DHW[i];

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
    res << mDHW_DIST_LOSS[i] << ", ";
   res << mDHW DIST LOSS[11] << " }";
   P.ModKey(Id,"DHW-DIST-LOSS",res.str());
  }

}

double ZONE::fHW DEMAND ADJUST NOT(PROJECTDefaults& D, PROJECTTables&
T, BOC& P) {
  if(func report)    cout    <<    endl    <<    "         "    <<
"ZONE::fHW DEMAND ADJUST NOT";
  // The following adjustment is needed in order to adjust the hot
water demand of top-lit zones so that the
  // hot water energy consumption calculated later on in
DHW GENERATOR::fDHW ENERGY is correct. This is because
  // according to the 2010 NCM Modelling Guide, the efficiency of the
```

```
hot water system depends on whether the
  // zone is side-lit or top-lit but in SBEM, the DHW-GENERATOR
object can only have one efficiency, which in
  // DHW GENERATOR::fHEAT GEN EFF BH is set to that corresponding to
not daylit zones (according to P.Vint(gId,"NOT-CASE")=9)
  // from the tables. - AN (15/06/2010). The adjustment below relates
the demand to the correct efficiency from
  // the tables according to P.Vint(Id,"NOT-CASE") of the zone, i.e.,
according to DRIVER2A.

  int HS_Id=P.Vcodid(P.Vrefid(Id,"DHW-GENERATOR"),"HEAT-SOURCE");
  int FT_Id=P.Vcodid(P.Vrefid(Id,"DHW-GENERATOR"),"FUEL-TYPE");
  int GT_Id=P.Vcodid(P.Vrefid(Id,"DHW-GENERATOR"),"HEAT-GEN-TYPE");

 double mADJUST_FAC=1.0;

  // if hot water generator type is a heat pump
  if (GT_Id==7) {
    // Check if the heating fuel for hot water is grid electricity,
i.e., electric heat pump
    if (FT_Id==10)
      mADJUST_FAC=T.Value("Not_Water_Heat_SEff_4",1,P.Vint(gId,"NOT-
CASE"))/T.Value("Not_Water_Heat_SEff_4",1,P.Vint(Id,"NOT-CASE"));
    else // i.e., non-electric heat pump
      mADJUST_FAC=T.Value("Not_Water_Heat_SEff_3",1,P.Vint(gId,"NOT-
CASE"))/T.Value("Not_Water_Heat_SEff_3",1,P.Vint(Id,"NOT-CASE"));
  } // end if
  else { // i.e., not an independent heat pump
    // Check if the hot water is provided by an HVAC system through
an electric heat pump
    // 1=Using Central Heating boiler , 18=Heat pump (electric): air
source , 19=Heat pump (electric): ground or water source
    // 10=Grid Supplied Electricity
    if ( (GT_Id==1) && ((HS_Id==18) || (HS_Id==19)) && (FT_Id==10) )
      mADJUST_FAC=T.Value("Not_Water_Heat_SEff_4",1,P.Vint(gId,"NOT-
CASE"))/T.Value("Not_Water_Heat_SEff_4",1,P.Vint(Id,"NOT-CASE"));

    else {
      // Check if the hot water is provided by an HVAC system through
a non-electric heat pump
      // 1=Using Central Heating boiler , 12=Heat pump (gas/oil): air
source , 13=Heat pump (gas/oil): ground or water source
      if ( (GT_Id==1) && ((HS_Id==12) || (HS_Id==13)) )

mADJUST_FAC=T.Value("Not_Water_Heat_SEff_3",1,P.Vint(gId,"NOT-
CASE"))/T.Value("Not_Water_Heat_SEff_3",1,P.Vint(Id,"NOT-CASE"));

      else {
        // Check if the heating fuel for hot water is district
heating - (no adjustment needed here)
        // 1=Using Central Heating boiler , 13=District Heating ,
15=District heating
        if ( (GT_Id==1) && (FT_Id==13) && (HS_Id==15) )
          mADJUST_FAC=1.0;

        else { // i.e., any case other than those above
          // Check if the heating fuel for hot water is one whose
emission factor <= 0.156 kg/kWh
          // 3=Biogas, 9=Biomass, 12=Waste heat
          if ( (FT_Id==3) || (FT_Id==9) || (FT_Id==12) )

mADJUST_FAC=T.Value("Not_Water_Heat_SEff_1",1,P.Vint(gId,"NOT-
CASE"))/T.Value("Not_Water_Heat_SEff_1",1,P.Vint(Id,"NOT-CASE"));
          else // i.e., the heating fuel for hot water is one whose
emission factor > 0.156 kg/kWh

mADJUST_FAC=T.Value("Not_Water_Heat_SEff_2",1,P.Vint(gId,"NOT-
CASE"))/T.Value("Not_Water_Heat_SEff_2",1,P.Vint(Id,"NOT-CASE"));
        } // end else
```

```
      } // end else
    } // end else
  } // end else

  return(mADJUST_FAC);

} // end fHW DEMAND ADJUST NOT



// Lighting starts here

void ZONE::fLIGHT ENERGY(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func report) cout << endl << "  " << "ZONE::fLIGHT ENERGY";
//calculate lighting energy use for the zone

  fLIGHT WHJ(D,T,P);
  fLIGHT DISPLAY(D,T,P);
  fLIGHT_PARASITIC(D,T,P);
  fLIGHT FOD(D,T,P);
  double mPDJ=P.Vnum(Id,"LIGHT-DISP-PW");
  double mPP=P.Vnum(Id,"LIGHT-PARAS-PW");
  double mFOD=P.Vnum(Id,"LIGHT-FOD");
  double mQ I LI[12];



// 30 Oct 08, JO: Add final correction for activity based on OCC-T-
HOURS correction.
   deque<double> mOCC T HOURS=P.Vlnum(Id,"OCC-T-HOURS");
   deque<double> mLIG T START=P.Vlnum(Id,"LIGHT-T-START");
   deque<double> mLIG T END=P.Vlnum(Id,"LIGHT-T-END");

//   cout << endl << P.Vtext(Id,"OCC-T-HOURS");

  double mDAYS IN MONTH[12];
  for(int i=0;i<12;i++) mDAYS IN MONTH[i]=T.Value("NJ",1,i+1);

  double mPDJ M[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };

  for(int                                    month=0;month<12;month++)
mPDJ M[month]=mOCC T HOURS[month]/mDAYS IN MONTH[month]*mPDJ*mFOD;///
mDAYS IN MONTH[month];

  double mLIGHT METERING=P.Vnum(gId,"LIGHT-METERING");
  double mlight corr, mDISP HOURS IN MONTH;
  for(int i=0;i<12;i++) {
    if ((mLIG T END[i]-mLIG T START[i])>0)
//
mlight_corr=mOCC_T_HOURS[i]/(mDAYS_IN_MONTH[i]*(mLIG_T_END[i]-
mLIG T START[i]));
  //  !!  The  correction  below  by  adding  1  to  (mLIG T END[i]-
mLIG_T_START[i]) was added because mPJF was calculated
  // !! in the function fLIGHT WHJ from mLIG T START to mLIG T END
inclusive. - AN (25/05/2010)
      mlight_corr=mOCC_T_HOURS[i]/(mDAYS_IN_MONTH[i]*((mLIG_T_END[i]-
mLIG T START[i])+1.0));
    else
      mlight_corr=0.0;

    //mDISP HOURS IN MONTH=24*mDAYS IN MONTH[i];      //   mFOD   will
correct if there is a mechanism to switch off display lighting
//
mLIGHT ENERGY[i]=(mDAYS IN MONTH[i]*(mPJF[i]*mlight corr+24*(mPP+mPDJ
*mFOD)))/1000*P.Vnum(gId,"LIGHT-METERING");  //KWh/m2/month  //v1.0.b
addition of metering correction
    // Based on meeting with Aecom, the line above has been adjusted
so that the display lighting is operated
```

```
    // during lighting hours like the general lighting instead of 24
hours a day. AN (8/04/2010).

mLIGHT ENERGY[i]=(mDAYS IN MONTH[i]*((mPJF[i])*mlight corr+(24*mPP)+m
PDJ_M[i]))/1000*mLIGHT_METERING;   //KWh/m2/month   //v1.0.b addition
of metering correction
  }

  double mLIGHT GAIN F=P.Vnum(Id,"LIGHT-GAIN-F");
  if (mLIGHT GAIN F==-6666) mLIGHT GAIN F=D.Value("LIGHT GAIN F");
  for(int i=0;i<12;i++) {
     if(mLIGHT METERING!=0)
       mLIGHT GAIN[i]=mLIGHT ENERGY[i]*mLIGHT GAIN F/mLIGHT METERING;
     else
       mLIGHT GAIN[i]=mLIGHT ENERGY[i]*mLIGHT GAIN F;
     mQ I LI[i]=mLIGHT GAIN[i]*P.Vnum(Id,"AREA")*3.6;
  }

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mLIGHT ENERGY[i] << ", ";
   res << mLIGHT ENERGY[11] << " }";
   P.ModKey(Id,"LIGHT-ENERGY",res.str());  //KWH/m2
  }

  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
      res << mLIGHT GAIN[i] << ", ";
   res << mLIGHT GAIN[11] << " }";
   P.ModKey(Id,"LIGHT-GAIN",res.str());  //KWH/m2
  }
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mQ I LI[i] << ", ";
   res << mQ I LI[11] << " }";
   P.ModKey(Id,"Q-I-LI",res.str());      //MJ
  }

}

void ZONE::fLIGHT WHJ(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "   " << "ZONE::fLIGHT_WHJ";

  fLIGHT PW(D,T,P);

  double mPW=P.Vnum(Id,"LIGHT-PW");
  deque<double> mBR LIGHT CHECK=P.Vlnum(Id,"BR-LIGHT-CHECK");
  double mPLG=0.0;

  fLIGHT DF(D,T,P);
  fLIGHT_FO(D,T,P);

  double mDFf, mDFb, mDF;
  mDF=P.Vnum(Id,"LIGHT-DF");
  mDFf=P.Vnum(Id,"LIGHT-DF-F");
  mDFb=P.Vnum(Id,"LIGHT-DF-B");
  double mDL_AREA=P.Vnum(Id,"LIGHT-DL-AREA-PERC")/100.0;

  double mExt=0.0;
  double mEIntf=0.0;
  double mEIntb=0.0;
  double mEInt=0.0;
  double mFDf=1.0;
  double mFDb=1.0;

  double mFD_PH=1.0;
```

```
  double mFD MS=1.0;

  double mLIGHT ILLUM=P.Vnum(Id,"LIGHT-ILLUM");

  for(int month=0;month<12;month++) {
      for(double h=0.0;h<24.0;h++) {
          if(mLIGHT T START[month]<=h && mLIGHT T END[month]>=h) {
               // If any kind of control
               if(P.Vcodid(Id,"LIGHT-CONTROL")!=4) {
                 mExt=fLIGHT EEXT H(D,T,P,month+1,h);
                 mEIntf=mExt*mDFf*10;
                 mEIntb=mExt*mDFb*10;
                 mEInt=mExt*mDF*10;
               }
               // If Photoelectric
               if          (P.Vcodid(Id,"LIGHT-CONTROL")==2          ||
P.Vcodid(Id,"LIGHT-CONTROL")==3) {
                 if (P.Vcodid(Id,"LIGHT-PHOTO-TYPE")==2) { // dimming
                   // front
                   if(mEIntf>mLIGHT_ILLUM) mFDf=0.0;
                   else
                     if(mLIGHT ILLUM>0)            mFDf=(mLIGHT ILLUM-
mEIntf)/mLIGHT ILLUM;
                     else mFDf=1.0;
                   // back
                   if(mEIntb>mLIGHT ILLUM) mFDb=0.0;
                   else
                     if(mLIGHT ILLUM>0)            mFDb=(mLIGHT ILLUM-
mEIntb)/mLIGHT ILLUM;
                     else mFDb=1.0;
                 }
                 else { // switching
                   // front
                   if(mEIntf>mLIGHT ILLUM) mFDf=0.0;
                   else if(mEIntf>0.5*mLIGHT ILLUM) mFDf=0.5;
                   else mFDf=1.0;
                   // back
                   if(mEIntb>mLIGHT ILLUM) mFDb=0.0;
                   else if(mEIntb>0.5*mLIGHT_ILLUM) mFDb=0.5;
                   else mFDb=1.0;

                 }
                 // If there is NO photosensor in the back of the
room AND the zone does not contain windows in opposite
                 // orientations AND it does not contain rooflights
                 if(       (P.Vcodid(Id,"LIGHT-PH-BACK")==2)       &&
(P.Vcodid(Id,"LIGHT-WIN/ROOF")!=3) )// 2=NO , 3=OTHER
                   mFDb=1.0;

//                 mFD PH=(mFDf+mFDb)/2.0;
                 mFD_PH=mDL_AREA*mFDf+(1.0-mDL_AREA)*mFDb;
               }

               // if (there is manual switching) AND (the zone is NOT
one of the restricted activites) AND (it is less than 30m2 in area)
               //LIGHT-CONTROL:  1-manual(def),  2-photoelectric,  3-
both, 4-none
               if     (     ((P.Vcodid(Id,"LIGHT-CONTROL")==1)     ||
(P.Vcodid(Id,"LIGHT-CONTROL")==3))   &&   (mBR LIGHT CHECK[1]==0)   &&
(P.Vnum(Id,"AREA")<D.Value("LIGHT_CEN_AREA")) ) {
//                 fLIGHT FD MS(D,T,P,month+1,h);
                 if(mEInt>mLIGHT ILLUM) {
                    mFD_MS=0.5;
                    if(P.Vcodid(Id,"LIGHT-WIN/ROOF")==1)  // 1=SIDE
                      mFD MS=mDL AREA*mFD MS+(1.0-mDL AREA)*mFD MS;
                 }
                 else mFD MS=1.0;
               }
               mFD = mFD_PH;
```

```
                if(mFD MS<mFD PH) mFD=mFD MS;

//                              double  mDL AREA=P.Vnum(Id,"LIGHT-DL-AREA-
PERC")/100.0;
//                mFD = mDL_AREA*mFD+(1.0-mDL_AREA)*1.0;

                if (mPW != -1.0) {
                  double mPJF_month=mPW*mFD*mFO[month];
                  mPJF[month]+=mPJF month;


                  if(month==8) // Sept for peak lighting gains  // JO
04  Jun  09,  Change  mPJF month  for  mPW  to  neglect  influence  of
controls...
                     if(mPW>mPLG) mPLG=mPW;
                }

           }

      }

  }

    {
    ostringstream res; res << "{ ";
    for (int i=0;i<11;i++)
        res << mPJF[i] << ", ";
    res << mPJF[11] << " }";
    P.ModKey(Id,"LIGHT-WHJ",res.str());
    }
    {
    ostringstream res; res << mPLG;
    P.ModKey(Id,"AE-PLG",res.str());
    }
}

void ZONE::fLIGHT PW(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "ZONE::fLIGHT PW";
  double mAREA=P.Vnum(Id,"AREA");
  double mLIGHT POWER DENS=0.0;                          //light power
density
  double  mLIGHT_ILLUM=P.Vnum(Id,"LIGHT-ILLUM");          //DATABASE
illuminance in zone
  deque<double> mBR LIGHT CHECK=P.Vlnum(Id,"BR-LIGHT-CHECK");
  double     mLIGHT_FACTOR_EQU=T.Value("NOT_PDC_0",3,P.Vint(gId,"NOT-
CASE"));   // initialising to the value of C in equation below

  //find which of the three cases you are in:
  switch (P.Vcodid(Id,"LIGHT-CASE")) {
     //lighting parameters not available:
     case 1: {
        // If this is the actual building or the aggregate notional,
apply the new lighting tables and calculation
        if(   (P.Vcod(gId,"ACT-NOT")=="ACT")   ||   ((P.Vcod(gId,"ACT-
NOT")=="NOT") && (P.Vint(Id,"NOT-CASE")>=9)) ) { // 9=EW NOT 2010

          double mWALL_FLOOR=P.Vnum(Id,"LIGHT-WALL/FLOOR-AREA");
          double A=0.0; double B=0.0; double C=0.0;

          if(mWALL_FLOOR>=0) {
            if(P.Vcod(gId,"ACT-NOT")=="ACT") {
              A=T.Value("ACT PDC",1,P.Vint(Id,"DRIVER2B")+1);
              B=T.Value("ACT_PDC",2,P.Vint(Id,"DRIVER2B")+1);
              C=T.Value("ACT PDC",3,P.Vint(Id,"DRIVER2B")+1);
            } // end if
            else { // i.e., 2010 notional
              switch(P.Vint(Id,"DRIVER2B")) {
                case 0: {  // Unobstructed spaces
                          A=T.Value("NOT_PDC_0",1,P.Vint(Id,"NOT-
```

```
CASE"));
                          B=T.Value("NOT PDC 0",2,P.Vint(Id,"NOT-
CASE"));
                          C=T.Value("NOT PDC 0",3,P.Vint(Id,"NOT-
CASE"));
                         }
                         break;
                case 1: { // Spaces with fixed partitions
                          A=T.Value("NOT PDC 1",1,P.Vint(Id,"NOT-
CASE"));
                          B=T.Value("NOT_PDC_1",2,P.Vint(Id,"NOT-
CASE"));
                          C=T.Value("NOT PDC 1",3,P.Vint(Id,"NOT-
CASE"));
                         }
                         break;
                case 2: { // Spaces with moveable obstructions
                          A=T.Value("NOT PDC 2",1,P.Vint(Id,"NOT-
CASE"));
                          B=T.Value("NOT_PDC_2",2,P.Vint(Id,"NOT-
CASE"));
                          C=T.Value("NOT PDC 2",3,P.Vint(Id,"NOT-
CASE"));
                         }
                         break;
                default: {
                           A=T.Value("NOT PDC 0",1,P.Vint(Id,"NOT-
CASE"));
                           B=T.Value("NOT PDC 0",2,P.Vint(Id,"NOT-
CASE"));
                           C=T.Value("NOT PDC 0",3,P.Vint(Id,"NOT-
CASE"));
                          }
              } // end switch
            } // end else
          } // end if

          mLIGHT FACTOR EQU=(A*pow(mWALL FLOOR,2))+(B*mWALL FLOOR)+C;
          mLIGHT_POWER_DENS=mLIGHT_FACTOR_EQU*(mLIGHT_ILLUM/100.0);

          // In the actual building, adjust the power density by the
ratio of the luminous efficacy of the notional
          // building to that of the actual, which is retrieved from
the tables depending on the configuration of
          // the activity as side-lit or top-lit and the lamp type
selected for the zone.
          if(P.Vcod(gId,"ACT-NOT")=="ACT")

mLIGHT POWER DENS*=D.Value("dLUMIN EFF NOT")/T.Value("tLUMINOUS-
EFFICACY",P.Vint(Id,"DRIVER2A")+1,P.Vcodid(Id,"LIGHT-TYPE"));
        } // end if
        else  // other notional buildings, i.e., NOT-CASE<9

mLIGHT_POWER_DENS=T.Value("NOT_LIGHT_FACTORS",P.Vcodid(Id,"LIGHT-
TYPE"),P.Vint(gId,"NOT-CASE"))*(mLIGHT ILLUM/100.0);
     }
     break;
     //full lighting design carried out, including verification that
illuminances meet standard levels:
     case 2: { // only the actual building can have this option
        //use actual lighting circuit Power, divide this by zone
area:
        if (mAREA!=0.0) {
          double     mLIGHT LUX DESIGN=P.Vnum(Id,"LIGHT-LUX-DESIGN");
//Added in version 1.2.a to correct illuminance with user values. JO
27Jul06
//          if (mLIGHT LUX DESIGN==-6666)
          if  ((mLIGHT LUX DESIGN==-6666)||(mLIGHT LUX DESIGN==0))  //
condition added to prevent division by 0
```

```
            mLIGHT LUX DESIGN=mLIGHT ILLUM;
          double mLIGHT ACT WATT=P.Vnum(Id,"LIGHT-ACT-WATT");
          if (mLIGHT ACT WATT==-6666) mLIGHT POWER DENS=10;
          else
mLIGHT_POWER_DENS=(mLIGHT_ACT_WATT*mLIGHT_ILLUM/mLIGHT_LUX_DESIGN/mAR
EA);  //Modified in version 1.2.a to correct illuminance with user
values. JO 27Jul06
        }
        else
            mLIGHT POWER DENS=-1.0;

     }
     break;
     //lighting chosen but full illuminance calculation has not been
carried out:
     case 3: { // only the actual building can have this option
       double mWALL_FLOOR=P.Vnum(Id,"LIGHT-WALL/FLOOR-AREA");
       double A=0.0; double B=0.0; double C=0.0;

       if(mWALL_FLOOR>=0) {
         A=T.Value("ACT PDC",1,P.Vint(Id,"DRIVER2B")+1);
         B=T.Value("ACT PDC",2,P.Vint(Id,"DRIVER2B")+1);
         C=T.Value("ACT PDC",3,P.Vint(Id,"DRIVER2B")+1);
       } // end if
       mLIGHT FACTOR EQU=(A*pow(mWALL FLOOR,2))+(B*mWALL FLOOR)+C;
       mLIGHT POWER DENS=mLIGHT FACTOR EQU*(mLIGHT ILLUM/100.0);

       // In the actual building, adjust the power density by the
ratio of the luminous efficacy of the notional
       // building to that of the actual, which is retrieved from the
tables depending on the configuration of
       // the activity as side-lit or top-lit and the lamp type
selected for the zone.
       if ( (P.Vnum(Id,"LAMP-BALLAST-EFF")>0.0) && (P.Vnum(Id,"LIGHT-
OUTPUT-RATIO")>0.0) )

mLIGHT POWER DENS*=D.Value("dLUMIN EFF NOT")/(P.Vnum(Id,"LAMP-
BALLAST-EFF")*P.Vnum(Id,"LIGHT-OUTPUT-RATIO"));
       else
         mLIGHT POWER DENS = -1.0;
     }
     break;
  }
  //store power/m^2:
  {
    ostringstream res; res << mLIGHT POWER DENS;
    P.ModKey(Id,"LIGHT-PW",res.str());
  }

  // If this is the actual building or the aggregate notional
  if(     (P.Vcod(gId,"ACT-NOT")=="ACT")     ||     ((P.Vcod(gId,"ACT-
NOT")=="NOT") && (P.Vint(Id,"NOT-CASE")>=9)) )  // 9=EW NOT 2010
   {
     ostringstream res; res << mLIGHT_FACTOR_EQU;
     P.ModKey(Id,"LIGHT-FACTOR-EQU",res.str());
   }

  // Adjust the parasitic power of the aggregate 2010 notional
building to the lesser of the current value and 3%
  // of the installed lighting power as calculated above, if it's an
activity with light dimming.
  if  ( (P.Vcod(gId,"ACT-NOT")=="NOT")  &&  (P.Vint(Id,"NOT-CASE")>=9)
&& (P.Vcodid(Id,"DIMMING-NOT")==1) ) { // 1=YES
    if                                                          (
(mLIGHT POWER DENS*D.Value("dPARAS PWR NOT2"))<P.Vnum(Id,"LIGHT-
PARAS-PW") )
      {
        ostringstream             res;             res             <<
(mLIGHT_POWER_DENS*D.Value("dPARAS_PWR_NOT2"));
```

```
        P.ModKey(Id,"LIGHT-PARAS-PW",res.str());
      }
  } // end if

}

void ZONE::fLIGHT DF(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "  " << "ZONE::fLIGHT_DF";
// 31 Dec 07 JO
// This function is OK to add the calculation of the automatic
daylight zoning
// in SBEM. This will depend on the amount of windows and rooflights,
orientations,
// and size of the zone.
// Everything is already considered here for the calculation of the
DF, so this is
// the ideal place to add this calculation. Additionaly and very
important, this
// function is called before the claculation of FD MS or FD PH.

  double mWINDOW AREA=0.0;
  double mROOFLIGHT AREA=0.0;
  double mSURFACE AREA=0.0;
  double mDF1=0.0;
  double mDF2=0.0;
  double mDF=0.0;
  double mDFf=0.0;
  double mDFb=0.0;
  int mWind[9]={0, 0, 0, 0, 0, 0, 0, 0, 0};
  double  mArea Check[9]={0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,
0.0}; // to check ratio of daylight contribution between opposite
sides

  // 31 Dec 07 Added
  double mAREA=P.Vnum(Id,"AREA");
  double mDL AREA=0.0;
  double  mWALL LENGTH[9]={0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,
0.0};
  double  mWALL DL AREA[9]={0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,
0.0};
  double mWALL WITHWIN LENGTH[9]={0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0};
  double  mWALL LENGTH SUBS[9]={0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,
0.0, 0.0};
  double           mROOFLIGHT AREA 1,           mROOFLIGHT AREA 1 cov,
mROOFLIGHT AREA 1 inc;
  double mROOF DL AREA=0.0;
  double mHEIGHT=P.Vnum(Id,"HEIGHT"); if(mHEIGHT==0) mHEIGHT=3.0;
  bool fWindows;
  int mORIENTATION;

  // mLIGHT TRANS DEF is the glazing light transmittance for normal
incidence for which the DF equations were derived
  // these equations are adjusted for the actual transmittance below
  double mLIGHT TRANS DEF=D.Value("dGlazing L Solar");
  double  mLIGHT TRANS=D.Value("dGlazing L Solar");        // this is
overwritten below by the actual transmittance
  double mLIGHT TRANS FACTOR=1;
//  if (mLIGHT TRANS DEF==0) mLIGHT TRANS DEF=0.76;

  for (int i=0;i<vW.size();i++) {

mSURFACE_AREA+=P.Vnum(vW[i].Id,"AREA")*P.Vnum(vW[i].Id,"MULTIPLIER");
    fWindows=0;
    mORIENTATION=P.Vcodid(vW[i].Id,"ORIENTATION")-1;
    for (int j=0;j<vW[i].vW.size();j++) {

//mSURFACE AREA+=P.Vnum(vW[i].vW[j].Id,"AREA")*P.Vnum(vW[i].vW[j].Id,
"MULTIPLIER");
```

```
        mWind[mORIENTATION]++;
        // sum area of glazing in each orientation to check ratio of
daylight contribution between opposite sides
        mArea Check[mORIENTATION]+=P.Vnum(vW[i].vW[j].Id,"AREA")*(1-
P.Vnum(vW[i].vW[j].Id,"FRAME-
FACTOR"))*P.Vnum(vW[i].vW[j].Id,"MULTIPLIER")*P.Vnum(vW[i].Id,"MULTIP
LIER");

        // Use the transmittance value for normal incidence
        mLIGHT TRANS=P.Vnum(vW[i].vW[j].Id,"LIG-SOL-TRANS-ORI");

        if (mLIGHT TRANS DEF!=0.0)
          mLIGHT TRANS FACTOR=mLIGHT TRANS/mLIGHT TRANS DEF;

        if(P.Vcodid(vW[i].Id,"TYPE-ENV")==1) {                    // a
Wall
            mWINDOW_AREA+=P.Vnum(vW[i].vW[j].Id,"AREA")*(1-
P.Vnum(vW[i].vW[j].Id,"FRAME-
FACTOR"))*P.Vnum(vW[i].vW[j].Id,"MULTIPLIER")*P.Vnum(vW[i].Id,"MULTIP
LIER")*mLIGHT_TRANS_FACTOR;
            fWindows=1;
        }
        else if(P.Vcodid(vW[i].Id,"TYPE-ENV")==2) {               // a
Roof
            mROOFLIGHT AREA 1=P.Vnum(vW[i].vW[j].Id,"AREA")*(1-
P.Vnum(vW[i].vW[j].Id,"FRAME-
FACTOR"))*P.Vnum(vW[i].vW[j].Id,"MULTIPLIER")*P.Vnum(vW[i].Id,"MULTIP
LIER")*mLIGHT TRANS FACTOR;
            mROOFLIGHT AREA+=mROOFLIGHT AREA 1;
// The lines below have been commented out as we will assume that
rooflights are spread evenly across the roof and
// therefore, affect the whole area of the zone equally and there is
no need to use ROOF-LIGHT-COVER. We will only
//  be calculating the area of the zone affected by windows in
mDL AREA (below) and then saving the percentage
// in LIGHT-DL-AREA-PERC - AN (30/09/2010)
/*

mROOFLIGHT_AREA_1_cov=mROOFLIGHT_AREA_1*P.Vnum(vW[i].vW[j].Id,"ROOF-
LIGHT-COVER");
//
mROOFLIGHT_AREA_1_inc=4.0*(mHEIGHT*4.5/3.0)*(mHEIGHT*4.5/3.0+sqrt(mRO
OFLIGHT AREA 1 cov));

mROOFLIGHT_AREA_1_inc=pow((sqrt(mROOFLIGHT_AREA_1_cov)+2.0*mHEIGHT*D.
Value("DL ROOFHEIGHT GAIN")),2.0);
//cout << endl << mROOFLIGHT AREA 1 << "   " << mROOFLIGHT AREA 1 cov
<< "   " << mROOFLIGHT_AREA_1_inc << endl;
            mROOF DL AREA+=mROOFLIGHT AREA 1 inc;
*/
        }

    }

mWALL LENGTH[mORIENTATION]+=P.Vnum(vW[i].Id,"LENGTH")*P.Vnum(vW[i].Id
,"MULTIPLIER");

    if(fWindows)

mWALL_WITHWIN_LENGTH[mORIENTATION]+=P.Vnum(vW[i].Id,"LENGTH")*P.Vnum(
vW[i].Id,"MULTIPLIER");
  }

  // Calculation of percentage of area affected by daylight
  if(P.Vcod(Id,"LIGHT-DL-AUT-ZONE")=="YES") {
// The line below have been commented out as we will assume that
rooflights are spread evenly across the roof and
// therefore, affect the whole area of the zone equally and there is
no need to use ROOF-LIGHT-COVER. We will only
```

```
// be calculating the area of the zone affected by windows in
mDL AREA (below) and then saving the percentage
// in LIGHT-DL-AREA-PERC - AN (30/09/2010)
// Before we save LIGHT-DL-AREA-PERC, a further check has been added:
// If there are rooflights in the zone but no windows, then assume
that 100% of the zone benefits from daylight
// through the rooflights - AN (13/11/2010)

//    mDL AREA=mROOF DL AREA;
    // Simple algorithm to work out overlapping areas of daylight due
to
    // windows in walls with different orientations.
    int j,k;
    double           com max a45=D.Value("DL A O45");           double
perc shading a45=D.Value("DL P O45");
    double       com max a90=D.Value("DL A O90");               double
perc_shading_a90=D.Value("DL_P_O90");
    double area temp, area perc temp, area sus temp;
    // Area due to 6 m daylight zoning rule
    for                          (int                         i=0;i<8;i++)
mWALL DL AREA[i]=mWALL WITHWIN LENGTH[i]*D.Value("DL Depth");
    double mMAX WALL DL AREA=0;
    for  (int  i=0;i<8;i++)  if  (mWALL DL AREA[i]>mMAX WALL DL AREA)
mMAX WALL DL AREA=mWALL DL AREA[i];
    // Area that orientation i overlaps with orientations i+1 and i+2
    for (int i=0;i<8;i++) {
      j=i+1; if(j>=8) j-=8;
      k=i+2; if(k>=8) k-=8;
      if(mWALL DL AREA[i]>0 && mWALL DL AREA[j]>0) {
        area temp=mWALL DL AREA[i]+mWALL DL AREA[j];
        area perc temp=perc shading a45*area temp;
        area sus temp=  (area perc temp<com max a45)  ?  area perc temp
: com max a45;
        mWALL DL AREA[i]-=area sus temp/2.0;
        mWALL DL AREA[j]-=area sus temp/2.0;
      }
      else if(mWALL WITHWIN LENGTH[i]>0 && mWALL WITHWIN LENGTH[k]>0)
{
        area_temp=mWALL_DL_AREA[i]+mWALL_DL_AREA[k];
        area perc temp=perc shading a90*area temp;
        area sus temp=  (area perc temp<com max a90)  ?  area perc temp
: com_max_a90;
        mWALL DL AREA[i]-=area sus temp/2.0;
        mWALL DL AREA[k]-=area sus temp/2.0;
      }
    }
    for (int i=0;i<8;i++) mDL AREA+=mWALL DL AREA[i];

    if (mDL AREA<mMAX WALL DL AREA) mDL AREA=mMAX WALL DL AREA;
    if (mDL AREA>mAREA) mDL AREA=mAREA;

    // !!! If there are rooflights in the zone but no windows, then
assume that 100% of the zone benefits from
    // daylight through the rooflights - AN (13/11/2010)
    if((mROOFLIGHT AREA>0.0)&&(mDL AREA==0.0))
      mDL AREA=mAREA;

    {
     ostringstream res; res << mDL AREA/mAREA*100;
     P.ModKey(Id,"LIGHT-DL-AREA-PERC",res.str());
    }
  } // end if
  else // i.e., sub-zoning due to daylight done by user
    mDL AREA=P.Vnum(Id,"LIGHT-DL-AREA-PERC")*mAREA/100.0;

  if(mSURFACE_AREA==0) {mDF1=0; mDF2=0; mDF=0; mDFf=0; mDFb=0;}
  else {
        mDF1=D.Value("DF WINDOW")*mWINDOW AREA/mSURFACE AREA;
        mDF2=D.Value("DF_ROOFLIGHT")*mROOFLIGHT_AREA/mSURFACE_AREA;
```

```
        mDF=mDF1+mDF2;

//        mDFf=1.75*mDF;
//        mDFb=mDF*2-mDFf;

// New calculation for Front and Back daylight factors
//  We are assuming that the front is affected by daylight from
windows and if they exist, rooflights, while the back
// is affected by rooflights only. If there are windows at both ends,
so that the whole zone benefits from windows,
// this should still be covered by mDL_AREA - AN (30/09/2010)
// Note mDL AREA were calculated above based on windows only in the
zone (not rooflights)
        if  (mDL_AREA>0.0)  //  Ensure  no  divison  by  zero  -  AN
(12/11/2010)
          mDFf=mDF1+mDF2;
        else
          mDFf=mDF2;

        mDFb=mDF2;
  }


  string mLIGHT WIN ROOF="SIDE";

  if(mWind[8]!=0) mLIGHT WIN ROOF="OTHER";
  else {
    int sum=0;
    for(int i=0;i<8;i++) sum+=mWind[i];

    if(sum==0) mLIGHT WIN ROOF="NONE";
    else
      // Check if there are opposite windows (Just need to check half
range orientations)
      for(int i=0;i<5;i++) {
         // Check that there is glazing in opposite sides and that
the ratio of daylight contribution between the
         // opposite sides is less than 3:1 (the check is done on the
net area of the glazing)
         // if( (mWind[0]!=0 && mWind[4]!=0))
         if(mWind[0]!=0    &&    (mWind[3]!=0    ||    mWind[4]!=0    ||
mWind[5]!=0)) {
           if(mArea Check[3]!=0)
             if(         (mArea Check[0]/mArea Check[3]<3.0)         &&
(mArea_Check[0]/mArea_Check[3]>=0.334) ) {
                mLIGHT WIN ROOF="OTHER";
                break;
             }
           if(mArea Check[4]!=0)
             if(         (mArea Check[0]/mArea Check[4]<3.0)         &&
(mArea_Check[0]/mArea_Check[4]>=0.334) ) {
                mLIGHT WIN ROOF="OTHER";
                break;
             }
           if(mArea Check[5]!=0)
             if(         (mArea Check[0]/mArea Check[5]<3.0)         &&
(mArea_Check[0]/mArea_Check[5]>=0.334) ) {
                mLIGHT WIN ROOF="OTHER";
                break;
             }
         } // end if
         mWind[8]=mWind[0];  mWind[0]=mWind[1];  mWind[1]=mWind[2];
         mWind[2]=mWind[3];  mWind[3]=mWind[4];  mWind[4]=mWind[5];
         mWind[5]=mWind[6];  mWind[6]=mWind[7];  mWind[7]=mWind[8];
      } // end for
  } // end else

 {
  ostringstream res; res << mLIGHT_WIN_ROOF;
```

```
  P.ModKey(Id,"LIGHT-WIN/ROOF",res.str());
 }

 if(P.Vnum(Id,"LIGHT-DF")==-6666)
    {
     ostringstream res; res << mDF;
     P.ModKey(Id,"LIGHT-DF",res.str());
    }
    {
     ostringstream res; res << mDFf;
     P.ModKey(Id,"LIGHT-DF-F",res.str());
    }
    {
     ostringstream res; res << mDFb;
     P.ModKey(Id,"LIGHT-DF-B",res.str());
    }

}

//calculate occupancy correction factor for zone:
void ZONE::fLIGHT FO(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "   " << "ZONE::fLIGHT FO";
  deque<double> mBR LIGHT CHECK=P.Vlnum(Id,"BR-LIGHT-CHECK");

  //FO equals 1 if the lighting is switched on 'centrally'
  for(int i=0;i<12;i++) {
    mFO[i]=1.0;

    //LIGHT-OCC-SENS-T:  1)MAN-ON-OFF+EXT,  2)AUTO-ON-DIMMED,  3)AUTO-
ON-OFF, 4)MAN-ON-DIMMED, 5)MAN-ON-AUTO-OFF, 6)NONE
    int mLIGHT OCC SENS T=P.Vcodid(Id,"LIGHT-OCC-SENS-T");
    //LIGHT-CONTROL: 1)MANUAL, 2)PHOTOELECTRIC, 3)BOTH, 4)NONE
    //BR LIGHT CHECK03 => mBR LIGHT CHECK[2]: whether zone activity
is meeting room
    //BR LIGHT CHECK02 => mBR LIGHT CHECK[1]: whether zone activity
is one of corridor or other circulation area, dry sports, changing
room, swimming pool, sales area, cold store, display area, and
performance area (stage)
    //In corridors and other circulation areas and sales and display
areas etc. as above, FO equals 1 even if occupancy sensing or manual
control is provided, unless a time switch switches off the lighting

    //if (it is a meeting room) OR (its area is < 30m2) AND (there is
occupancy sensing)
    if            (            ((mBR LIGHT CHECK[2]==1)            ||
(P.Vnum(Id,"AREA")<D.Value("LIGHT CEN AREA")))                       &&
(mLIGHT OCC SENS T!=6) )
       // if it is NOT one of the restricted activity areas
       if (mBR LIGHT CHECK[1]==0)
           mFO[i]=(T.Value("LIGHT FOC",1,mLIGHT OCC SENS T));
       // else it IS one of the restricted activity areas
       else if ( (mLIGHT OCC SENS T==2) || (mLIGHT OCC SENS T==3) )
// 2 = Auto on/off, 3 = Auto on/dimmed
              mFO[i]=(T.Value("LIGHT_FOC",1,mLIGHT_OCC_SENS_T));

    // Outside the above conditions, NO benefit from occupancy
sensing is considered and NO parasitic power is added for it
  }
}


double ZONE::fLIGHT EEXT H(PROJECTDefaults& D, PROJECTTables& T, BOC&
P, int month, double h) {
  if(func report) cout << endl << "   " << "ZONE::fLIGHT EEXT H " <<
month << "   " << h;

 string Tname;
 char Temp[10]; char *pTemp=" ";
 double mExt_h;
```

```
 string mWEATHER;

    if(h>=D.Value("T3 Eext ini") && h<=D.Value("T3 Eext end")+1) {
        sprintf(Temp,"%s%d",pTemp,(int)(floor(h+.5)));
        string s(Temp);
        Tname=s+"30";
        // If the calculation is for the Republic of Ireland or
Jersey
        if(   (P.Vcodid(cId,"BR-TYPE")==5)   ||   (P.Vcodid(cId,"EPC-
TYPE")==6) || (P.Vcodid(cId,"BR-TYPE")==6) )
          // Continue to use the Kew illuminance data
          mExt h=T.Value("T3 Eext"+Tname,1,month);
        else { // For all other calculations, use the location-
specific illuminance data provided by AECOM
          mWEATHER=" "+P.Vcod(gId,"WEATHER");
          mExt h=T.Value("T3 Eext"+mWEATHER+Tname,1,month);
        } // end else
    }
    else {
        mExt_h=0.0;
    }

    return(mExt h);
}

void ZONE::fLIGHT DISPLAY(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func report) cout << endl << "  " << "ZONE::fLIGHT DISPLAY";
  double mLIGHT DISPLAY POWER;
  //find installed power for display lighting:

  //are efficent lamps used for display lighting:

 {
  ostringstream res; res << P.Vnum(Id,"LIGHT-DISP-PW");
  P.ModKey(Id,"LIGHT-DISP-PW-ORI",res.str());
 }

  // !!! N.B. The 2010 notional building uses efficienct lamps for
display lighting
  // Check if display lighting uses efficieny lamps
  if(P.Vcodid(Id,"LIGHT-DISPLAY-EFF")==1) {  // 1=yes
     mLIGHT DISPLAY POWER=P.Vnum(Id,"LIGHT-DISP-PW");
     if(P.Vnum(Id,"LAMP-BALLAST-EFF-DL")!=0.0) {
        mLIGHT_DISPLAY_POWER*=D.Value("LIGHT_5")/P.Vnum(Id,"LAMP-
BALLAST-EFF-DL");

        //store power for display lighting:
        {
          ostringstream res; res << mLIGHT DISPLAY POWER;
          P.ModKey(Id,"LIGHT-DISP-PW",res.str());
        }
     }
  } // end if

}

void  ZONE::fLIGHT PARASITIC(PROJECTDefaults&  D,  PROJECTTables&  T,
BOC& P) {
  if(func_report) cout << endl << "  " << "ZONE::fLIGHT_PARASITIC";

  deque<double> mBR LIGHT CHECK=P.Vlnum(Id,"BR-LIGHT-CHECK");

/*
// 1st alternative to reduce Parasitic Power when considering it not
working 24 hours a day
// JO Added 03Oct06
  deque<double> mLIGHT T START=P.Vlnum(Id,"LIGHT-T-START");
  deque<double> mLIGHT_T_END=P.Vlnum(Id,"LIGHT-T-END");
```

```
  double mLIGHT HOURS=0;
  for(int i=0;i<12;i++) {
    mLIGHT HOURS+=(mLIGHT T END[i]-
mLIGHT_T_START[i])*T.Value("NJ",1,i+1);
  }
  double mRED FACTOR=mLIGHT HOURS/8760;
*/

// 2nd alternative to reduce Parasitic Power when considering it not
working 24 hours a day
// JO Added 03Oct06
//     deque<double>   mHOURS=P.Vlnum(Id,"SYS-T-HOURS");   //   TO   BE
CORRECTED TO LIGHT-T-HOURS when available
//  double mLIGHT HOURS=0;
//  for(int i=0;i<12;i++) {
//    mLIGHT_HOURS+=mHOURS[i];
//  }
//  double mRED FACTOR=mLIGHT HOURS/8760;
//  if(mRED_FACTOR<=0 || mRED_FACTOR>1) mRED_FACTOR=1.0;

// !! The above lines have been commented out in order that the
parasitic power is assumed to be on all the time,
// !! rather than following the lighting operating hours. This is in
response to Aecom's email on 22/05/2010.
// !! - AN(24/05/2010)
  double mRED FACTOR=1.0;

//cout << "light-control: " << P.Vcod(Id,"LIGHT-CONTROL") << endl;
//cout << mPARASITIC POWER << endl; system("PAUSE");

  double mPARASITIC POWER=P.Vnum(Id,"LIGHT-PARAS-PW");

  if(mPARASITIC POWER==-6666)
    if((P.Vcodid(Id,"LIGHT-CONTROL")==2)    ||    (P.Vcodid(Id,"LIGHT-
CONTROL")==3))
      mPARASITIC POWER=D.Value("dParasiticPower");
    else
      mPARASITIC_POWER=0.0;
  else
    if((P.Vcodid(Id,"LIGHT-CONTROL")!=2)    &&    (P.Vcodid(Id,"LIGHT-
CONTROL")!=3))
      if(mPARASITIC POWER!=0) {
        mPARASITIC POWER=0.0;
        {  ostringstream  logtext;  logtext  <<  endl  <<  "WARNING:
Parasitic power for photoelectric device in " <<
          "zone  without  photoelectric  control...  Please  check.
Parasitic power in this zone has been set to 0" << endl;
          SendLog(logtext.str()); }
      }

// Adding parasitic power for occupancy sensing - AN - 29/02/08 -
Modified 20/06/08
  //if (it is a meeting room) OR (its area is < 30m2) AND (there is
occupancy sensing)
  if             (             ((mBR LIGHT CHECK[2]==1)             ||
(P.Vnum(Id,"AREA")<D.Value("LIGHT_CEN_AREA")))                        &&
(P.Vcodid(Id,"LIGHT-OCC-SENS-T")!=6) ) // 6 = NONE
     // if it is NOT one of the restricted activity areas
     if (mBR_LIGHT_CHECK[1]==0)
//        mPARASITIC POWER+=D.Value("dParasiticPowerOcc");
        mPARASITIC POWER+=P.Vnum(Id,"LIGHT-PARAS-PW-OCC");     //  new
keyword for parasitic power of occupancy sensing
     // else it IS one of the restricted activity areas
     else     if     (     (P.Vcodid(Id,"LIGHT-OCC-SENS-T")==2)     ||
(P.Vcodid(Id,"LIGHT-OCC-SENS-T")==3) )   // 2 = Auto on/off, 3 = Auto
on/dimmed
//             mPARASITIC POWER+=D.Value("dParasiticPowerOcc");
            mPARASITIC_POWER+=P.Vnum(Id,"LIGHT-PARAS-PW-OCC");     //
```

```
new keyword for parasitic power of occupancy sensing

// Outside the above conditions, NO benefit from occupancy sensing is
considered and NO parasitic power is added for it

  {
   ostringstream res; res << mPARASITIC POWER*mRED FACTOR;
   P.ModKey(Id,"LIGHT-PARAS-PW",res.str());
  }

}

void ZONE::fLIGHT FOD(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "  " << "ZONE::fLIGHT_FOD";
  double mLIGHT FOD=1.0;
  double mDISP HOFF=0.0;

//  switch(P.Vcodid(Id,"LIGHT-TIME-SWITCH")) {
//       //an automatic  time switch switch a fraction, f of the
display lighting off for a no of hours per day
//     case 1: {
//        mDISP HOFF=P.Vnum(Id,"LIGHT-DISP-HOFF");
//                              mLIGHT FOD=(((1.0-P.Vnum(Id,"LIGHT-DISP-
F"))*mDISP HOFF/24.0)+(24.0-mDISP HOFF)/24.0);
//     }
//     break;
//       //an automatic time switch dim the display lighting  to a
fraction of its total illuminance for a no of hours per day
//     case 2:  {
//        mDISP HOFF=P.Vnum(Id,"LIGHT-DISP-HOFF");
//        if((1.0-D.Value("LIGHT RF"))!=0.0)
//                        mLIGHT FOD=((1.0-P.Vnum(Id,"LIGHT-DISP-F")*(1-
D.Value("LIGHT RW"))/(1-D.Value("LIGHT RF"))
)*mDISP HOFF/24.0)+(24.0-mDISP HOFF)/24.0;
//        else
//           mLIGHT FOD=-1.0;
//     }
//     break;
//  }

  // If there is time switching for the display lighting
  if(P.Vcodid(Id,"LIGHT-TIME-SWITCH")==1) // 1=YES
    mLIGHT FOD=1.0-D.Value("dDISP LIGHT SWITCH");


   //store display lighting occupancy correction factor for zone:
  {
    ostringstream res; res << mLIGHT_FOD;
    P.ModKey(Id,"LIGHT-FOD",res.str());
  }
}


void ZONE::fIFRBCO2(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "ZONE::fIFRBCO2";
//calculation of KG CO2 for zone (used to calculate IF)

  //improvement  factor  (IF)  and  LZC  benchmark  depends  on  building
services strategy - ADL2 p.14
//SERVICE-STRATEGY categories are:
          // 1)Heated and Naturally Ventilated
          // 2)Heated and Mechanically Ventilated
          // 3)Air Conditioned
          // 4)No heating or cooling
          // From 01 Jan 08
          // 5)Mixed-mode and Naturally Ventilated
          // 6)Mixed-mode and Mechanically Ventilated
//Heated and Mechanically Ventilated category includes heating only
systems with any of:
```

```
          //- HVAC level ventilation
          //- zone level mechanical supply
          //- zones with extract systems serving more than one space
  int system1=(int)P.Vnum(Id,"SERVICE-STRATEGY");

  //for systems with no heating or cooling set type to 1
      //the same IF and RB is applied in this case as for heating and
natural ventilation systems
//  if (system==4) system=1;
// 01 Jan 08 Now this correction 4 to 1 is not needed as the tables
tIF and tRB have
// been updated to include values for configuration 4: No heating or
cooling,
// 5: mixed-mode with natural ventilation and 6: mixed-mode with
mechanical ventilation

  double mIF=T.Value("tIF",system1,P.Vint(Id,"NOT-CASE"));
  double mRB=T.Value("tRB",system1,P.Vint(Id,"NOT-CASE"));

// total KGCO2:
  double mKGCO2=P.Vnum(Id,"KG-CO2");

  double mIFCO2=mIF*mKGCO2;
  double mRBCO2=mRB*mKGCO2;

  {
    ostringstream res; res << mIFCO2;
    P.ModKey(Id,"IFCO2",res.str());
  }
  {
    ostringstream res; res << mRBCO2;
    P.ModKey(Id,"RBCO2",res.str());
  }
}


void ZONE::fQ AUX(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func report) cout << endl << "  " << "ZONE::fQ AUX";
//this function calculates auxiliary energy for the zone

  double mAUX M[12]= {0,0,0,0,0,0,0,0,0,0,0,0};
//  double mAUXNOT_M[12]= {0,0,0,0,0,0,0,0,0,0,0,0};
  double mFlag Corr[12]= {0,0,0,0,0,0,0,0,0,0,0,0};
  double mAREA=P.Vnum(Id,"AREA");
  double tAEnergy=0.0;
  deque<double> mSYS T HOURS;
  deque<double> mOA FLOW PERS, mOA FLOW PERS DES;

  if (T.Value("tSYS COOL",1,P.Vcodid(pId,"TYPE"))==1)
    mSYS T HOURS=P.Vlnum(Id,"SYS-T-HOURS");
  else
    mSYS T HOURS=P.Vlnum(Id,"SYS-HEAT-T-HOURS");

  double mQ_V_M_EXH;
  double mSFP EXH=0;
  double mSFP=0;

  double mSys Corr=1.0;      // correction factor due to leakage and
controls

  bool flag ACT;
  if (P.Vcod(gId,"ACT-NOT")=="ACT") {
    flag_ACT=1;

    mSys Corr=P.Vnum(pId,"AUX-ENERGY-CORR");  //  adjustments  due  to
duct leakage, controls, etc.

    // Retrieve from the tables the type of pump power associated
with the type of the parent HVAC
```

```
    // 0 = pump power not applicable
    // 1 = pump power applicable to LTHW only
    // 2 = pump power applicable to both LTHW and CHW
    int mPump type=T.Vint("tPump type",1,P.Vcodid(pId,"TYPE"));
    // If variable speed pumping is not applicable to the parent HVAC
system, i.e., constant speed
    if (P.Vcodid(pId,"VARIABLE-SPEED-PUMP")==1)  // 1=NO
      // Note 1 is added to the value in mPump type to make up the
column number
      tAEnergy=T.Value("tVRP Type",4,mPump type+1);           // Note
tAEnergy is in W/m2
    else // i.e., variable speed pumping is applicable
      // Note tAEnergy is in W/m2
      tAEnergy=T.Value("tVRP_Type",P.Vcodid(pId,"VARIABLE-SPEED-PUMP-
TYPE"),P.Vcodid(pId,"VARIABLE-SPEED-PUMP"));
  } // end if
  else { // i.e., this is the notional building
    flag ACT=0;
    // For the 2010 notional building in E, W, & NI only
    if (P.Vint(Id,"NOT-CASE")>=9) {
      // !!! Do not include any pump power in the E&W 2010 notional
building if this is a top-lit zone with
      // !!! no cooling and where the notional building has been
determined to have radiant heating
      // !!! ( which has no pumps)
      // Check if the following conditions are NOT fulfilled:
      // This is a top-lit space and the parent HVAC system provides
heating only (i.e., no cooling),
      // the heat source of the parent HVAC system is NOT a heat pump
and is NOT district heating, and its
      // fuel is NOT biogas, biomass, or waste heat
      // 11=EW NOT 2010 Top Lit , 0=no cooling
      //  18=Heat  pump  (electric):  air  source  ,  19=Heat  pump
(electric): ground or water source
      // 12=Heat pump (gas/oil): air source , 13=Heat pump (gas/oil):
ground or water source
      // 15=District Heating ,  3=Biogas, 9=Biomass, 12=Waste heat
      if       (       !(       (P.Vint(Id,"NOT-CASE")==11)       &&
(T.Value("tSYS_COOL",1,P.Vcodid(pId,"TYPE"))==0)
                &&         (P.Vcodid(pId,"HEAT-SOURCE")!=18)         &&
(P.Vcodid(pId,"HEAT-SOURCE")!=19)
                &&         (P.Vcodid(pId,"HEAT-SOURCE")!=12)         &&
(P.Vcodid(pId,"HEAT-SOURCE")!=13)
                &&         (P.Vcodid(pId,"HEAT-SOURCE")!=15)         &&
(P.Vcodid(pId,"FUEL-TYPE")!=3)
                &&          (P.Vcodid(pId,"FUEL-TYPE")!=9)           &&
(P.Vcodid(pId,"FUEL-TYPE")!=12) ) )
       tAEnergy=T.Value("tNot_Pump_power",1,P.Vcodid(pId,"TYPE"));
    } // end if
  } // end else

//SERVICE-STRATEGY categories are:
          // 1)Heated and Naturally Ventilated
          // 2)Heated and Mechanically Ventilated
          // 3)Air Conditioned
          // 4)No heating or cooling
          // From 01 Jan 08
          // 5)Mixed-mode and Naturally Ventilated
          // 6)Mixed-mode and Mechanically Ventilated
//Heated and Mechanically Ventilated category includes heating only
systems with any of:
          //- HVAC level ventilation
          //- zone level mechanical supply
          //- zones with extract systems serving more than one space
  int mSERVICE STRATEGY=(int)P.Vnum(Id,"SERVICE-STRATEGY");
  if (mSERVICE_STRATEGY==5) mSERVICE_STRATEGY=1;
  if (mSERVICE STRATEGY==6) mSERVICE STRATEGY=2;
  switch(mSERVICE STRATEGY) {
    //Heated and Nat. Ventilated:
```

```
    case 1: {
              //actual building:
              if(flag ACT) {
                //check if system has auxiliary energy associated with
it:
                if (T.Value("tSYS AUX",1,P.Vcodid(pId,"TYPE"))==1)  {
                  for(int i=0;i<12;i++)
                    mAUX_M[i]=(tAEnergy/1000)*mSYS_T_HOURS[i];
//kWh/m2
                }
                //if  there  is  mechanical  exhaust  -  add  exhaust
auxiliary energy to monthly auxiliary energy
                //use  the  value  of  SFP  entered  by  user  for  exhaust
ventilation
                //use the exhaust flow rate entered by user, Q-V-M-EXH
                if (P.Vcodid(Id,"VENT-MECH-EXH")==1) {
                  mQ_V_M_EXH=P.Vnum(Id,"Q-V-M-EXH");  //l/s.m2
                  mSFP_EXH=P.Vnum(Id,"VENT-SFP-EXH"); //W/(l/s)
                  for(int i=0;i<12;i++)

mAUX_M[i]+=mQ_V_M_EXH*mSFP_EXH*mSYS_T_HOURS[i]/1000; //kWh/m2
                }
              } // end if
              //notional building:
              else {
                // For the 2010 notional building in E, W, & NI only
                if (P.Vint(Id,"NOT-CASE")>=9) {
                  for(int i=0;i<12;i++)
                    mAUX_M[i]=(tAEnergy/1000)*mSYS_T_HOURS[i];
//kWh/m2
                } // end if
                else { // i.e., other notional buildings
                  for(int i=0;i<12;i++)
                   //  !! The table "Not HVAC Eff" now contains the
value in W/m2 rather than kWh/m2
                    mAUX_M[i]=(T.Value("Not HVAC Eff",1+5*(3-
1),P.Vint(gId,"NOT-CASE"))/1000)*mSYS_T_HOURS[i]; //kWh/m2
                } // end else

                //if  there  is  mechanical  exhaust  -  add  exhaust
auxiliary energy to monthly auxiliary energy
                //use the exhaust flow rate entered by user, Q-V-M-EXH
-  !!!!  DO  NOT  consider  for  Reference  building  -  Remember  service
strategy 5 is set to = 1 above
                if                                    ((P.Vcodid(Id,"VENT-MECH-
EXH")==1)&&(P.Vcod(gId,"NOT-TYPE")!="REF")) {
                  mQ_V_M_EXH=P.Vnum(Id,"Q-V-M-EXH");
                  for(int i=0;i<12;i++)
                    mAUX_M[i]+=mQ_V_M_EXH*P.Vnum(Id,"VENT-SFP-
EXH")*mSYS_T_HOURS[i]/1000; //kWh/m2
                }
              } // end else
             }
             break;
    //Heated and Mech. Ventilated:
    case 2: {
              //actual building:
              if(flag ACT) {
                //check if system has auxiliary energy associated with
it:
                if (T.Value("tSYS AUX",1,P.Vcodid(pId,"TYPE"))==1)  {
                  for(int i=0;i<12;i++)
                    mAUX_M[i]=(tAEnergy/1000)*mSYS_T_HOURS[i];
//kWh/m2
                }

                //if  there  is  mechanical  exhaust  -  add  exhaust
auxiliary energy to monthly auxiliary energy
                //use  the  value  of  SFP  entered  by  user  for  exhaust
```

```
ventilation:
                //use the exhaust flow rate entered by user, Q-V-M-EXH
                if (P.Vcodid(Id,"VENT-MECH-EXH")==1) {
                  mQ V M EXH=P.Vnum(Id,"Q-V-M-EXH");  //l/s.m2
                  mSFP_EXH=P.Vnum(Id,"VENT-SFP-EXH"); //W/(l/s)
                  for(int i=0;i<12;i++)

mAUX_M[i]+=mQ_V_M_EXH*mSFP_EXH*mSYS_T_HOURS[i]/1000; //kWh/m2
                }
              } // end if
              //notional building:
              else {
                // For the 2010 notional building in E, W, & NI only
                if (P.Vint(Id,"NOT-CASE")>=9) {
                  for(int i=0;i<12;i++)
                    mAUX M[i]=(tAEnergy/1000)*mSYS T HOURS[i];
//kWh/m2
                } // end if
                else { // i.e., other notional buildings
                  for(int i=0;i<12;i++)
                    // !! The table "Not HVAC Eff" now contains the
value in W/m2 rather than kWh/m2
                    mAUX M[i]=(T.Value("Not HVAC Eff",1+5*(3-
1),P.Vint(gId,"NOT-CASE"))/1000)*mSYS T HOURS[i]; //kWh/m2
                } // end else

                //if there is mechanical exhaust - add exhaust
auxiliary energy to monthly auxiliary energy
                //use the exhaust flow rate entered by user, Q-V-M-EXH
                if (P.Vcodid(Id,"VENT-MECH-EXH")==1) {
                  mQ V M EXH=P.Vnum(Id,"Q-V-M-EXH");
                  for(int i=0;i<12;i++)
                    // Note that the value in VENT-SFP-EXH for the
notional building has already been retrieved from
                    // the Table tNOT SFP in ZONE::Notional
                    mAUX M[i]+=mQ V M EXH*P.Vnum(Id,"VENT-SFP-
EXH")*mSYS T HOURS[i]/1000; //kWh/m2
                }
              } // end else
            }
            break;
    //Air Conditioned (heated and cooled):
    case 3: {
              //actual building:
              if(flag_ACT) {
                //check if system has auxiliary energy associated with
it:
                if (T.Value("tSYS_AUX",1,P.Vcodid(pId,"TYPE"))==1)  {
                  for(int i=0;i<12;i++)
                    mAUX M[i]=(tAEnergy/1000)*mSYS T HOURS[i];
//kWh/m2
                }
                //if there is mechanical exhaust - add exhaust
auxiliary energy to monthly auxiliary energy
                //use the value of SFP entered by user for exhaust
ventilation:
                //use the exhaust flow rate entered by user, Q-V-M-EXH
                if (P.Vcodid(Id,"VENT-MECH-EXH")==1) {
                  mQ V M EXH=P.Vnum(Id,"Q-V-M-EXH");  //l/sm2
                  mSFP_EXH=P.Vnum(Id,"VENT-SFP-EXH"); //W/(l/s)
                  for(int i=0;i<12;i++)

mAUX_M[i]+=mQ_V_M_EXH*mSFP_EXH*mSYS_T_HOURS[i]/1000; //kWh/m2
                }
              } // end if
              //notional building:
              else {
                // JO: 01 Jan 08 SERVICE-STRATEGY will be always 5 for
the Reference building, so
```

```
                // the Reference building will not have ever this
correction applied but that below
                // depending on the ventilation strategy: case 5 .

                // For the 2010 notional building in E, W, & NI only
                if (P.Vint(Id,"NOT-CASE")>=9) {
                  for(int i=0;i<12;i++)
                    mAUX_M[i]=(tAEnergy/1000)*mSYS_T_HOURS[i];
//kWh/m2
                } // end if
                else { // i.e., other notional buildings
                  for(int i=0;i<12;i++)
                    // !! The table "Not HVAC Eff" now contains the
value in W/m2 rather than kWh/m2
                    mAUX M[i]=(T.Value("Not HVAC Eff",2+5*(3-
1),P.Vint(gId,"NOT-CASE"))/1000)*mSYS T HOURS[i];
                }

                //if there is mechanical exhaust - add exhaust
auxiliary energy to monthly auxiliary energy
                //use the exhaust flow rate entered by user, Q-V-M-EXH
                if (P.Vcodid(Id,"VENT-MECH-EXH")==1) {
                  mQ V M EXH=P.Vnum(Id,"Q-V-M-EXH");
                  for(int i=0;i<12;i++)
                    mAUX M[i]+=mQ V M EXH*P.Vnum(Id,"VENT-SFP-
EXH")*mSYS T HOURS[i]/1000; //kWh/m2
                }
              } // end else
            }
            break;


    //zones with no heating and cooling
    case 4: {
             //zones with no heating and cooling - for actual
building treat as heating and nat vent:
             if(flag ACT) {
               //check if system has auxiliary energy associated with
it:
               if (T.Value("tSYS AUX",1,P.Vcodid(pId,"TYPE"))==1)  {
                 for(int i=0;i<12;i++)
                   mAUX_M[i]=(tAEnergy/1000)*mSYS_T_HOURS[i];
//kWh/m2
               }

               //if there is mechanical exhaust - add exhaust
auxiliary energy to monthly auxiliary energy
               //use the value of SFP entered by user for exhaust
ventilation
               //use the exhaust flow rate entered by user, Q-V-M-EXH
               if (P.Vcodid(Id,"VENT-MECH-EXH")==1) {
                 mQ V M EXH=P.Vnum(Id,"Q-V-M-EXH");  //l/s/m2
                 mSFP EXH=P.Vnum(Id,"VENT-SFP-EXH"); //W/l/s
                 for(int i=0;i<12;i++)

mAUX M[i]+=mQ V M EXH*mSFP EXH*mSYS T HOURS[i]/1000; //kWh/m2
               }
             } // end if
             //notional building:
             else {
               for(int i=0;i<12;i++)  {
                 //notional auxiliary energy = 0:
                   mAUX_M[i]=0; //kWh/m2
               }

               //if there is mechanical exhaust - add exhaust
auxiliary energy to monthly auxiliary energy
               //use the exhaust flow rate entered by user, Q-V-M-EXH
               if (P.Vcodid(Id,"VENT-MECH-EXH")==1) {
```

```
                  mQ V M EXH=P.Vnum(Id,"Q-V-M-EXH");
                  for(int i=0;i<12;i++)
                    mAUX M[i]+=mQ V M EXH*P.Vnum(Id,"VENT-SFP-
EXH")*mSYS T HOURS[i]/1000; //kWh/m2
                    }
                } // end else
              }
              break;
    //Mixed-mode with Natural ventilation
    case 5: {

              }
              break;
    //Mixed-mode with Mechanical Ventilation
    case 6: {

              }
              break;

    default: {
              }
              break;
  }

  // Auxiliary energy due to pumps and exhaust fans only - to be
added in fAUX ENERGY to auxiliary energy due to mech vent fans
  {
   ostringstream res; res << "{ ";
   for (int i=0;i<11;i++)
     res << mAUX M[i] << ", ";
   res << mAUX M[11] << " }";
   P.ModKey(Id,"KWH-PUMP/EXH-AUX-M",res.str());   //KWH/m2
  }

  // Save the correction factor to be applied later in fAUX ENERGY
  {
   ostringstream res; res << mSys Corr;
   P.ModKey(Id,"SYSTEM-CORRECTION",res.str());
  }

}


int ZONE::FuelNot2010(int HS, int FT) {
  // 18=Heat pump (electric): air source , 19=Heat pump (electric):
ground or water source
  // 10=Grid Supplied Electricity
  if ( ((HS==18) || (HS==19)) && (FT==10) )
    return(FT);   // same as heating fuel in the actual building

  else { // i.e., not an electric heat pump
    // Check if the heating fuel in the actual building is one whose
emission factor is >= that of oil,
    // i.e., >= 0.306 kg/kWh
    //  4=Oil,  5=Coal,  6=Anthracite,  7=Smokeless  Fuel  (inc  Coke),
10=Grid Supplied Electricity
    if ( (FT==4) || (FT==5) || (FT==6) || (FT==7) || (FT==10) )
      return(4);

    else // i.e., the heating fuel in the actual building is one
whose emission factor < that of oil,
       // i.e., < 0.306 kg/kWh
      return(FT);   // same as heating fuel in the actual building
  } // end else
}


double ZONE::fHSSeff(PROJECTDefaults& D, PROJECTTables& T, BOC& P,
string HS, string FT, double BH_LOAD, int pT, int gId, int Id) {
```

```
   int HS Id=HeatSourceId(HS);
   int FT Id=FuelTypeId(FT);

   int FTnoti=FuelNot2010(HS Id, FT Id);

   double tHSSeff=0.0;
   //--- Define efficiency
   // check if the heat source in the actual building is provided by
an electric heat pump
   // 18=Heat pump (electric): air source , 19=Heat pump (electric):
ground or water source
   // 10=Grid Supplied Electricity
   if ( ((HS Id==18) || (HS Id==19)) && (FT Id==10) )
     tHSSeff=T.Value("Not_Space_Heat_SEff_4",1,P.Vint(Id,"NOT-
CASE"));

   else { // i.e., not an electric heat pump
     // check if the heat source in the actual building is a non-
electric heat pump
     // 12=Heat pump (gas/oil): air source , 13=Heat pump (gas/oil):
ground or water source
     if ((HS Id==12) || (HS Id==13))
       tHSSeff=T.Value("Not Space Heat SEff 3",1,P.Vint(Id,"NOT-
CASE"));

     else { // i.e., heat source is not a heat pump
       // Check if the heating fuel and heat source in the actual
building is district heating
       // 13=District Heating , 15=District Heating
       if ((FT Id==13) && (HS Id==15) ) {
         // Heating efficiency depends on the emission factor input
by the user for district heating
         if ((P.Vnum(gId,"DH-CO2F"))<D.Value("dNOT DISTRICT HEAT1"))
           tHSSeff=T.Value("Not Space Heat SEff 1",1,9);
           else {
             if                                          ((P.Vnum(gId,"DH-
CO2F"))>D.Value("dNOT DISTRICT HEAT2"))
               tHSSeff=T.Value("Not Space Heat SEff 2",1,9);
             else // i.e., 0.15 <= emission factor <= 0.2
               tHSSeff=(((P.Vnum(gId,"DH-CO2F"))-
D.Value("dNOT DISTRICT HEAT1"))*(T.Value("Not Space Heat SEff 2",1,9)
-
T.Value("Not Space Heat SEff 1",1,9))/(D.Value("dNOT DISTRICT HEAT2")
-
D.Value("dNOT_DISTRICT_HEAT1")))+T.Value("Not_Space_Heat_SEff_1",1,9)
;
           } // end else
       } // end if

       else { // i.e., fuel type is not district heating
         // Check if the main heating fuel in the actual building is
one whose emission factor <= 0.156 kg/kWh
         // 3=Biogas, 9=Biomass, 12=Waste heat
         if ( (FT_Id==3) || (FT_Id==9) || (FT_Id==12) )
           tHSSeff=T.Value("Not Space Heat SEff 1",1,P.Vint(Id,"NOT-
CASE"));
         else { // i.e., the main heating fuel in the actual building
is one whose emission factor > 0.156 kg/kWh
           // If this is a top-lit space and the parent HVAC system
provides heating only (i.e., no cooling)
           // 11=EW NOT 2010 Top Lit
           if         (         (P.Vint(Id,"NOT-CASE")==11)         &&
(T.Value("tSYS_COOL",1,pT)==0) )

tHSSeff=T.Value("Not Space Heat SEff 5",1,P.Vint(Id,"NOT-CASE"));
           else // not a top-lit space with heating only

tHSSeff=T.Value("Not Space Heat SEff 2",1,P.Vint(Id,"NOT-CASE"));
         } // end else
```

```
      } // end else
    } // end else

  } // end else

  return(BH LOAD*T.Value("tKGCO2",1,FTnoti)/tHSSeff);

}

//v2.0.a function moved from HVAC to zone level:
void ZONE::fQ_PRIM_HEAT(PROJECTDefaults& D, PROJECTTables& T, BOC& P)
{
  if(func report) cout << endl << "  " << "ZONE::fQ PRIM HEAT";
//calculates primary energy required for heating for this ZONE

 double mQ DEM HEAT=0.0;
 double mQ_PRIM_HEAT_CHP=0.0;
 deque<double> mQ DEM HEAT M;
//  double  mQ DEM HEAT CORR M[12]= { 0.0,  0.0,  0.0,  0.0,  0.0,  0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0 };
 double mQ PRIM HEAT CHP M[12]= { 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0 };

 //heat demand for zone:
 deque<double> mQ DEM HEAT M Z;

//correction factor for the NOTIONAL building HEAT-SSEEF
 //correction factor is different to 1 if you have heating only and
mechanical ventilation, notional building only - actual building to
have value 1,1
// double mNOT EFF MECH CORR;

  //heat demand for zone for each month (12 values):
  mQ DEM HEAT M=P.Vlnum(Id,"Q-DEM-HEAT-M-ROOM");

  //correction factor for zone:
//  mNOT EFF MECH CORR=P.Vnum(Id,"NOT-EFF-MECH-CORR");

 // Check the heating fuel of the parent HVAC system in the actual
building if the calculation is for Scotland in order
 // to trigger the adjustment to the heating demand of the notional
building if it is grid-supplied electricity. This
 // adjustment is intended to maintain the 2006 relationship of this
fuel to the oil and gas baselines.
 // If this is the 2010 notional building for Scotland
 if ( (P.Vcod(gId,"ACT-NOT")=="NOT") && (P.Vint(gId,"NOT-CASE")==4) )
{
   if (P.Vcodid(pId,"ACT-ELEC-HEAT")==1) { // 1=YES
     // Check the notional fuel
     if (P.Vcodid(gId,"NOT-FUEL")==2)  // 2=GAS
       for (int i=0;i<12;i++)
         mQ DEM HEAT M[i]*=D.Value("dNOT HEAT GAS");
     else // i.e., NOT-FUEL=OIL
       for (int i=0;i<12;i++)
         mQ DEM HEAT M[i]*=D.Value("dNOT HEAT OIL");
   } // end if
 } // end if


  //for each month of year:
  for(int m=0;m<12;m++) {
    //add zone demand to annual zone demand:
    mQ_DEM_HEAT+=mQ_DEM_HEAT_M[m];
    //add demand to zone demand for that month:
//    mQ DEM HEAT CORR M[m]=mQ DEM HEAT M[m]*mNOT EFF MECH CORR;
  }

 //if parent HVAC system has CHP, reduce the heat demand by this
amount:
```

```
 if (P.Vcod(pId,"CHP")=="YES") {
    //fraction of building heating demand supplied by the CHP system:
    double   mBUILD CHP SHEAT=P.Vnum(P.Vint(gId,"CHP-POS"),"BUILD-CHP-
SHEAT");
    //CHP thermal seasonal efficiency:
    double mCHP HEAT SEER=P.Vnum(P.Vint(gId,"CHP-POS"),"HEAT-SEER");
    //monthly heating energy supplied by CHP
    //CHP may serve more than 1 HVAC system so here we take KWH-HEAT-
M from CHP-generator and add to it
    deque<double>  mKWH CHP M=P.Vlnum(P.Vint(gId,"CHP-POS"),"KWH-HEAT-
M");
    //annual heating energy supplied by CHP
    double mKWH CHP=0;

    if(mCHP HEAT SEER!=0) {
      //for each month of year
      for (int m=0;m<12;m++) {
        //monthly primary energy for CHP

mQ_PRIM_HEAT_CHP_M[m]=mBUILD_CHP_SHEAT*mQ_DEM_HEAT_M[m]/mCHP_HEAT_SEE
R;
        //convert from MJ to kWh:

mKWH CHP M[m]+=(mQ PRIM HEAT CHP M[m]/3.6)*P.Vnum(Id,"MULTIPLIER");
        //add to annual CHP energy
        mKWH CHP+=mKWH CHP M[m];

        mQ DEM HEAT M[m]*=(1-mBUILD CHP SHEAT);
//        mQ DEM HEAT CORR M[m]*=(1-mBUILD CHP SHEAT);
      }
      mQ DEM HEAT*=(1-mBUILD CHP SHEAT);
      //store heating energy for CHP system in kWh, monthly and
annual:
      // 10 April 2010 JO: Shell and Core new approach for Part L
2010 implies CHP should not be considered when Shell and Core is
applicable...
      if(!(P.Vcodid(cId,"SHELL-AND-CORE")==1   &&   P.Vcodid(cId,"EPC-
TYPE")==1   &&   P.Vcodid(cId,"BR-STAGE")==2   &&   P.Vcodid(Id,"SHELL-
ASSUMED")==1)) {
        {
         ostringstream res; res << "{ ";
         for (int i=0;i<11;i++)
           res << mKWH CHP M[i] << ", ";
         res << mKWH CHP M[11] << " }";
         P.ModKey(P.Vint(gId,"CHP-POS"),"KWH-HEAT-M",res.str());
        }
        {
         ostringstream res; res << mKWH_CHP*P.Vnum(Id,"MULTIPLIER");
         P.ModKey(P.Vint(gId,"CHP-POS"),"KWH-HEAT",res.str());
        }
      }
    }
 }

 //store heating demand for ZONE
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ DEM HEAT M[i] << ", ";
  res << mQ_DEM_HEAT_M[11] << " }";
  P.ModKey(Id,"Q-DEM-HEAT-M",res.str());
 }

 {
  ostringstream res; res << mQ DEM HEAT;
  P.ModKey(Id,"Q-DEM-HEAT",res.str());
 }

 //primary energy for heating for ZONE
```

```
 double mQ PRIM HEAT=0.0;
 double mQ PRIM HEAT M[12]={ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0 };
 double tHSSeff=0.0;
 double dHSSeff=0.0;  // partial efficiency
 double tHSPFb=0.0;
 double tHSPFa=0.0;

 // Check if this is the 2010 notional as it affects the heating
efficiency which in this case, depends on the DRIVER2A
 // keyword and the heating fuel of the parent HVAC system in the
actual building
 if ( (P.Vcod(gId,"ACT-NOT")=="NOT") && (P.Vint(Id,"NOT-CASE")>=9) )
{ // 9=EW_NOT_2010
   deque<string> BH HS=P.Vltext(pId,"BH-HEAT-SOURCE");
   deque<string> BH FT=P.Vltext(pId,"BH-FUEL-TYPE");
   deque<double> BH_LOAD=P.Vlnum(pId,"BH-LOAD");

   int Nfuel=BH FT.size();   // number of fuel types
   for(int i=0;i<Nfuel;i++)
     dHSSeff+=fHSSeff(D,   T,   P,   BH HS[i],   BH FT[i],   BH LOAD[i],
P.Vcodid(pId,"TYPE"), gId, Id);

   //  The  efficiency  equivalent  to  the  combination  of  bivalent
systems
   if(dHSSeff!=0)
     tHSSeff=T.Value("tKGCO2",1,P.Vcodid(pId,"FUEL-TYPE"))/dHSSeff;
   else
     tHSSeff=T.Value("tKGCO2",1,P.Vcodid(pId,"FUEL-TYPE"));   //  This
should never happen


   //--- Calculate consumption

   //calculate primary energy for heating by applying efficiency to
the heating demand
   if(tHSSeff>0)
     for(int m=0;m<12;m++) {
       mQ_PRIM_HEAT_M[m]=mQ_DEM_HEAT_M[m]/tHSSeff;
       mQ PRIM HEAT+=mQ PRIM HEAT M[m];
     }
 } // end if

 else { // i.e., not the 2010 notional
   //heating thermal efficiency
   tHSSeff=P.Vnum(pId,"HEAT-SSEFF");

   //calculate primary energy for heating by applying efficiency to
the heating demand (after correction)
   if(tHSSeff!=0)
     for(int m=0;m<12;m++) {
  //     mQ PRIM HEAT M[m]=mQ DEM HEAT CORR M[m]/tHSSeff;
       mQ PRIM HEAT M[m]=mQ DEM HEAT M[m]/tHSSeff;
       mQ_PRIM_HEAT+=mQ_PRIM_HEAT_M[m];
     }
   else {
        {ostringstream os; os << "ERROR: tHSSeff=0.0" << endl;
         SendLog(os.str()); WriteLog();}
     assert(1==0);
   }
 } // end else


// Applying the correction for electric power factor is now done on
ZONE calculated energy
// rather than CO2 so would now be included in mQ-PRIM-HEAT-M and mQ-
PRIM-HEAT - AN 18/01/08
 if ((P.Vcodid(pId,"FUEL-TYPE")==10)) { // If the fuel of the parent
HVAC is grid-electricity and this is the actual building
```

```
   for(int m=0;m<12;m++)
     mQ PRIM HEAT M[m]*=(1-
T.Value("tPOWER FACTOR",1,P.Vcodid(gId,"ELEC-POWER-FACTOR")));
   mQ PRIM HEAT*=(1-T.Value("tPOWER FACTOR",1,P.Vcodid(gId,"ELEC-
POWER-FACTOR")));
 }

 //store primary energy for heating for ZONE, monthly and annual
figures in MJ and kWh
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ PRIM HEAT M[i] << ", ";
  res << mQ_PRIM_HEAT_M[11] << " }";
  P.ModKey(Id,"Q-PRIM-HEAT-M",res.str());
 }
 {
  ostringstream res; res << mQ PRIM HEAT;
  P.ModKey(Id,"Q-PRIM-HEAT",res.str());
 }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ PRIM HEAT M[i]/3.6 << ", ";
  res << mQ PRIM HEAT M[11]/3.6 << " }";
  P.ModKey(Id,"KWH-HEAT-M",res.str());
 }
 {
  ostringstream res; res << mQ PRIM HEAT/3.6;
  P.ModKey(Id,"KWH-HEAT",res.str());
 }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ PRIM HEAT CHP M[i] << ", ";
  res << mQ PRIM HEAT CHP M[11] << " }";
  P.ModKey(Id,"Q-PRIM-HEAT-CHP-M",res.str());
 }
 {
  ostringstream res; res << mQ_PRIM_HEAT_CHP;
  P.ModKey(Id,"Q-PRIM-HEAT-CHP",res.str());
 }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ PRIM HEAT CHP M[i]/3.6 << ", ";
  res << mQ PRIM HEAT CHP M[11]/3.6 << " }";
  P.ModKey(Id,"KWH-HEAT-CHP-M",res.str());
 }
 {
  ostringstream res; res << mQ_PRIM_HEAT_CHP/3.6;
  P.ModKey(Id,"KWH-HEAT-CHP",res.str());
 }
}

//v2.0.a function moved from HVAC to zone level:
void ZONE::fQ PRIM COOL(PROJECTDefaults& D, PROJECTTables& T, BOC& P)
{
  if(func report) cout << endl << "  " << "ZONE::fQ PRIM COOL";
//calculates primary energy required for cooling for this ZONE

  double mQ DEM COOL=0;
  double mQ_DEM_COOL_M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
  deque <double> mQ DEM COOL M ROOM;
  double mQ PRIM HEATCOOL CHP=0.0;
  double mQ_PRIM_HEATCOOL_CHP_M[12]= { 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
```

```
0.0, 0.0, 0.0, 0.0, 0.0, 0.0 };

//we are now calculating "Q-DEM-COOL-M-ROOM" where there is no
cooling
//if there is no cooling "Q-DEM-COOL-M" is set to zero below
//  if (T.Value("tSYS COOL",1,P.Vcodid(pId,"TYPE"))==1)  {
  //cooling for this zone
//                        mQ_DEM_COOL_M=P.Vlnum_Mul(Id,"Q-DEM-COOL-M-
ROOM",P.Vnum(Id,"MULTIPLIER")); //ZONE MULTIPLIER APPLIED HERE

    mQ_DEM_COOL_M_ROOM=P.Vlnum(Id,"Q-DEM-COOL-M-ROOM");

    // If this is the 2010 notional building for Scotland
    if   (   (P.Vcod(gId,"ACT-NOT")=="NOT")   &&   (P.Vint(gId,"NOT-
CASE")==4) )
      for (int i=0;i<12;i++)
        mQ_DEM_COOL_M_ROOM[i]*=D.Value("dNOT_COOL1");

  //add monthly values to HVAC cooling demand, monthly and annual:
      for(int m=0;m<12;m++) {
        mQ DEM COOL M[m]=mQ DEM COOL M ROOM[m];
        mQ DEM COOL+=mQ DEM COOL M[m];
      }
 // }

 //if parent HVAC system has CHP, reduce the heat demand by this
amount:
 if                    (P.Vcod(pId,"CHP")=="YES"                    &&
T.Vint("tSYS COOL",1,P.Vcodid(pId,"TYPE"))) {
  if (P.Vcod(P.Vint(gId,"CHP-POS"),"TRIGENERATION")=="YES" ) {
    //fraction of building cooling demand supplied by the CCHP
system:
    double   mBUILD CHP SCOOL=P.Vnum(P.Vint(gId,"CHP-POS"),"BUILD-CHP-
SCOOL");
    //CHP thermal seasonal efficiency:
    double mCHP HEAT SEER=P.Vnum(P.Vint(gId,"CHP-POS"),"HEAT-SEER");
    //CCHP energy seasonal efficiency ratio (coeficient of
performance):
    double mCHP_COOL_SEER=P.Vnum(P.Vint(gId,"CHP-POS"),"COOL-SEER");
    //monthly cooling energy supplied by CCHP
    //CCHP may serve more than 1 HVAC system so here we take KWH-
HEATCOOL-M from CCHP-generator and add to it
    deque<double>      mKWH CHP M=P.Vlnum(P.Vint(gId,"CHP-POS"),"KWH-
HEATCOOL-M");
    //annual heating energy supplied by CHP
    double mKWH CHP=0;

    if(mCHP_HEAT_SEER!=0 && mCHP_COOL_SEER!=0) {
      //for each month of year
      for (int m=0;m<12;m++) {
        //monthly primary energy for CHP

mQ PRIM HEATCOOL CHP M[m]=mBUILD CHP SCOOL*mQ DEM COOL M[m]/mCHP HEAT
_SEER/mCHP_COOL_SEER;
        //convert from MJ to kWh:

mKWH_CHP_M[m]+=(mQ_PRIM_HEATCOOL_CHP_M[m]/3.6)*P.Vnum(Id,"MULTIPLIER"
);
        //add to annual CHP energy
        mKWH_CHP+=mKWH_CHP_M[m];
        mQ DEM COOL M[m]*=(1-mBUILD CHP SCOOL);
      }
      mQ_DEM_COOL*=(1-mBUILD_CHP_SCOOL);
      //store cooliing energy for CHP system in kWh, monthly and
annual:
      {
       ostringstream res; res << "{ ";
       for (int i=0;i<11;i++)
         res << mKWH_CHP_M[i] << ", ";
```

```
       res << mKWH_CHP_M[11] << " }";
       P.ModKey(P.Vint(gId,"CHP-POS"),"KWH-HEATCOOL-M",res.str());
      }
      {
       ostringstream res; res << mKWH_CHP*P.Vnum(Id,"MULTIPLIER");
       P.ModKey(P.Vint(gId,"CHP-POS"),"KWH-HEATCOOL",res.str());
      }
    }
  }
 }


 //store cooling demand, monthly and annual:
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ_DEM_COOL_M[i] << ", ";
  res << mQ_DEM_COOL_M[11] << " }";
  P.ModKey(Id,"Q-DEM-COOL-M",res.str());
 }

 {
  ostringstream res; res << mQ_DEM_COOL;
  P.ModKey(Id,"Q-DEM-COOL",res.str());
 }

 double mQ_PRIM_COOL=0;
 double mQ_PRIM_COOL_M[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
 double tCSSEER=0;

 // Check if this is the 2010 notional as it affects the cooling
efficiency which in this case, depends on the DRIVER2A keyword
 if ( (P.Vcod(gId,"ACT-NOT")=="NOT") && (P.Vint(Id,"NOT-CASE")>=9) )
{ // 9=EW NOT 2010
   // if the parent HVAC system operates mixed-mode cooling
   if (P.Vcod(pId,"MIXED-MODE")=="YES")
     tCSSEER=T.Value("Not Space Cool SEER",2,P.Vint(Id,"NOT-CASE"));
// mixed-mode SEER
   else // i.e., not mixed-mode
     tCSSEER=T.Value("Not Space Cool SEER",1,P.Vint(Id,"NOT-CASE"));
// regular SEER
 } // end if

 else // i.e., not 2010 notional
   //cooling seasonal efficiency
   tCSSEER=P.Vnum(pId,"COOL-SSEER");



// cout << endl << "Not type: " << P.Vcod(gId,"NOT-TYPE") << "
Cooling Eff: " << tCSSEER << endl;

//        if    ((T.Value("tSYS_COOL",1,P.Vcodid(pId,"TYPE"))==1    ||
P.Vcodid(gId,"NOT-TYPE")>1) && P.Vint(Id,"SERVICE-STRATEGY")!=4)  {
  if        ((T.Value("tSYS_COOL",1,P.Vcodid(pId,"TYPE"))==1        ||
P.Vcodid(gId,"NOT-TYPE")==2) && P.Vint(Id,"SERVICE-STRATEGY")!=4)  {
   //calculate primary energy for cooling by applying efficiency to
demand
   if(tCSSEER!=0)
     for(int m=0;m<12;m++) {
       mQ_PRIM_COOL_M[m]=mQ_DEM_COOL_M[m]/tCSSEER;
       mQ_PRIM_COOL+=mQ_PRIM_COOL_M[m];
     }
   else {
      {ostringstream os; os << "ERROR: tCSSEER=0.0" << endl;
       SendLog(os.str()); WriteLog();}
     assert(1==0);
   }
  }
```

```
// Applying the correction for electric power factor is now done on
ZONE calculated energy
// rather than CO2 so would now be included in mQ PRIM COOL M and
mQ_PRIM_COOL - AN 18/01/08
 //First we need to determine the fuel type for cooling
 int mFUEL CODID;

// In the next line, the codid of the cooling fuel type is matched to
the codid of the full list of fuel types
 // (as in heating)
 mFUEL CODID=(int)T.Value("tFUEL TYPE COOL",1,P.Vcodid(pId,"FUEL-
TYPE-COOL"));

 if ((mFUEL CODID==10)) { // If the fuel for cooling is grid-
electricity and this is the actual building
   for(int m=0;m<12;m++)
     mQ PRIM COOL M[m]*=(1-
T.Value("tPOWER FACTOR",1,P.Vcodid(gId,"ELEC-POWER-FACTOR")));
   mQ_PRIM_COOL*=(1-T.Value("tPOWER_FACTOR",1,P.Vcodid(gId,"ELEC-
POWER-FACTOR")));
 }


 //store primary energy for cooling for ZONE, monthly and annual
values in kWh and MJ
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ PRIM COOL M[i] << ", ";
  res << mQ PRIM COOL M[11] << " }";
  P.ModKey(Id,"Q-PRIM-COOL-M",res.str());
 }
 {
  ostringstream res; res << mQ PRIM COOL;
  P.ModKey(Id,"Q-PRIM-COOL",res.str());
 }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ PRIM COOL M[i]/3.6 << ", "; //3.6 used to convert from
MJ to kWh
  res << mQ PRIM COOL M[11]/3.6 << " }";
  P.ModKey(Id,"KWH-COOL-M",res.str());
 }
 {
  ostringstream res; res << mQ PRIM COOL/3.6; //3.6 used to convert
from MJ to kWh
  P.ModKey(Id,"KWH-COOL",res.str());
 }

}

//v2.0.a function moved from HVAC to zone level:
void ZONE::fQ PRIM EQUP(PROJECTDefaults& D, PROJECTTables& T, BOC& P)
{
  if(func report) cout << endl << "  " << "ZONE::fQ PRIM EQUP";
//calculates primary energy needed for equipment in this zone

  double mAREA ZONE=0;
  double aux=0;
  double mQ_PRIM_EQUP=0;
  double mQ PRIM EQUP M[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  deque<double> mQ PRIM EQUP Z SEN;
  deque<double> mQ_PRIM_EQUP_Z_LAT;

  //primary energy required for equipment, sensible and latent
  mQ_PRIM_EQUP_Z_SEN=P.Vlnum(Id,"EQUP-HG-SEN");
```

```
  mQ_PRIM_EQUP_Z_LAT=P.Vlnum(Id,"EQUP-HG-LAT");
  //area of this zone * multiplier
//      mAREA_ZONE=P.Vnum(Id,"AREA")*P.Vnum(Id,"MULTIPLIER");   //ZONE
MULTIPLIER APPLIED HERE
  mAREA_ZONE=P.Vnum(Id,"AREA");

    //for each month:
    for (int m=0;m<12;m++) {
      //check if you are using accumulated activity database,
      if(P.Vcod(Id,"ACT-DBS-ACU")=="YES")

aux=(mQ_PRIM_EQUP_Z_SEN[m]+mQ_PRIM_EQUP_Z_LAT[m])*mAREA_ZONE*D.Value(
"D008ACU");
      else

aux=(mQ_PRIM_EQUP_Z_SEN[m]+mQ_PRIM_EQUP_Z_LAT[m])*D.Value("D008")*T.V
alue("NJ",1,m+1)*mAREA_ZONE;
      mQ_PRIM_EQUP_M[m]+=aux;
      mQ_PRIM_EQUP+=aux;
    }

// Applying the correction for electric power factor is now done on
ZONE calculated energy
// rather than CO2 so would now be included in mQ_PRIM_EQUP_M and
mQ_PRIM_EQUP - AN 18/01/08
// If this is the actual building (fuel for equipment is always grid-
electricity)
   for(int m=0;m<12;m++)
     mQ_PRIM_EQUP_M[m]*=(1-
T.Value("tPOWER_FACTOR",1,P.Vcodid(gId,"ELEC-POWER-FACTOR")));
   mQ_PRIM_EQUP*=(1-T.Value("tPOWER_FACTOR",1,P.Vcodid(gId,"ELEC-
POWER-FACTOR")));


 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ_PRIM_EQUP_M[i] << ", ";
  res << mQ_PRIM_EQUP_M[11] << " }";
  P.ModKey(Id,"Q-PRIM-EQUP-M",res.str());
 }
 {
  ostringstream res; res << mQ_PRIM_EQUP;
  P.ModKey(Id,"Q-PRIM-EQUP",res.str());
 }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ_PRIM_EQUP_M[i]/3.6 << ", ";
  res << mQ_PRIM_EQUP_M[11]/3.6 << " }";
  P.ModKey(Id,"KWH-EQUP-M",res.str());
 }
 {
  ostringstream res; res << mQ_PRIM_EQUP/3.6;
  P.ModKey(Id,"KWH-EQUP",res.str());
 }

}

//v2.0.a function moved from HVAC to zone level:
void ZONE::fQ_PRIM_LI(PROJECTDefaults& D, PROJECTTables& T, BOC& P) {
  if(func_report) cout << endl << "  " << "ZONE::fQ_PRIM_LI";
//calculates primary energy needed for lighting in zones served by
HVAC system

  double mQ_PRIM_LI=0;
  double mQ_PRIM_LI_M[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
  deque<double> mQ_PRIM_LI_Z;
```

```
  //primary lighting energy for zone
  mQ_PRIM_LI_Z=P.Vlnum(Id,"LIGHT-ENERGY");
  //internal gain from lighting will only be equal to LIGHT-ENERGY if
LIGHT_GAIN_F==1
  //for each month of year:
    for (int m=0;m<12;m++) {
      mQ_PRIM_LI_M[m]=mQ_PRIM_LI_Z[m]*3.6*P.Vnum(Id,"AREA");
//convert to MJ, JO 17Jan08 I have taken the + out to avoid confusion
in the future in case code changes.
      mQ_PRIM_LI+=mQ_PRIM_LI_M[m];

    }

// Applying the correction for electric power factor is now done on
ZONE calculated energy
//  rather  than  CO2  so  would  now  be  included  in  mQ PRIM LI M  and
mQ PRIM LI - AN 18/01/08
// If this is the actual building (fuel for lighting is always grid-
electricity)
   for(int m=0;m<12;m++)
     mQ PRIM LI M[m]*=(1-
T.Value("tPOWER FACTOR",1,P.Vcodid(gId,"ELEC-POWER-FACTOR")));
   mQ PRIM LI*=(1-T.Value("tPOWER FACTOR",1,P.Vcodid(gId,"ELEC-POWER-
FACTOR")));

/*
cout << endl <<
        T.Value("tPOWER FACTOR",1,P.Vcodid(gId,"ELEC-POWER-
FACTOR"))<< endl <<
        mQ PRIM LI Z[0] << endl <<
        mQ PRIM LI << endl <<
        "" << endl; system("PAUSE");
*/

 //store lighting energy, monthly and annual values in kWh and MJ
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ PRIM LI M[i] << ", ";
  res << mQ PRIM LI M[11] << " }";
  P.ModKey(Id,"Q-PRIM-LI-M",res.str());
 }

 {
  ostringstream res; res << mQ PRIM LI;
  P.ModKey(Id,"Q-PRIM-LI",res.str());
 }

 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ PRIM LI M[i]/3.6 << ", ";
  res << mQ_PRIM_LI_M[11]/3.6 << " }";
  P.ModKey(Id,"KWH-LI-M",res.str());
 }
 {
  ostringstream res; res << mQ PRIM LI/3.6;
  P.ModKey(Id,"KWH-LI",res.str());
 }

}

//v2.0.a function moved from HVAC to zone level:
void ZONE::fFUEL TYPE CO2(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func report) cout << endl << "  " << "ZONE::fFUEL TYPE CO2";
//calculates primary energy needed for each fuel type for zone
//and CO2 output for zone
```

```
 double mKWH NATGAS M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH NATGAS=0.0;
 double mKWH LPG M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH_LPG=0.0;
 double mKWH_BIOGAS M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH BIOGAS=0.0;
 double mKWH_OIL_M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH OIL=0.0;
 double mKWH COAL M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH_COAL=0.0;
 double mKWH ANTHRACITE M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH ANTHRACITE=0.0;
 double mKWH_SMOKELESS_M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH SMOKELESS=0.0;
 double mKWH DUELFUEL M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH_DUELFUEL=0.0;
 double mKWH BIOMASS M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH BIOMASS=0.0;
 double mKWH_SUPELEC_M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH SUPELEC=0.0;
 double mKWH WASTEHEAT M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH WASTEHEAT=0.0;
 double mKWH DISTRICT HEATING M[12]={0,0,0,0,0,0,0,0,0,0,0,0};
 double mKWH DISTRICT HEATING=0.0;

// New parameters to save the annual heating fuel ONLY (space heating
and DHW only).
 double mKWH NATGAS heat=0.0;
 double mKWH LPG heat=0.0;
 double mKWH BIOGAS heat=0.0;
 double mKWH OIL heat=0.0;
 double mKWH COAL heat=0.0;
 double mKWH ANTHRACITE heat=0.0;
 double mKWH SMOKELESS heat=0.0;
 double mKWH DUELFUEL heat=0.0;
 double mKWH BIOMASS heat=0.0;
 double mKWH SUPELEC heat=0.0;
 double mKWH_WASTEHEAT_heat=0.0;
 double mKWH DISTRICT HEATING heat=0.0;
 double  mKWH HEATPUMP heat=0.0;            // annual heating (space
heating and DHW only) attributed to heat pumps

 // 10 April 2010 JO: Shell and Core approach for Part L 2010 (Steve
Irving). System & Lighting add 0 if Shell and Core building at built
stage.
 if(!(P.Vcodid(cId,"SHELL-AND-CORE")==1     &&     P.Vcodid(cId,"EPC-
TYPE")==1   &&   P.Vcodid(cId,"BR-STAGE")==2   &&   P.Vcodid(Id,"SHELL-
ASSUMED")==1))
 {
     //add heating fuel use to correct fuel type
     deque<double>  mKWH HEAT FUEL=P.Vlnum(Id,"KWH-HEAT-M");   //NOTE:
ZONE  MULTIPLIER  IS  ALREADY  APPLIED  TO  THESE  FIGURES  -  !!!!  AN
(27/03/2008): No, it doesn't seem to be. The multiplier is applied
later when they are summed at HVAC level.
     for (int m=0;m<12;m++) {
       switch (P.Vcodid(pId,"FUEL-TYPE"))   {
         //nat gas
       case 1: {
         mKWH_NATGAS_M[m]+=mKWH_HEAT_FUEL[m];
         mKWH NATGAS+=mKWH HEAT FUEL[m];
         mKWH NATGAS heat+=mKWH HEAT FUEL[m];
         }
         break;
       //LPG
       case 2: {
         mKWH LPG M[m]+=mKWH HEAT FUEL[m];
         mKWH LPG+=mKWH HEAT FUEL[m];
         mKWH_LPG_heat+=mKWH_HEAT_FUEL[m];
```

```
        }
        break;
      //biogas
      case 3: {
        mKWH_BIOGAS_M[m]+=mKWH_HEAT_FUEL[m];
        mKWH BIOGAS+=mKWH HEAT FUEL[m];
        mKWH BIOGAS heat+=mKWH HEAT FUEL[m];
        }
        break;
      //oil
      case 4: {
        mKWH OIL M[m]+=mKWH HEAT FUEL[m];
        mKWH OIL+=mKWH HEAT FUEL[m];
        mKWH_OIL_heat+=mKWH_HEAT_FUEL[m];
        }
        break;
      //coal
      case 5: {
        mKWH COAL M[m]+=mKWH HEAT FUEL[m];
        mKWH_COAL+=mKWH_HEAT_FUEL[m];
        mKWH COAL heat+=mKWH HEAT FUEL[m];
        }
        break;
      //anthracite
      case 6: {
        mKWH ANTHRACITE M[m]+=mKWH HEAT FUEL[m];
        mKWH ANTHRACITE+=mKWH HEAT FUEL[m];
        mKWH ANTHRACITE heat+=mKWH HEAT FUEL[m];
        }
        break;
      //smokeless fuel
      case 7: {
        mKWH SMOKELESS M[m]+=mKWH HEAT FUEL[m];
        mKWH SMOKELESS+=mKWH HEAT FUEL[m];
        mKWH SMOKELESS heat+=mKWH HEAT FUEL[m];
        }
        break;
      //dual fuel
      case 8: {
        mKWH DUELFUEL M[m]+=mKWH HEAT FUEL[m];
        mKWH DUELFUEL+=mKWH HEAT FUEL[m];
        mKWH_DUELFUEL_heat+=mKWH_HEAT_FUEL[m];
        }
        break;
      //biomass
      case 9: {
        mKWH BIOMASS M[m]+=mKWH HEAT FUEL[m];
        mKWH_BIOMASS+=mKWH_HEAT_FUEL[m];
        mKWH BIOMASS heat+=mKWH HEAT FUEL[m];
        }
        break;
      //grid supplied electricity
      case 10: {
        mKWH_SUPELEC_M[m]+=mKWH_HEAT_FUEL[m];
        mKWH SUPELEC+=mKWH HEAT FUEL[m];
        mKWH SUPELEC heat+=mKWH HEAT FUEL[m];
        }
        break;
      //waste heat
      case 12: {
        mKWH WASTEHEAT M[m]+=mKWH HEAT FUEL[m];
        mKWH WASTEHEAT+=mKWH HEAT FUEL[m];
        mKWH_WASTEHEAT_heat+=mKWH_HEAT_FUEL[m];
        }
        break;
      //district heating
      case 13: {
        mKWH DISTRICT HEATING M[m]+=mKWH HEAT FUEL[m];
        mKWH_DISTRICT_HEATING+=mKWH_HEAT_FUEL[m];
```

```
          mKWH DISTRICT HEATING heat+=mKWH HEAT FUEL[m];
          }
          break;
        }
        if ((P.Vcodid(pId,"HEAT-SOURCE")==12) || (P.Vcodid(pId,"HEAT-
SOURCE")==13)       ||       (P.Vcodid(pId,"HEAT-SOURCE")==18)       ||
(P.Vcodid(pId,"HEAT-SOURCE")==19)) // if the heat source of the
parent HVAC is a heat pump
           mKWH HEATPUMP heat+=mKWH HEAT FUEL[m];
     }

    // JO (24-07-2007) Change to Fuel for cooling depending on Heat
Source.
    // If Heat Pump then Fuel Type for the Heat Pump defined in the
Heating Tab
    // is also used for the cooling system.
     deque<double> mCOOL=P.Vlnum(Id,"KWH-COOL-M");
    /*
     if    (P.Vcod(gId,"ACT-NOT")=="ACT"    &&    (P.Vcodid(pId,"HEAT-
SOURCE")==12 || P.Vcodid(pId,"HEAT-SOURCE")==13)) { // 05 Sep 08, JO:
It is OK leaving only 12, 13 (not including 18, 19) as the latter are
electric Heat Pumps, therefore covered by the else clause
        for (int m=0;m<12;m++) {
            switch (P.Vcodid(pId,"FUEL-TYPE"))   {
              //nat gas
            case 1: {
              mKWH NATGAS M[m]+=mCOOL[m];
              mKWH NATGAS+=mCOOL[m];
              }
              break;
            //LPG
            case 2: {
              mKWH LPG M[m]+=mCOOL[m];
              mKWH LPG+=mCOOL[m];
              }
              break;
            //biogas
            case 3: {
              mKWH_BIOGAS_M[m]+=mCOOL[m];
              mKWH BIOGAS+=mCOOL[m];
              }
              break;
            //oil
            case 4: {
              mKWH_OIL_M[m]+=mCOOL[m];
              mKWH OIL+=mCOOL[m];
              }
              break;
            //coal
            case 5: {
              mKWH_COAL_M[m]+=mCOOL[m];
              mKWH COAL+=mCOOL[m];
              }
              break;
            //anthracite
            case 6: {
              mKWH_ANTHRACITE_M[m]+=mCOOL[m];
              mKWH ANTHRACITE+=mCOOL[m];
              }
              break;
            //smokeless fuel
            case 7: {
              mKWH_SMOKELESS_M[m]+=mCOOL[m];
              mKWH SMOKELESS+=mCOOL[m];
              }
              break;
            //dual fuel
            case 8: {
              mKWH_DUELFUEL_M[m]+=mCOOL[m];
```

```
              mKWH DUELFUEL+=mCOOL[m];
              }
              break;
            //biomass
            case 9: {
              mKWH BIOMASS M[m]+=mCOOL[m];
              mKWH BIOMASS+=mCOOL[m];
              }
              break;
            //grid supplied electricity
            case 10: {
              mKWH SUPELEC M[m]+=mCOOL[m];
              mKWH SUPELEC+=mCOOL[m];
              }
              break;
            //waste heat
            case 12: {
              mKWH WASTEHEAT M[m]+=mCOOL[m];
              mKWH WASTEHEAT+=mCOOL[m];
              }
              break;
            //district heating
            case 13: {
              mKWH DISTRICT HEATING M[m]+=mCOOL[m];
              mKWH DISTRICT HEATING+=mCOOL[m];
              }
              break;
            }
          }
      }
      else {
        for (int m=0;m<12;m++) {
            mKWH SUPELEC M[m]+=mCOOL[m];
            mKWH SUPELEC+=mCOOL[m];
        }
      }
     */

      // !!! 15/12/2008: Cooling now has its own fuel type keyword
      // Below, the codid of the cooling fuel type is matched to the
codid of the full list of fuel types(as in heating)
      if (P.Vcod(gId,"ACT-NOT")=="ACT") {
        for (int m=0;m<12;m++) {
          switch  ((int)T.Value("tFUEL TYPE COOL",1,P.Vcodid(pId,"FUEL-
TYPE-COOL")))   {
              //nat gas
            case 1: {
              mKWH_NATGAS_M[m]+=mCOOL[m];
              mKWH NATGAS+=mCOOL[m];
              }
              break;
            //LPG
            case 2: {
              mKWH_LPG_M[m]+=mCOOL[m];
              mKWH LPG+=mCOOL[m];
              }
              break;
            //biogas
            case 3: {
              mKWH_BIOGAS_M[m]+=mCOOL[m];
              mKWH BIOGAS+=mCOOL[m];
              }
              break;
            //oil
            case 4: {
              mKWH_OIL_M[m]+=mCOOL[m];
              mKWH OIL+=mCOOL[m];
              }
              break;
```

```
            //grid supplied electricity
            case 10: {
              mKWH SUPELEC M[m]+=mCOOL[m];
              mKWH SUPELEC+=mCOOL[m];
              }
              break;
            }
          }
      }
      else { // for the notional building, the cooling is always
electric
        for (int m=0;m<12;m++) {
            mKWH SUPELEC M[m]+=mCOOL[m];
            mKWH_SUPELEC+=mCOOL[m];
        }
      }

      //add lighting, auxiliary load to grid supplied electricity:
      //equipment is not added here as it is not covered by building
Regs part L2
      deque<double> mLI=P.Vlnum(Id,"KWH-LI-M");
      deque<double> mAUX=P.Vlnum(Id,"KWH-AUX-M");

      for (int m=0;m<12;m++) {
        mKWH SUPELEC M[m]+=mLI[m]+mAUX[m]+0.0; // Taken EQUP out of
CO2 calculation
        mKWH SUPELEC+=mLI[m]+mAUX[m]+0.0; // Taken EQUP out of CO2
calculation
      }

      // 10 April 2010 Shell and core. To pro-rate the renewable
energy source we need to know actual energy for those systems/zones
not within the catch.
      double mKWH ACTUAL=0.0;
      for (int m=0;m<12;m++) {
        mKWH ACTUAL+=mKWH HEAT FUEL[m]+mCOOL[m]+mAUX[m]+mLI[m]+0.0; //
Taken EQUP out of CO2 calculation
      }
      {
       ostringstream res; res << mKWH ACTUAL;
       P.ModKey(Id,"KWH-ACTUAL",res.str());
      }
 }
 else { // 10 April 2010 JO: When Shell and Core applies
   // store assumed energy for later calcs regarding renewable
sources.
      deque<double> mHEAT=P.Vlnum(Id,"KWH-HEAT-M");
      deque<double> mCOOL=P.Vlnum(Id,"KWH-COOL-M");
      deque<double> mLI=P.Vlnum(Id,"KWH-LI-M");
      deque<double> mAUX=P.Vlnum(Id,"KWH-AUX-M");
      double mKWH_ASSUMED=0.0;
      for (int m=0;m<12;m++) {
        mKWH ASSUMED+=mHEAT[m]+mCOOL[m]+mAUX[m]+mLI[m]+0.0;  //  Taken
EQUP out of CO2 calculation
      }
      {
       ostringstream res; res << mKWH_ASSUMED;
       P.ModKey(Id,"KWH-ASSUMED",res.str());
      }
 }

 //store energy used by fuel type in both kWh and MJ, and both annual
and monthly:
 {
  ostringstream res; res << mKWH NATGAS heat;
  P.ModKey(Id,"KWH-NATGAS-HEAT",res.str());  // for heating (and DHW)
only
 }
 {
```

```
  ostringstream res; res << mKWH NATGAS;
  P.ModKey(Id,"KWH-NATGAS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH NATGAS M[i] << ", ";
  res << mKWH_NATGAS_M[11] << " }";
  P.ModKey(Id,"KWH-NATGAS-M",res.str());
 }
 {
  ostringstream res; res << mKWH NATGAS*3.6;
  P.ModKey(Id,"Q-NATGAS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH NATGAS M[i]*3.6 << ", ";
  res << mKWH NATGAS M[11]*3.6 << " }";
  P.ModKey(Id,"Q-NATGAS-M",res.str());
 }
 {
  ostringstream res; res << mKWH LPG heat;  // for heating (and DHW)
only
  P.ModKey(Id,"KWH-LPG-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH LPG;
  P.ModKey(Id,"KWH-LPG",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH LPG M[i] << ", ";
  res << mKWH LPG M[11] << " }";
  P.ModKey(Id,"KWH-LPG-M",res.str());
 }
 {
  ostringstream res; res << mKWH_LPG*3.6;
  P.ModKey(Id,"Q-LPG",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_LPG_M[i]*3.6 << ", ";
  res << mKWH LPG M[11]*3.6 << " }";
  P.ModKey(Id,"Q-LPG-M",res.str());
 }
 {
  ostringstream res; res << mKWH BIOGAS heat;  // for heating (and
DHW) only
  P.ModKey(Id,"KWH-BIOGAS-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH BIOGAS;
  P.ModKey(Id,"KWH-BIOGAS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH BIOGAS M[i] << ", ";
  res << mKWH BIOGAS M[11] << " }";
  P.ModKey(Id,"KWH-BIOGAS-M",res.str());
 }
 {
  ostringstream res; res << mKWH_BIOGAS*3.6;
  P.ModKey(Id,"Q-BIOGAS",res.str());
 }
 {
```

```
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH BIOGAS M[i]*3.6 << ", ";
  res << mKWH BIOGAS M[11]*3.6 << " }";
  P.ModKey(Id,"Q-BIOGAS-M",res.str());
 }
 {
  ostringstream res; res << mKWH OIL heat;  // for heating (and DHW)
only
  P.ModKey(Id,"KWH-OIL-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH OIL;
  P.ModKey(Id,"KWH-OIL",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH OIL M[i] << ", ";
  res << mKWH_OIL_M[11] << " }";
  P.ModKey(Id,"KWH-OIL-M",res.str());
 }
 {
  ostringstream res; res << mKWH OIL*3.6;
  P.ModKey(Id,"Q-OIL",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH OIL M[i]*3.6 << ", ";
  res << mKWH OIL M[11]*3.6 << " }";
  P.ModKey(Id,"Q-OIL-M",res.str());
 }
 {
  ostringstream res; res << mKWH COAL heat;     // for heating (and
DHW) only
  P.ModKey(Id,"KWH-COAL-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH COAL;
  P.ModKey(Id,"KWH-COAL",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH COAL M[i] << ", ";
  res << mKWH COAL M[11] << " }";
  P.ModKey(Id,"KWH-COAL-M",res.str());
 }
 {
  ostringstream res; res << mKWH_COAL*3.6;
  P.ModKey(Id,"Q-COAL",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_COAL_M[i]*3.6 << ", ";
  res << mKWH COAL M[11]*3.6 << " }";
  P.ModKey(Id,"Q-COAL-M",res.str());
 }
 {
  ostringstream res; res << mKWH ANTHRACITE heat;     // for heating
(and DHW) only
  P.ModKey(Id,"KWH-ANTHRACITE-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH ANTHRACITE;
  P.ModKey(Id,"KWH-ANTHRACITE",res.str());
 }
```

```
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH ANTHRACITE M[i] << ", ";
  res << mKWH_ANTHRACITE_M[11] << " }";
  P.ModKey(Id,"KWH-ANTHRACITE-M",res.str());
 }
 {
  ostringstream res; res << mKWH ANTHRACITE*3.6;
  P.ModKey(Id,"Q-ANTHRACITE",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH ANTHRACITE M[i]*3.6 << ", ";
  res << mKWH ANTHRACITE M[11]*3.6 << " }";
  P.ModKey(Id,"Q-ANTHRACITE-M",res.str());
 }
 {
  ostringstream res; res << mKWH SMOKELESS heat;       // for heating
(and DHW) only
  P.ModKey(Id,"KWH-SMOKELESS-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH SMOKELESS;
  P.ModKey(Id,"KWH-SMOKELESS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH SMOKELESS M[i] << ", ";
  res << mKWH SMOKELESS M[11] << " }";
  P.ModKey(Id,"KWH-SMOKELESS-M",res.str());
 }
 {
  ostringstream res; res << mKWH SMOKELESS*3.6;
  P.ModKey(Id,"Q-SMOKELESS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_SMOKELESS_M[i]*3.6 << ", ";
  res << mKWH SMOKELESS M[11]*3.6 << " }";
  P.ModKey(Id,"Q-SMOKELESS-M",res.str());
 }
 {
  ostringstream res; res << mKWH DUELFUEL heat;        // for heating
(and DHW) only
  P.ModKey(Id,"KWH-DUELFUEL-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH DUELFUEL;
  P.ModKey(Id,"KWH-DUELFUEL",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH DUELFUEL M[i] << ", ";
  res << mKWH DUELFUEL M[11] << " }";
  P.ModKey(Id,"KWH-DUELFUEL-M",res.str());
 }
 {
  ostringstream res; res << mKWH_DUELFUEL*3.6;
  P.ModKey(Id,"Q-DUELFUEL",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_DUELFUEL_M[i]*3.6 << ", ";
```

```
  res << mKWH DUELFUEL M[11]*3.6 << " }";
  P.ModKey(Id,"Q-DUELFUEL-M",res.str());
 }
 {
  ostringstream res; res << mKWH BIOMASS heat;         // for heating
(and DHW) only
  P.ModKey(Id,"KWH-BIOMASS-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH BIOMASS;
  P.ModKey(Id,"KWH-BIOMASS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH BIOMASS M[i] << ", ";
  res << mKWH_BIOMASS_M[11] << " }";
  P.ModKey(Id,"KWH-BIOMASS-M",res.str());
 }
 {
  ostringstream res; res << mKWH BIOMASS*3.6;
  P.ModKey(Id,"Q-BIOMASS",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH BIOMASS M[i]*3.6 << ", ";
  res << mKWH BIOMASS M[11]*3.6 << " }";
  P.ModKey(Id,"Q-BIOMASS-M",res.str());
 }
 {
  ostringstream res; res << mKWH SUPELEC heat;         // for heating
(and DHW) only
  P.ModKey(Id,"KWH-SUPELEC-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH SUPELEC;
  P.ModKey(Id,"KWH-SUPELEC",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH SUPELEC M[i] << ", ";
  res << mKWH SUPELEC M[11] << " }";
  P.ModKey(Id,"KWH-SUPELEC-M",res.str());
 }
 {
  ostringstream res; res << mKWH_SUPELEC*3.6;
  P.ModKey(Id,"Q-SUPELEC",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_SUPELEC_M[i]*3.6 << ", ";
  res << mKWH SUPELEC M[11]*3.6 << " }";
  P.ModKey(Id,"Q-SUPELEC-M",res.str());
 }
 {
  ostringstream res; res << mKWH WASTEHEAT heat;       // for heating
(and DHW) only
  P.ModKey(Id,"KWH-WASTEHEAT-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH WASTEHEAT;
  P.ModKey(Id,"KWH-WASTEHEAT",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
```

```
    res << mKWH WASTEHEAT M[i] << ", ";
  res << mKWH WASTEHEAT M[11] << " }";
  P.ModKey(Id,"KWH-WASTEHEAT-M",res.str());
 }
 {
  ostringstream res; res << mKWH WASTEHEAT*3.6;
  P.ModKey(Id,"Q-WASTEHEAT",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH WASTEHEAT M[i]*3.6 << ", ";
  res << mKWH WASTEHEAT M[11]*3.6 << " }";
  P.ModKey(Id,"Q-WASTEHEAT-M",res.str());
 }
 {
  ostringstream res; res << mKWH DISTRICT HEATING heat;       // for
heating (and DHW) only
  P.ModKey(Id,"KWH-DISTRICT-HEATING-HEAT",res.str());
 }
 {
  ostringstream res; res << mKWH DISTRICT HEATING;
  P.ModKey(Id,"KWH-DISTRICT-HEATING",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH DISTRICT HEATING M[i] << ", ";
  res << mKWH DISTRICT HEATING M[11] << " }";
  P.ModKey(Id,"KWH-DISTRICT-HEATING-M",res.str());
 }
 {
  ostringstream res; res << mKWH DISTRICT HEATING*3.6;
  P.ModKey(Id,"Q-DISTRICT-HEATING",res.str());
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH_DISTRICT_HEATING_M[i]*3.6 << ", ";
  res << mKWH DISTRICT HEATING M[11]*3.6 << " }";
  P.ModKey(Id,"Q-DISTRICT-HEATING-M",res.str());
 }

 {
  ostringstream res; res << mKWH HEATPUMP heat;       // for heating
(and DHW) only
  P.ModKey(Id,"KWH-HEATPUMP-HEAT",res.str());
 }


//calculate CO2 for zone:

//for each fuel type, annual values
 double mKG_CO2_NATGAS=0.0;
 double mKG CO2 LPG=0.0;
 double mKG CO2 BIOGAS=0.0;
 double mKG_CO2_OIL=0.0;
 double mKG CO2 COAL=0.0;
 double mKG CO2 ANTHRACITE=0.0;
 double mKG_CO2_SMOKELESS=0.0;
 double mKG CO2 DUELFUEL=0.0;
 double mKG CO2 BIOMASS=0.0;
 double mKG_CO2_SUPELEC=0.0;
 double mKG CO2 WASTEHEAT=0.0;
 double mKG CO2 DISTRICT HEATING=0.0;
 // Total
 double mKG CO2=0.0;

 if((P.Vcodid(cId,"BR-TYPE")==5)   ||   (P.Vcodid(cId,"EPC-TYPE")==6)){
```

```
//Republic of Ireland only
    //use conversion factor from table to calc kG CO2 for each fuel
type, USE PRIMARY ENERGY IN KWH FOR CALCULATION:
     mKG CO2 NATGAS=P.Vnum(Id,"KWH-NATGAS")*T.Value("tKGCO2 IR",1,1);
     mKG_CO2_LPG=P.Vnum(Id,"KWH-LPG")*T.Value("tKGCO2_IR",1,2);
     mKG CO2 BIOGAS=P.Vnum(Id,"KWH-BIOGAS")*T.Value("tKGCO2 IR",1,3);
     mKG CO2 OIL=P.Vnum(Id,"KWH-OIL")*T.Value("tKGCO2 IR",1,4);
     mKG_CO2_COAL=P.Vnum(Id,"KWH-COAL")*T.Value("tKGCO2_IR",1,5);
     mKG CO2 ANTHRACITE=P.Vnum(Id,"KWH-
ANTHRACITE")*T.Value("tKGCO2 IR",1,6);
     mKG_CO2_SMOKELESS=P.Vnum(Id,"KWH-
SMOKELESS")*T.Value("tKGCO2 IR",1,7);
     mKG CO2 DUELFUEL=P.Vnum(Id,"KWH-
DUELFUEL")*T.Value("tKGCO2_IR",1,8);
     mKG CO2 BIOMASS=P.Vnum(Id,"KWH-
BIOMASS")*T.Value("tKGCO2 IR",1,9);
    //             double             mKG_CO2_SUPELEC=P.Vnum(Id,"KWH-
SUPELEC")*T.Value("tKGCO2 IR,10)*(1-
T.Value("tPOWER FACTOR",1,P.Vcodid(gId,"ELEC-POWER-FACTOR")));      //
v1.0.b Addition of power factor correction
    // Applying the correction to electric power factor is now done
on ZONE calculated energy
    // rather than CO2 so would now be included in KWH-SUPELEC - AN
18/01/08
     mKG CO2 SUPELEC=P.Vnum(Id,"KWH-
SUPELEC")*T.Value("tKGCO2 IR",1,10);
     mKG CO2 WASTEHEAT=P.Vnum(Id,"KWH-
WASTEHEAT")*T.Value("tKGCO2 IR",1,12);
     mKG CO2 DISTRICT HEATING=P.Vnum(Id,"KWH-DISTRICT-
HEATING")*T.Value("tKGCO2 IR",1,13);
 }
 else if((P.Vcodid(cId,"BR-TYPE")==6)){  //Jersey
    //use conversion factor from table to calc kG CO2 for each fuel
type, USE PRIMARY ENERGY IN KWH FOR CALCULATION:
     mKG CO2 NATGAS=P.Vnum(Id,"KWH-
NATGAS")*T.Value("tKGCO2 JER",1,1);
     mKG CO2 LPG=P.Vnum(Id,"KWH-LPG")*T.Value("tKGCO2 JER",1,2);
     mKG CO2 BIOGAS=P.Vnum(Id,"KWH-
BIOGAS")*T.Value("tKGCO2_JER",1,3);
     mKG CO2 OIL=P.Vnum(Id,"KWH-OIL")*T.Value("tKGCO2 JER",1,4);
     mKG CO2 COAL=P.Vnum(Id,"KWH-COAL")*T.Value("tKGCO2 JER",1,5);
     mKG_CO2_ANTHRACITE=P.Vnum(Id,"KWH-
ANTHRACITE")*T.Value("tKGCO2 JER",1,6);
     mKG CO2 SMOKELESS=P.Vnum(Id,"KWH-
SMOKELESS")*T.Value("tKGCO2_JER",1,7);
     mKG CO2 DUELFUEL=P.Vnum(Id,"KWH-
DUELFUEL")*T.Value("tKGCO2 JER",1,8);
     mKG_CO2_BIOMASS=P.Vnum(Id,"KWH-
BIOMASS")*T.Value("tKGCO2 JER",1,9);
    //             double             mKG CO2 SUPELEC=P.Vnum(Id,"KWH-
SUPELEC")*T.Value("tKGCO2_JER,10)*(1-
T.Value("tPOWER FACTOR",1,P.Vcodid(gId,"ELEC-POWER-FACTOR")));      //
v1.0.b Addition of power factor correction
    // Applying the correction to electric power factor is now done
on ZONE calculated energy
    // rather than CO2 so would now be included in KWH-SUPELEC - AN
18/01/08
     mKG CO2 SUPELEC=P.Vnum(Id,"KWH-
SUPELEC")*T.Value("tKGCO2 JER",1,10);
     mKG_CO2_WASTEHEAT=P.Vnum(Id,"KWH-
WASTEHEAT")*T.Value("tKGCO2 JER",1,12);
     mKG CO2 DISTRICT HEATING=P.Vnum(Id,"KWH-DISTRICT-
HEATING")*T.Value("tKGCO2_JER",1,13);
 }
 else{ // all UK
    //use conversion factor from table to calc kG CO2 for each fuel
type, USE PRIMARY ENERGY IN KWH FOR CALCULATION:
     mKG CO2 NATGAS=P.Vnum(Id,"KWH-NATGAS")*T.Value("tKGCO2",1,1);
     mKG_CO2_LPG=P.Vnum(Id,"KWH-LPG")*T.Value("tKGCO2",1,2);
```

```
     mKG CO2 BIOGAS=P.Vnum(Id,"KWH-BIOGAS")*T.Value("tKGCO2",1,3);
     mKG CO2 OIL=P.Vnum(Id,"KWH-OIL")*T.Value("tKGCO2",1,4);
     mKG CO2 COAL=P.Vnum(Id,"KWH-COAL")*T.Value("tKGCO2",1,5);
     mKG CO2 ANTHRACITE=P.Vnum(Id,"KWH-
ANTHRACITE")*T.Value("tKGCO2",1,6);
     mKG CO2 SMOKELESS=P.Vnum(Id,"KWH-
SMOKELESS")*T.Value("tKGCO2",1,7);
     mKG_CO2_DUELFUEL=P.Vnum(Id,"KWH-
DUELFUEL")*T.Value("tKGCO2",1,8);
     mKG CO2 BIOMASS=P.Vnum(Id,"KWH-BIOMASS")*T.Value("tKGCO2",1,9);
    //             double             mKG_CO2_SUPELEC=P.Vnum(Id,"KWH-
SUPELEC")*T.Value("tKGCO2",1,10)*(1-
T.Value("tPOWER FACTOR",1,P.Vcodid(gId,"ELEC-POWER-FACTOR")));     //
v1.0.b Addition of power factor correction
    // Applying the correction to electric power factor is now done
on ZONE calculated energy
    // rather than CO2 so would now be included in KWH-SUPELEC - AN
18/01/08
     mKG CO2 SUPELEC=P.Vnum(Id,"KWH-SUPELEC")*T.Value("tKGCO2",1,10);
     mKG_CO2_WASTEHEAT=P.Vnum(Id,"KWH-
WASTEHEAT")*T.Value("tKGCO2",1,12);
     mKG CO2 DISTRICT HEATING=P.Vnum(Id,"KWH-DISTRICT-
HEATING")*T.Value("tKGCO2",1,13);
 }


mKG CO2=mKG CO2 NATGAS+mKG CO2 LPG+mKG CO2 BIOGAS+mKG CO2 OIL+mKG CO2
 COAL+mKG CO2 ANTHRACITE+mKG CO2 SMOKELESS+mKG CO2 DUELFUEL+mKG CO2 B
IOMASS+mKG CO2 SUPELEC+mKG CO2 WASTEHEAT+mKG CO2 DISTRICT HEATING;

 {
  ostringstream res; res << mKG CO2 NATGAS;
  P.ModKey(Id,"KG-CO2-NATGAS",res.str());
 }
 {
  ostringstream res; res << mKG CO2 LPG;
  P.ModKey(Id,"KG-CO2-LPG",res.str());
 }
 {
  ostringstream res; res << mKG CO2 BIOGAS;
  P.ModKey(Id,"KG-CO2-BIOGAS",res.str());
 }
 {
  ostringstream res; res << mKG CO2 OIL;
  P.ModKey(Id,"KG-CO2-OIL",res.str());
 }
 {
  ostringstream res; res << mKG_CO2_COAL;
  P.ModKey(Id,"KG-CO2-COAL",res.str());
 }
 {
  ostringstream res; res << mKG CO2 ANTHRACITE;
  P.ModKey(Id,"KG-CO2-ANTHRACITE",res.str());
 }
 {
  ostringstream res; res << mKG CO2 SMOKELESS;
  P.ModKey(Id,"KG-CO2-SMOKELESS",res.str());
 }
 {
  ostringstream res; res << mKG_CO2_DUELFUEL;
  P.ModKey(Id,"KG-CO2-DUELFUEL",res.str());
 }
 {
  ostringstream res; res << mKG CO2 BIOMASS;
  P.ModKey(Id,"KG-CO2-BIOMASS",res.str());
 }
 {
  ostringstream res; res << mKG CO2 SUPELEC;
  P.ModKey(Id,"KG-CO2-SUPELEC",res.str());
```

```
  }
  {
   ostringstream res; res << mKG CO2 WASTEHEAT;
   P.ModKey(Id,"KG-CO2-WASTEHEAT",res.str());
  }
  {
   ostringstream res; res << mKG CO2 DISTRICT HEATING;
   P.ModKey(Id,"KG-CO2-DISTRICT-HEATING",res.str());
  }
  {
   ostringstream res; res << mKG_CO2;
   P.ModKey(Id,"KG-CO2",res.str());
  }

}

void ZONE::Recommendation(PROJECTDefaults& D, PROJECTTables& T, BOC&
P) {
  if(func report) cout << endl << "  " << "ZONE::Recommendation";
  // First Child Recommendation?
  for (int i=0;i<vW.size();i++) {
    vW[i].Recommendation(D,T,P);
  }

  // Then Zone Recommendation?


}

void ZONE::fAUX ENERGY(PROJECTDefaults& D, PROJECTTables& T, BOC& P)
{
  if(func report) cout << endl << "  " << "ZONE::fAUX ENERGY";

  double mAREA=P.Vnum(Id,"AREA");
  // AE-PSC
  double   mPSC=P.Vnum(Id,"AE-PIG")+P.Vnum(Id,"AE-PLG")+P.Vnum(Id,"AE-
PSG");
 {
  ostringstream res; res << mPSC;
  P.ModKey(Id,"AE-PSC",res.str());
 }

  double  mSys Corr=P.Vnum(Id,"SYSTEM-CORRECTION");    //  correction
factor due to leakage and controls

  // AE-PSH
    double mHEAT SET POINT=P.V1lnum(Id,"HEAT-SET-POINT",1);
    // AE-PFL
    double mPFL=P.Vnum(Id,"H-TR")*(mHEAT SET POINT-0.0)/mAREA;
    // AE-PVL
    double mH_VENT_HEAT=P.V1lnum(Id,"H-VENT-HEAT",1);
    double mPVL=mH VENT HEAT*(mHEAT SET POINT-0.0)/mAREA;

    double mPSH=mPFL+mPVL;

 {
  ostringstream res; res << mPFL;
  P.ModKey(Id,"AE-PFL",res.str());
 }

 {
  ostringstream res; res << mPVL;
  P.ModKey(Id,"AE-PVL",res.str());
 }
 {
  ostringstream res; res << mPSH;
  P.ModKey(Id,"AE-PSH",res.str());
 }
```

```
  // SCR
  double mSCR=mPSH;
  if(mPSC>mPSH   &&   T.Value("tSYS COOL",1,P.Vcodid(pId,"TYPE"))==1)
mSCR=mPSC;
  // 1.2 kg/m3 , 1.018 kJ/kgK , 8 K
  mSCR=mSCR/(1.2*1.018*8);

 {
  ostringstream res; res << mSCR;
  P.ModKey(Id,"AE-SCR",res.str());
 }

 // FAR
 double mFAR;
 // Use the design air flow rate if the air flow regulation type of
the demand-controlled ventilation is "Damper control"
 // and use the air flow rate as modified by the demand-controlled
ventilation if the regulation type is "Speed control".
 // It is because it is assumed that, in damper control, although the
dampers will be adjusted, the fan will be using
 // the same energy as for the design flow rate; hence the auxiliary
energy will not be reduced when driven by ventilation
 // needs. Note that the new methodology allows for heating/cooling
loads to drive the auxiliary energy consumption.
 if   ((P.Vcodid(Id,"F-CTRL-VENT-DEM"))!=1)   {   //   1=   No   demand-
controlled ventilation
   if (P.Vcodid(Id,"FLOW-REGU-TYPE")==1) // 1=Damper control
     mFAR=P.V1lnum(Id,"OA-FLOW-PERS-DES",1)/P.V1lnum(Id,"OCC-T-
HOURS",1);
   else // i.e., Speed control
     mFAR=P.V1lnum(Id,"OA-FLOW-PERS",1)/P.V1lnum(Id,"OCC-T-HOURS",1);
 }
 else // i.e., there is no demand controlled ventilation
   mFAR=P.V1lnum(Id,"OA-FLOW-PERS",1)/P.V1lnum(Id,"OCC-T-HOURS",1);

 {
  ostringstream res; res << mFAR;
  P.ModKey(Id,"AE-FAR",res.str());
 }

 //FPS1

 double
mFPS1=(mFAR*P.Vnum(pId,"SFP")*mSys Corr)+(mSCR*P.Vnum(Id,"SFP-TU"));
 {
  ostringstream res; res << mFPS1;
  P.ModKey(Id,"AE-FPS1",res.str());
 }
 //FPS2
 double mFPS2=mFAR;
 if(mSCR>mFAR) mFPS2=mSCR;
 mFPS2*=P.Vnum(pId,"SFP")*mSys Corr;
 {
  ostringstream res; res << mFPS2;
  P.ModKey(Id,"AE-FPS2",res.str());
 }
 //FPS3
// double mFPS3=mPSH/(1.22*8)*P.Vnum(Id,"SFP-TU");
 double mFPS3=mFAR;
 if((mSCR/5)>mFAR) mFPS3=mSCR/5;
 mFPS3*=P.Vnum(pId,"SFP")*mSys Corr;
 {
  ostringstream res; res << mFPS3;
  P.ModKey(Id,"AE-FPS3",res.str());
 }
 //FPS4
// double mFPS4=mPSH/(1.22*8)*P.Vnum(pId,"SFP")*mSys Corr;
 double mFPS4=mFAR*P.Vnum(pId,"SFP")*mSys Corr;
 {
```

```
  ostringstream res; res << mFPS4;
  P.ModKey(Id,"AE-FPS4",res.str());
 }

// FPS
 double mFPS;
 int mFPStype=T.Vint("tFPStype",1,P.Vcodid(pId,"TYPE"));
 if(P.Vcod(gId,"ACT-NOT")=="ACT")
   switch(mFPStype) {
     case 0: mFPS=0.0; break;
     case 1: mFPS=mFPS1; break;
     case 2: mFPS=mFPS2; break;
     case 3: mFPS=mFPS3; break;
     case 4: mFPS=mFPS4; break;
   }
 else {
   switch(mFPStype) {
     case 0: mFPS=0.0; break;
     case 1:
     case 2:
     case 3:
     case 4: {
             mFPS=mFPS1;
             if(mFPS2<mFPS) mFPS=mFPS2;
           }
          break;
   }
   //For E&W notional 2010, there is no fan power if there is mixed
mode operation according to the 2010 NCM Modelling Guide (para. 49)
   if(P.Vint(Id,"NOT-CASE")>=9 && P.Vcod(pId,"MIXED-MODE")=="YES")
     mFPS=0.0;
  }

 // Zonal mech vent (relevant only if the HVAC system does not
provide ventilation, i.e., it is 0 in tFPStype)
 if(P.Vcod(Id,"Q-V-TYPE")=="Mechanical")
   switch(mFPStype) {
//     case 0:
//     case 3:  mFPS+=mFAR*P.Vnum(Id,"VENT-SFP"); break;
     case 0:  mFPS+=mFAR*P.Vnum(Id,"VENT-SFP"); break;
   }

 //FPS
 {
  ostringstream res; res << mFPS;
  P.ModKey(Id,"AE-FPS",res.str());
 }

 //AUX-ENERGY
 double mAUX ENERGY[12]={ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
 deque<double> mSYS_T_HOURS=P.Vlnum(Id,"SYS-T-HOURS");

 // Do not apply the new fan energy calculation except to the actual
and the aggregate notional building for E, W, & NI.
 if(     (P.Vcod(gId,"ACT-NOT")=="ACT")     ||     ((P.Vcod(gId,"ACT-
NOT")=="NOT") && (P.Vint(Id,"NOT-CASE")>=9)) )  // 9=EW NOT 2010
   for(int i=0;i<12;i++)
     mAUX ENERGY[i]=mFPS*mSYS T HOURS[i]/1000; // kWh/m2 Aux Energy
(Fan only)
 else { // check if it's the Scottish notional building or the RoI
reference building and add fan energy if zone has mechanical
ventilation
   // 4=S6_NOT_2010 , 5=RoI_REF_2002 , 2=Mechanical
   if( ((P.Vint(Id,"NOT-CASE")==4) || (P.Vint(Id,"NOT-CASE")==5)) &&
(P.Vcodid(Id,"Q-V-TYPE")==2) )
     for(int i=0;i<12;i++)
        mAUX ENERGY[i]=P.V1lnum(Id,"OA-FLOW-PERS",i)*P.Vnum(Id,"VENT-
SFP")/1000;  // kWh/m2 Aux Energy (Fan only)
 } // end else
```

```
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mAUX_ENERGY[i] << ", ";
  res << mAUX ENERGY[11] << " }";
  P.ModKey(Id,"AE-FAN",res.str());
 }

  // Auxiliary energy due to pumps and exhaust fans only
  deque<double> mAUX_M=P.Vlnum(Id,"KWH-PUMP/EXH-AUX-M");   //KWH/m2

  //Add auxiliary energy due to night cooling if applicable in the
actual buildling
  double mMAX HOURS NC=P.Vnum(Id,"MAX-HOURS-NC");
  double mSFP NC=P.Vnum(Id,"SFP-NC");       // SFP for night cooling,
in W/(l/s);
  deque<double>  mOPT FLOW NC=P.Vlnum(Id,"OPT-FLOW-NC");      //  best
flow rate for night cooling, in l/s.m2
  deque<double>  mOPT_TO_MAX_NC=P.Vlnum(Id,"RATIO-OPT-MAX-NC");    //
ratio of best to max flow
  if(P.Vcod(gId,"ACT-NOT")=="ACT") {
    // Check if there is night cooling
    if(P.Vcod(Id,"NIGHT-COOLING")=="YES")
      for(int i=0;i<12;i++)
        mAUX M[i]+=mOPT FLOW NC[i]*mSFP NC*mMAX HOURS NC/1000;
//kWh/m2
  }

  // Add auxiliary energy due to mechanical ventilation fans
  for(int i=0;i<12;i++)
    mAUX M[i]+=mAUX ENERGY[i];  //KWH/m2

 //if there are destratification fans in zone - add the aux energy
for these - actual building only:
  double mDesAux;
  if(P.Vcod(gId,"ACT-NOT")=="ACT")
    if (P.Vcod(Id,"DESTRAT-FAN")=="YES") {
      deque<double> mTSC_KWH_AUX=P.Vlnum(Id,"TSC-KWH-AUX");
      for(int i=0;i<12;i++) {
        mDesAux=D.Value("dDESTRAT AUX")/3.6-mTSC KWH AUX[i];
        if(mDesAux<0.0) mDesAux=0.0;
        mAUX M[i]+=mDesAux;  //KWH/m2
      }
    }
  double mKWH AUX=0.0;
  double mQ AUX=0.0;

  for(int i=0;i<12;i++)
    mKWH AUX+=mAUX M[i];

  mQ AUX=mKWH AUX*3.6*mAREA;  //MJ
  mKWH AUX*=mAREA;    //KWH

  double mQ AUX M[12]= {0,0,0,0,0,0,0,0,0,0,0,0};
  double mKWH AUX M[12]= {0,0,0,0,0,0,0,0,0,0,0,0};
  for(int i=0;i<12;i++)  {
    mQ AUX M[i]=mAUX M[i]*3.6*mAREA; //MJ
    mKWH AUX M[i]=mAUX M[i]*mAREA;   //KWH
  }

// Applying the correction for electric power factor is now done on
ZONE calculated energy
// rather than CO2 so would now be included in mQ AUX M, mQ AUX,
mKWH AUX M, and mKWH AUX - AN 18/01/08
// If this is the actual building (fuel for auxiliary is always grid-
electricity)
   for(int m=0;m<12;m++) {
     mQ_AUX_M[m]*=(1-T.Value("tPOWER_FACTOR",1,P.Vcodid(gId,"ELEC-
```

```
POWER-FACTOR")));
     mKWH AUX M[m]*=(1-T.Value("tPOWER FACTOR",1,P.Vcodid(gId,"ELEC-
POWER-FACTOR")));
    }
   mQ_AUX*=(1-T.Value("tPOWER_FACTOR",1,P.Vcodid(gId,"ELEC-POWER-
FACTOR")));
   mKWH AUX*=(1-T.Value("tPOWER FACTOR",1,P.Vcodid(gId,"ELEC-POWER-
FACTOR")));


 {
  ostringstream res; res << mQ AUX;
  P.ModKey(Id,"Q-AUX",res.str());  //MJ
 }
 {
  ostringstream res; res << mKWH AUX;
  P.ModKey(Id,"KWH-AUX",res.str());   //KWh
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mKWH AUX M[i] << ", ";
  res << mKWH AUX M[11] << " }";
  P.ModKey(Id,"KWH-AUX-M",res.str());   //KWh
 }
 {
  ostringstream res; res << "{ ";
  for (int i=0;i<11;i++)
    res << mQ AUX M[i] << ", ";
  res << mQ AUX M[11] << " }";
  P.ModKey(Id,"Q-AUX-M",res.str());     //MJ
 }

}
```

Section 3

SBEM libraries: Matrix, BEEMLibDef, BEEMLibTab, ActlibraryTab

This section includes auxiliary library files with values used in the c++ code.

Matrix

Coded file

```
$,Config file for object and keyword in SBEM application, , , , , , , , , , , , , ,
$,File created on: , , ,15/11/2010 12:10:43, , , , , , , , , ,
$,Definitions:, , , , , , , , , , , , , ,
$,$,This symbol introduces comments in the file, , , , , , , , , , , ,
$,1,This symbol introduces a new object, , , , , , , , , , , , ,
$,2,This symbol introduces a new keyword for the object inmediately above, , , , , , , , , , , , ,
$,3,This symbol introduces a new code for the keyword inmediatyely above, , , , , , , , , , , , ,
$,ID,Identification number for Objects, keywords and codes. IT HAS TO BE IN INCREASING ORDER., , , , , , , , , , , , ,
$,Object/keyword Name: Group of characters (letters and numbers without spaces), , , , , , , , , , , , , ,
$,TYPE: Type of keywords. Types allowed are NUM (Numeric), STR (String), COD (Code), LIST (List), , , , , , , , , , , , , ,
$,Def Value: Default Value of this keyword, , , , , , , , , , , , , ,
$,(Error)Min Value: Inferior limit that produces an error if keyword value is smaller, , , , , , , , , , , , , ,
$,(Error)Max Value: Superior limit that produces an error if keyword value is bigger., , , , , , , , , , , , , ,
$,(Warning)Min Value: Inferior limit that produces a warning if keyword value is smaller, , , , , , , , , , , , , ,
$,(Warning)Max Value: Superior limit that produces a warning if keyword value is bigger., , , , , , , , , , , , , ,
$,(Certification check warning)Min Value: Inferior limit that produces a certification check warning if keyword value is smaller, , , , , , , , , , , , , ,
$,(Certification check warning)Max Value: Superior limit that produces a certification check warning if keyword value is bigger., , , , , , , , , , , , , ,
$,Unit: International System Unit., , , , , , , , , , , , , ,
$,2,3,4,5,6,7,8,9,10,11,12,13,14,15,
$,1,2,3,4,5,6,7,8,9,10,11,12,13,14,
$,FILTER:FILTER:FILTER:FILTER:FILTER:FILTER:FILTER:FILTER:FILTER:FILTER:FILTER:FILTER:FILTER:FILTER:FILTER:FILTER:FILTER:FILTER:FILTER:FILTER:FILTER:, , , , , , , , , , , , , ,
$,2, , , , ,Error, ,Warning, ,Certification check warning, , , , ,
$,2,ID,KEYWORD,TYPE,Def Value,Min Value,Max Value,Min Value,Max Value,Min Value,Max Value,Unit,$,Comments,
$,2, , ,(NUM),REQ (-8888) Required,X,X,X,X,X,X, ,$, ,
$,2, , ,(TEXT),EXP (-7777) Expression calculated in the programme, , , , , , , ,$, ,
$,2, , ,(REF),ODD (-5555), ,K in this column for a REF keyword means the type of object referred depends on the value of keyword named left to this cell, , , , , ,$, ,
$,2, , ,(COD),Numero,X,X, , , , , ,$, ,
```

```
$,2, , ,(LNUM),Id código,X,X,X,X,X,X, ,$, ,
$,2, , ,(LREF), ,NC (-6666) Don't check,NC (-6666) Don't check,NC (-6666) Don't check,NC (-6666) Don't check,NC (-6666) Don't check,NC (-6666) Don't check, ,$, ,
$,2, , , , , , , , , , , ,$, ,
$,1,ID and TYPE,OBJECT,MIN,MAX,PARENT OBJECT,Prop to be released, , , , , ,$, ,
$,1, , ,Numero,Numero,OBJECT, , , , , , ,$, ,
$,1, , , ,NC (-6666) Don't check, , , , , , , ,$, ,
$,1, , , , , , , , , , , , ,$, ,
$,3,ID,CODE, , , , , , , , , ,$, ,
$,3, , , , , , , , , , , , ,$, ,
 ,1,1,HVAC-SYSTEM,1,-6666, ,0, , , , , ,$, ,
 ,2,1,ISBEM-ID,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a Added _R21,
 ,2,2,TYPE,4,Single-duct VAV,1,30, , , , , ,$, ,
 ,3,1,No Heating or Cooling, , , , , , , , , ,$, ,
 ,3,2,Central heating using water: radiators, , , , , , , , , ,$, ,
 ,3,3,Central heating using water: convectors, , , , , , , , , ,$, ,
 ,3,4,Central heating using water: floor heating, , , , , , , , , ,$, ,
 ,3,5,Central heating using air distribution, , , , , , , , , ,$, ,
 ,3,6,Other local room heater - fanned, , , , , , , , , ,$,v2.1.a changed from 'Other local room heater',
 ,3,7,Unflued radiant heater, , , , , , , , , ,$, ,
 ,3,8,Flued radiant heater, , , , , , , , , ,$, ,
 ,3,9,Multiburner radiant heaters, , , , , , , , , ,$, ,
 ,3,10,Flued forced-convection air heaters, , , , , , , , , ,$, ,
 ,3,11,Unflued forced-convection air heaters, , , , , , , , , ,$, ,
 ,3,12,Single-duct VAV, , , , , , , , , ,$, ,
 ,3,13,Dual-duct VAV, , , , , , , , , ,$, ,
 ,3,14,Indoor packaged cabinet (VAV), , , , , , , , , ,$, ,
 ,3,15,Fan coil systems, , , , , , , , , ,$, ,
 ,3,16,Induction system, , , , , , , , , ,$, ,
 ,3,17,Constant volume system (fixed fresh air rate), , , , , , , , , ,$, ,
 ,3,18,Constant volume system (variable fresh air rate), , , , , , , , , ,$, ,
 ,3,19,Multizone (hot deck/cold deck), , , , , , , , , ,$, ,
 ,3,20,Terminal reheat (constant volume), , , , , , , , , ,$, ,
 ,3,21,Dual duct (constant volume), , , , , , , , , ,$, ,
 ,3,22,Chilled ceilings or passive chilled beams and displacement ventilation, , , , , , , , , ,$, ,
 ,3,23,Active chilled beams, , , , , , , , , ,$, ,
 ,3,24,Water loop heat pump, , , , , , , , , ,$, ,
 ,3,25,Variable refrigerant flow, , , , , , , , , ,$, ,
 ,3,26,Split or multi-split system, , , , , , , , , ,$, ,
 ,3,27,Split or multi-split system with ventilation, , , , , , , , , ,$, ,
 ,3,28,Single room cooling system, , , , , , , , , ,$, ,
 ,3,29,Other local room heater - unfanned, , , , , , , , , ,$,v2.1.a Added,
 ,2,3,HG-TYPE,1,-5555,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.3 Added,
 ,2,4,HEAT-GEN-SEFF,1,0.4,0,-6666,0.4,4.5,-6666,-6666,NC,$,v1.1.b changed from SYS-SEFF,
 ,2,5,RADIANT-EFF,1,0,0,1,0.1,1,0,1,NC,$,v1.1.b Added,
 ,2,6,RAD-TOT-EFF-RATIO,1,0.5,0,1,0,1,0,1,NC,$,v1.1.b Added,
 ,2,7,COOL-GEN-SEER,1,0.4,0,-6666,0.4,4.5,-6666,-6666,NC,$,v1.1.b Added,
 ,2,8,COOL-GEN-EER,1,0.4,0,-6666,0.4,6,-6666,-6666,NC,$,v1.1.b Added,
 ,2,9,SYS-SEFF-CRED,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.3 Added,
 ,2,10,SYS-CTRL-MIN-LIST,7,{ "information not provided by the user" },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.3 Added,
 ,2,11,SYS-CTRL-MIN,4,NO,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.3 Added,
 ,3,1,YES, , , , , , , , , ,$,v05.0.3 Added,
 ,3,2,NO, , , , , , , , , ,$,v05.0.3 Added,
 ,2,12,SYS-CTRL-ADD-LIST,7,{ "information not provided by the user" },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.3 Added,
 ,2,13,SYS-CTRL-ADD-CRED,5,{ 0.00 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.3 Added,
 ,2,14,AREA,1,EXP,0,1000000000,0,1000000000,0,1000000000,m2,$, ,
 ,2,15,Q-PERF-TOT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
 ,2,16,Q-PRIM-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
 ,2,17,Q-PRIM-HEAT-DHW,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22 Added,
```

,2,18,Q-PRIM-HEAT-CHP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.1.0 Added,
,2,19,KWH-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.3 Added,
,2,20,KG-CO2-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R01_CD Added ,
,2,21,KWH-HEAT-DHW,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22 Added,
,2,22,KWH-HEAT-CHP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.1.0 Added,
,2,23,Q-PRIM-COOL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,24,KWH-COOL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.3 Added,
,2,25,KG-CO2-COOL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R01_CD Added ,
,2,26,Q-PRIM-VENT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,7.2,
,2,27,Q-PRIM-EQUP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.3 Added,
,2,28,KWH-EQUP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.3 Added,
,2,29,Q-PRIM-LI,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,8,
,2,30,KWH-LI,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.3 Added,
,2,31,KG-CO2-LI,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R01_CD Added ,
,2,32,Q-PRIM-HUM,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,11,
,2,33,Q-PRIM-PUMP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,9,
,2,34,Q-PRIM-DHW,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,superseded v04.3.3,
,2,35,Q-AUX,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.8.4 Is the auxiliary energy consumption of heat generation appliances destined for heat supply to the heated zones in MJ,
,2,36,KWH-AUX,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,37,KG-CO2-AUX,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R01_CD Added ,
,2,38,HEAT-GEN-NUM,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.8.4 Is the number of heat generation appliances detined for heat supply to the heated zones, with a pilot flame,
,2,39,P-EFF-AGR,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,7.3.2.1 Is the agreed value for the effectiv epower of fans for the whole building, in kW,
,2,40,P-EFF-CONNECT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,41,Q-DEM-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,42,Q-DEM-COOL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,43,COOL-SSEER,1,-6666,0.1,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,44,HEAT-SSEFF,1,-6666,0.1,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,45,AUX-ENERGY,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,46,AUX-ENERGY-CORR,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,47,NOT-GROUP,4,Air conditioning,1,3, , , , ,$,v4.3.0 Added,
,3,1,Heating only, , , , , , , , ,$,v4.3.0 Added,
,3,2,Air conditioning, , , , , , , , ,$,v4.3.0 Added,
,3,3,Mechanical ventilation, , , , , , , , ,$,v4.3.0 Added,
,2,48,FUEL-TYPE,4,Oil,1,12, , , , ,$,v4.3.0 Added,
,3,1,Natural Gas, , , , , , , , ,$,v4.3.0 Added,
,3,2,LPG, , , , , , , , ,$,v4.3.0 Added,
,3,3,Biogas, , , , , , , , ,$,v4.3.0 Added,
,3,4,Oil, , , , , , , , ,$,v4.3.0 Added,
,3,5,Coal, , , , , , , , ,$,v4.3.0 Added,
,3,6,Anthracite, , , , , , , , ,$,v5.0.1 Added ID7,
,3,7,Smokeless Fuel (inc Coke), , , , , , , , ,$,v1.1 BUG from 1.0 corrected,
,3,8,Dual Fuel Appliances (Mineral + Wood), , , , , , , , ,$,v1.1 BUG from 1.0 corrected,
,3,9,Biomass, , , , , , , , ,$,v4.3.0 Added,
,3,10,Grid Supplied Electricity, , , , , , , , ,$,v4.3.0 Added,
,3,11,Grid Displaced Electricity, , , , , , , , ,$,v4.3.0 Added,
,3,12,Waste Heat, , , , , , , , ,$,v4.3.0 Added,
,3,13,District Heating, , , , , , , , ,$,v3.0.b_dev_R01 Added,
,2,49,FUEL-TYPE-COOL,4,Grid Supplied Electricity,1,12, , , , ,$,v3.3.b_dev_R04 Added,
,3,1,Natural Gas, , , , , , , , ,$,v3.3.b_dev_R04 Added,
,3,2,LPG, , , , , , , , ,$,v3.3.b_dev_R04 Added,
,3,3,Biogas, , , , , , , , ,$,v3.3.b_dev_R04 Added,
,3,4,Oil, , , , , , , , ,$,v3.3.b_dev_R04 Added,
,3,5,Grid Supplied Electricity, , , , , , , , ,$,v3.3.b_dev_R04 Added,
,2,50,HEAT-SOURCE,4,LTHW boiler,1,16, , , , ,$,v4.3.0 Added,
,3,1,LTHW boiler, , , , , , , , ,$,v4.3.0 Added,
,3,2,MTHW boiler, , , , , , , , ,$,v4.3.0 Added,
,3,3,HTHW boiler, , , , , , , , ,$,v4.3.0 Added,
,3,4,Steam boiler (hidden by now), , , , , , , , ,$,v4.3.0 Added,

,3,5,delete now, , , , , , , , ,$,v4.3.0 Added,
,3,6,Air heater, , , , , , , , ,$,v4.3.0 Added,
,3,7,Unflued gas warm air heater, , , , , , , , ,$,v4.3.0 Added,
,3,8,Unitary radiant heater, , , , , , , , ,$,v4.3.0 Added,
,3,9,Radiant heater, , , , , , , , ,$,v4.3.0 Added,
,3,10,Unflued radiant heater, , , , , , , , ,$,v4.3.0 Added,
,3,11,Direct or storage electric heater, , , , , , , , ,$,v4.3.0 Added,
,3,12,Heat pump (gas/oil): air source, , , , , , , , ,$,v3.2.c_R07 Mod,
,3,13,Heat pump (gas/oil): ground or water source, , , , , , , , ,$,v3.2.c_R07 Mod,
,3,14,Room heater, , , , , , , , ,$,v4.3.0 Added,
,3,15,District heating, , , , , , , , ,$,v4.3.0 Added,
,3,16,CHP, , , , , , , , ,$,v4.3.0 Added,
,3,17,Direct gas firing, , , , , , , , ,$,v4.3.0 Added,
,3,18,Heat pump (electric): air source, , , , , , , , ,$,v3.2.c_R07 Added,
,3,19,Heat pump (electric): ground or water source, , , , , , , , ,$,v3.2.c_R07 Added,
,2,51,Q-DEM-HEAT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,52,Q-DEM-COOL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,53,Q-PRIM-HEAT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,54,Q-PRIM-HEAT-CHP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.1.0 Added,
,2,55,Q-PRIM-COOL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,56,Q-AUX-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID8b,
,2,57,Q-PRIM-EQUP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,58,Q-PRIM-LI-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,59,KWH-HEAT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,60,KWH-HEAT-CHP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.1.0 Added,
,2,61,KWH-COOL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,62,KWH-AUX-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,63,KWH-EQUP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,64,KWH-LI-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,65,Q-NATGAS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,66,Q-NATGAS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,67,Q-LPG,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,68,Q-LPG-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,69,Q-BIOGAS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,70,Q-BIOGAS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,71,Q-OIL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,72,Q-OIL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,73,Q-COAL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,74,Q-COAL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,75,Q-ANTHRACITE,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,76,Q-ANTHRACITE-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,77,Q-SMOKELESS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,78,Q-SMOKELESS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,79,Q-DUELFUEL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,80,Q-DUELFUEL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,81,Q-BIOMASS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,82,Q-BIOMASS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,83,Q-SUPELEC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,84,Q-SUPELEC-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,85,Q-DISPELEC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,86,Q-DISPELEC-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,87,Q-WASTEHEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,88,Q-WASTEHEAT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,89,Q-DISTRICT-HEATING,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,90,Q-DISTRICT-HEATING-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,91,KWH-NATGAS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,92,KWH-NATGAS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,93,KWH-LPG,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,94,KWH-LPG-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,95,KWH-BIOGAS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,96,KWH-BIOGAS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,

```
,2,97,KWH-OIL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,98,KWH-OIL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,99,KWH-COAL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,100,KWH-COAL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,101,KWH-ANTHRACITE,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,102,KWH-ANTHRACITE-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,103,KWH-SMOKELESS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,104,KWH-SMOKELESS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,105,KWH-DUELFUEL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,106,KWH-DUELFUEL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,107,KWH-BIOMASS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,108,KWH-BIOMASS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,109,KWH-SUPELEC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,110,KWH-SUPELEC-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,111,KWH-DISPELEC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,112,KWH-DISPELEC-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,113,KWH-WASTEHEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,114,KWH-WASTEHEAT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,115,KWH-DISTRICT-HEATING,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,116,KWH-DISTRICT-HEATING-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,117,KWH-NATGAS-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,118,KWH-LPG-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,SBEM_v3.0.f_dev_R05_a01
Added,
,2,119,KWH-BIOGAS-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,120,KWH-OIL-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,SBEM_v3.0.f_dev_R05_a01
Added,
,2,121,KWH-COAL-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,122,KWH-ANTHRACITE-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,123,KWH-SMOKELESS-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,124,KWH-DUELFUEL-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,125,KWH-BIOMASS-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,126,KWH-SUPELEC-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,127,KWH-WASTEHEAT-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,128,KWH-DISTRICT-HEATING-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,129,KWH-HEATPUMP-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,130,HEAT-REC-SYSTEM,4,No heat recovery, , , , , , ,$,v04.3.4 Added ID1,
,3,1,No heat recovery, , , , , , , , ,$,v04.3.4 Added ID1,
,3,2,Plate heat exchanger (Recuperator), , , , , , , , ,$,v04.3.4 Added ID1,
,3,3,Two-elements-system, , , , , , , , ,$,v04.3.4 Added ID1,
,3,4,Loading cold with air-conditioning, , , , , , , , ,$,v04.3.4 Added ID1,
,3,5,Heat-pipes, , , , , , , , ,$,v04.3.4 Added ID1,
,3,6,Thermal wheel, , , , , , , , ,$,v04.3.4 Added ID1,
,3,7,Run around coil, , , , , , , , ,$,v04.3.4 Added ID1,
,2,131,HEAT-REC-SEFF,1,0,0,1,0,1,0,1,NC,$,v2.0.a Added,
,2,132,HEAT-REC-VAR-EFF,4,NO, , , , , , ,$,v4.0.b_dev_R66_a01 added,
,3,1,YES, , , , , , , , ,$,v4.0.b_dev_R66_a01 added,
,3,2,NO, , , , , , , , ,$,v4.0.b_dev_R66_a01 added,
,2,133,F-CTRL-VENT-TYPE,4,no recirculation, , , , , , ,$,v04.3.4 Added ID2,
,3,1,no recirculation, , , , , , , , ,$,v04.3.4 Added ID2,
,3,2,less than 10%, , , , , , , , ,$,v04.3.4 Added ID2,
```

```
,3,3,10% to 20%, , , , , , , , , ,$,v04.3.4 Added ID2,
,3,4,20% to 30%, , , , , , , , , ,$,v04.3.4 Added ID2,
,3,5,30% to 40%, , , , , , , , , ,$,v04.3.4 Added ID2,
,3,6,40% to 50%, , , , , , , , , ,$,v04.3.4 Added ID2,
,3,7,50% to 60%, , , , , , , , , ,$,v04.3.4 Added ID2,
,3,8,60% to 70%, , , , , , , , , ,$,v04.3.4 Added ID2,
,3,9,70% to 80%, , , , , , , , , ,$,v04.3.4 Added ID2,
,3,10,80% to 90%, , , , , , , , , ,$,v04.3.4 Added ID2,
,3,11,more than 90%, , , , , , , , , ,$,v04.3.4 Added ID2,
,2,134,F-CTRL-VENT-DEM,4,no demand controlled ventilation, , , , , , ,$,v1.2.a,
,3,1,no demand controlled ventilation, , , , , , , , , ,$,v1.2.a,
,2,135,Q-V-M-MAX,1,5,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.4 Added ID1,
,2,136,MULTIPLIER,1,1,1,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.4 Added,
,2,137,K,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.4 Added,
,2,138,GRAD,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.4 Added,
,2,139,AUX-FACTOR-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,140,AUX-FACTOR,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,141,CHP,4,NO, , , , , , , ,$,v05.0.8 Added,
,3,1,YES, , , , , , , , , ,$,v05.0.8 Added,
,3,2,NO, , , , , , , , , ,$,v05.0.8 Added,
,2,142,SFP,1,-6666,-6666,-6666,0.1,3.5,-6666,-6666,W/l.s,$,v05.1.9 Added,
,2,143,DUCT-LEAKAGE-PC,1,10,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R04_CD
Added,
,2,144,DUCT-LEAKAGE,4,Worse than A, , , , , , ,$,v05.1.9 Added,
,3,1,Class B, , , , , , , , , ,$,v05.1.9 Added,
,3,2,Class A, , , , , , , , , ,$,v05.1.9 Added,
,3,3,Worse than A, , , , , , , , , ,$,v05.1.9 Added,
,2,145,AHU-LEAKAGE,4,Worse than L3, , , , , , ,$,v05.1.9 Added,
,3,1,Class L1, , , , , , , , , ,$,v05.1.9 Added,
,3,2,Class L2, , , , , , , , , ,$,v05.1.9 Added,
,3,3,Class L3, , , , , , , , , ,$,v05.1.9 Added,
,3,4,Worse than L3, , , , , , , , , ,$,v05.1.9 Added,
,2,146,CTRL-METERING,4,NO, , , , , , ,$,v05.1.9 Added,
,3,1,YES, , , , , , , , , ,$,v05.1.9 Added,
,3,2,NO, , , , , , , , , ,$,v05.1.9 Added,
,2,147,CTRL-ALARM,4,NO, , , , , , ,$,v05.1.9 Added,
,3,1,YES, , , , , , , , , ,$,v05.1.9 Added,
,3,2,NO, , , , , , , , , ,$,v05.1.9 Added,
,2,148,CHILLER-TYPE,4,Not applicable, , , , , , ,$,v05.1.9 Added,
,3,1,Air cooled chiller, , , , , , , , , ,$,v3.2.c_R09 Mod,
,3,2,Water cooled chiller, , , , , , , , , ,$,v3.2.c_R09 Mod,
,3,3,Remote-condenser chiller, , , , , , , , , ,$,v3.2.c_R09 Mod,
,3,4,Not applicable, , , , , , , , , ,$,v05.2.0 Added,
,3,5,Heat pump (gas/oil), , , , , , , , , ,$,v3.2.c_R09 Mod,
,3,6,Heat pump (electric), , , , , , , , , ,$,v3.2.c_R09 Mod,
,2,149,CHILLER-KW,4,Not applicable, , , , , , ,$,v05.1.9 Added,
,3,1,Up to 100kW, , , , , , , , , ,$,v05.1.9 Added,
,3,2,101 to 500kW, , , , , , , , , ,$,v05.1.9 Added,
,3,3,501 to 750kW, , , , , , , , , ,$,v05.1.9 Added,
,3,4,751 to 3.5MW, , , , , , , , , ,$,v05.1.9 Added,
,3,5,Not applicable, , , , , , , , , ,$,v05.2.0 Added,
,2,150,ECA,4,Not applicable, , , , , , ,$,v05.1.9 Added,
,3,1,Not in the ECA list, , , , , , , , , ,$,v05.1.9 Added,
,3,2,ECA list (after 2001), , , , , , , , , ,$,v05.1.9 Added,
,3,3,ECA list (after 2002), , , , , , , , , ,$,v05.1.9 Added,
,3,4,ECA list (after 2004), , , , , , , , , ,$,v05.1.9 Added,
,3,5,ECA list (before 2005), , , , , , , , , ,$,v05.1.9 Added,
,3,6,ECA list (from 2005), , , , , , , , , ,$,v05.1.9 Added,
,3,7,ECA list (increases in 2006), , , , , , , , , ,$,v05.1.9 Added,
,3,8,ECA list (uplift in 2006), , , , , , , , , ,$,v05.1.9 Added,
,3,9,Not applicable, , , , , , , , , ,$,v05.2.0 Added,
```

```
,2,151,BOILER-POST98,4,Not applicable, , , , , , ,$,v05.1.9 Added,
,3,1,YES, , , , , , , , ,$,v05.1.9 Added,
,3,2,NO, , , , , , , , ,$,v05.1.9 Added,
,3,3,Not applicable, , , , , , , , ,$,v05.2.0 Added,
,2,152,HEAT-EFF,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.1.9 Added,
,2,153,COOL-EFF,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.0.b Added,
,2,154,METERING,1,1,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.2 Moved to compliance,
,2,155,HVACGUIDE-HEAT,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.2.a Added,
,2,156,HVACGUIDE-COOL,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.2.a Added,
,2,157,HVACGUIDE-SFP,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.2.a Added,
,2,158,SFP-CHECK,4,NO, , , , , , ,$,v1.2.a Added,
,3,1,YES, , , , , , , , ,$,v1.2.a Added,
,3,2,NO, , , , , , , , ,$,v1.2.a Added,
,2,159,AUX-ENERGY/KWH,1,0.02,0,-6666,0.001,1,-6666,-6666,NC,$,v2.0.a Added,
,2,160,REC-PERFORMANCE-HEAT,4,Not applicable, , , , , , ,$,v2.0.d_dev_R22 Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22 Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22 Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22 Added,
,3,4,Not applicable, , , , , , , , ,$,v2.0.d_dev_R22 Added,
,2,161,REC-PERFORMANCE-COOL,4,Not applicable, , , , , , ,$,v2.0.d_dev_R22 Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22 Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22 Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22 Added,
,3,4,Not applicable, , , , , , , , ,$,v2.0.d_dev_R22 Added,
,2,162,DHW-SERVED-REF,6,{ " " },DHW-GENERATOR, , , , , ,$,v2.0.d_dev_R22 Added,
,2,163,EFF-DEFAULT-COOL,4,YES, , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,1,YES, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,NO, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,164,EFF-DEFAULT-HEAT,4,YES, , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,1,YES, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,NO, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,165,CEN-TIME-CTRL,4,NO, , , , , , ,$,v2.0.d_dev_R39_CD Added,
,3,1,YES, , , , , , , , ,$,v2.0.d_dev_R39_CD Added,
,3,2,NO, , , , , , , , ,$,v2.0.d_dev_R39_CD Added,
,2,166,ROOM-TIME-CTRL,4,NO, , , , , , ,$,v2.0.d_dev_R39_CD Added,
,3,1,YES, , , , , , , , ,$,v2.0.d_dev_R39_CD Added,
,3,2,NO, , , , , , , , ,$,v2.0.d_dev_R39_CD Added,
,2,167,ROOM-TEMP-CTRL,4,NO, , , , , , ,$,v2.0.d_dev_R39_CD Added,
,3,1,YES, , , , , , , , ,$,v2.0.d_dev_R39_CD Added,
,3,2,NO, , , , , , , , ,$,v2.0.d_dev_R39_CD Added,
,2,168,OPTIMUM-START,4,NO, , , , , , ,$,v2.0.d_dev_R39_CD Added,
,3,1,YES, , , , , , , , ,$,v2.0.d_dev_R39_CD Added,
,3,2,NO, , , , , , , , ,$,v2.0.d_dev_R39_CD Added,
,2,169,WEATHER-COMP,4,NO, , , , , , ,$,v2.0.d_dev_R39_CD Added,
,3,1,YES, , , , , , , , ,$,v2.0.d_dev_R39_CD Added,
,3,2,NO, , , , , , , , ,$,v2.0.d_dev_R39_CD Added,
,2,170,EPC-F1,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47
Modified,
,2,171,EPC-F2,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47
Modified,
,2,172,EPC-F3,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47
Modified,
,2,173,EPC-F4,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47
Modified,
,2,174,EPC-F5,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47
Modified,
,2,175,EPC-F6,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47
Modified,
,2,176,EPC-H1,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47
Modified,
,2,177,EPC-H3,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47
```

```
Modified,
,2,178,EPC-H4,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R01_CD Added ,
,2,179,EPC-R1,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47
Modified,
,2,180,EPC-R5,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47
Modified,
,2,181,EPC-H2,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47
Modified,
,2,182,EPC-H5,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47
Modified,
,2,183,EPC-H6,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47
Modified,
,2,184,EPC-H7,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47
Modified,
,2,185,EPC-H8,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47
Modified,
,2,186,EPC-C1,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R01_CD Added ,
,2,187,EPC-C2,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47
Modified,
,2,188,EPC-C3,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47
Modified,
,2,189,MIXED-MODE,4,NO,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_R24 Added,
,3,1,YES, , , , , , , , ,$,v3.0.a_R24 Added,
,3,2,NO, , , , , , , , ,$,v3.0.a_R24 Added,
,2,190,VARIABLE-SPEED-PUMP,4,NO,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09_a01
Modified,
,3,1,NO, , , , , , , , ,$,v4.0.c_dev_R09_a01 Modified,
,3,2,LTHW, , , , , , , , ,$,v4.0.c_dev_R09_a01 Modified,
,3,3,BOTH, , , , , , , , ,$,v4.0.c_dev_R09_a01 Modified,
,2,191,VARIABLE-SPEED-PUMP-TYPE,4,With differential sensor across pump,-6666,-6666,-6666,-
6666,-6666,-6666,NC,$,v4.0.a_dev_R29_a05 Added,
,3,1,With differential sensor across pump, , , , , , , , ,$,v4.0.a_dev_R29_a05 Added,
,3,2,With differential sensor in the system, , , , , , , , ,$,v4.0.a_dev_R29_a05 Added,
,3,3,With multiple pressure sensors in the system, , , , , , , , ,$,v4.0.a_dev_R29_a05 Added,
,2,192,BH-HEAT-SOURCE,7,{ "-5555" },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R29
Added,
,2,193,BH-FUEL-TYPE,7,{ "-5555" },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R29 Added,
,2,194,BH-HEAT-SSEFF,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R29 Added,
,2,195,BH-HEAT-GEN-SEFF,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v4.0.a_dev_R29_a03 Added,
,2,196,BH-LOAD,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R29 Added,
,2,197,ACT-ELEC-HEAT,4,NO,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R40_a02 Added,
,3,1,YES, , , , , , , , ,$,v4.0.a_dev_R40_a02 Added,
,3,2,NO, , , , , , , , ,$,v4.0.a_dev_R40_a02 Added,
,1,2,CONSTRUCTION,1,-6666, ,1, , , , , ,$, ,
,2,1,TYPE,4,U_VALUE, , , , , , , ,$, ,
,3,1,U_VALUE, , , , , , , , ,$, ,
,3,2,LAYERS, , , , , , , , ,$, ,
,3,3,CT_003, , , , , , , , ,$, ,
,2,2,U-VALUE,1,REQ,0,10,0.05,6.5,0,10,W/m2K,$, ,
,2,3,U-VALUE-CORR,4,NO, , , , , , ,$,v2.0.d Added,
,3,1,YES, , , , , , , , ,$,v2.0.d Added,
,3,2,NO, , , , , , , , ,$,v2.0.d Added,
,2,4,CM,5,REQ,0,-6666,0.1,200,-6666,-6666,kJ/m2/K,$, ,
,2,5,METAL-CLADDING,4,NO, , , , , , ,$,v1.1.a Added,
,3,1,YES, , , , , , , , ,$,v1.1.a Added,
,3,2,NO, , , , , , , , ,$,v1.1.a Added,
,2,6,USED-IN-GEOMETRY,4,NO, , , , , , ,$,v3.4.b_dev_R12_a01 Added,
,3,1,YES, , , , , , , , ,$,v3.4.b_dev_R12_a01 Added,
,3,2,NO, , , , , , , , ,$,v3.4.b_dev_R12_a01 Added,
,1,3,GLASS,1,-6666, ,1, , , , , ,$, ,
```

```
,2,1,TYPE,4,GT_001, , , , , , ,$, ,
,3,1,GT_001, , , , , , , , ,$, ,
,3,2,GT_002, , , , , , , , ,$, ,
,3,3,GT_003, , , , , , , , ,$, ,
,2,2,U-VALUE,1,REQ,0,10,0.05,6.5,0,10,W/m2K,$, ,
,2,3,R,1,NC,0,10,0,10,0,10, ,$, ,
,2,4,ENERGY-TRANS,1,1,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.6.4.2 Is the total solar energy transmittance, 6.9.3 of NEN5128,
,2,5,TOT-SOL-TRANS,1,1,0,-6666,0.2,1,-6666,-6666,NC,$, ,
,2,6,LIG-SOL-TRANS,1,1,0,-6666,0.2,1,-6666,-6666,NC,$, ,
,2,7,DIR-SOL-TRANS,1,1,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,8,USED-IN-GEOMETRY,4,NO, , , , , , ,$,v3.4.b_dev_R12_a01 Added,
,3,1,YES, , , , , , , , ,$,v3.4.b_dev_R12_a01 Added,
,3,2,NO, , , , , , , , ,$,v3.4.b_dev_R12_a01 Added,
,1,4,WALL,0,-6666,ZONE,1, , , , ,$, ,
,2,1,TYPE,4,REQ, , , , , , ,$, ,
,3,1,Exterior, , , , , , , , ,$, ,
,3,2,Strongly ventilated spaces, , , , , , , , ,$, ,
,3,3,Unheated adjoining space, , , , , , , , ,$, ,
,3,4,Conditioned adjoining space, , , , , , , , ,$, ,
,3,5,Underground, , , , , , , , ,$, ,
,3,6,Same space, , , , , , , , ,$,v3.4.b_dev_R21_a05 Added,
,2,2,AREA,1,REQ,0.00001,1000000000,0.1,5000,0,1000000000,m2,$, ,
,2,3,NET-AREA,1,EXP,0,1000000000,0,1000000000,0,1000000000,m2,$,v04.3.3 Added,
,2,4,HEIGHT,1,EXP,0,1000000000,0.1,5000,0,1000000000,m,$,v3.0.a_R23 Added,
,2,5,LENGTH,1,1000,0,1000000000,0.1,5000,0,1000000000,m,$,v3.0.a_R23 Added v4.0.a_dev_R17 mod dev_R27 mod,
,2,6,PITCH,1,45,0,90,0,90,0,90,degrees,$,v4.0.a_dev_R26,
,2,7,ROOF-PITCH,4,45,-6666,-6666,-6666,-6666,-6666,-6666,degrees,$,v4.0.a_dev_R16 Added - v4.0.a_dev_R23_a02 Modified,
,3,1,0, , , , , , , , ,$, ,
,3,2,10, , , , , , , , ,$, ,
,3,3,15, , , , , , , , ,$, ,
,3,4,20, , , , , , , , ,$, ,
,3,5,25, , , , , , , , ,$, ,
,3,6,30, , , , , , , , ,$, ,
,3,7,35, , , , , , , , ,$, ,
,3,8,40, , , , , , , , ,$, ,
,3,9,45, , , , , , , , ,$, ,
,3,10,50, , , , , , , , ,$, ,
,3,11,55, , , , , , , , ,$, ,
,3,12,60, , , , , , , , ,$, ,
,3,13,65, , , , , , , , ,$, ,
,3,14,70, , , , , , , , ,$, ,
,3,15,90, , , , , , , , ,$, ,
,2,8,U-VALUE,1,EXP,0,10,0,10,0,10, ,$, ,
,2,9,U-VALUE-CORR,4,NO, , , , , , ,$,v2.0.d Added,
,3,1,YES, , , , , , , , ,$,v2.0.d Added,
,3,2,NO, , , , , , , , ,$,v2.0.d Added,
,2,10,U-VALUE-WALL/WIN,1,EXP,0,10,0,10,0,10, ,$,v05.0.0 Added,
,2,11,CM,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,12,CONSTRUCTION,3,REQ,CONSTRUCTION, , , , , ,$, ,
,2,13,A,1,EXP,0,10,0,10,0,10, ,$, ,
,2,14,H-TR,1,EXP,0,1000000000,0,1000000000,0,1000000000, ,$, ,
,2,15,ORIENTATION,4,South, , , , , , ,$, ,
,3,1,South, , , , , , , , ,$, ,
,3,2,South-East, , , , , , , , ,$, ,
,3,3,East, , , , , , , , ,$, ,
,3,4,North-East, , , , , , , , ,$, ,
,3,5,North, , , , , , , , ,$, ,
,3,6,North-West, , , , , , , , ,$, ,
```

,3,7,West, , , , , , , , ,$, ,
,3,8,South-West, , , , , , , , ,$, ,
,3,9,Horizontal, , , , , , , , ,$, ,
,2,16,Q-SUN-NT,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,10.4.2 Is the solar heat gain through non transparent construction in MJ,
,2,17,TYPE-ENV,4,REQ,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added,
,3,1,Wall, , , , , , , , ,$,v4.3.0 Added,
,3,2,Roof, , , , , , , , ,$,v4.3.0 Added,
,3,3,Floor or Ceiling, , , , , , , , ,$,v4.3.0 Added,
,2,18,FLAT-ROOF,4,NO, , , , , , , ,$,v4.3.0 Added,
,3,1,YES, , , , , , , , ,$,v4.3.0 Added,
,3,2,NO, , , , , , , , ,$,v4.3.0 Added,
,2,19,U-INF-WIND-WALL,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,20,U-INF-STACK-WALL,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.3.a Added,
,2,21,MULTIPLIER,1,1,1,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.4 Added,
,2,22,ABSORP-COEFF,1,0.9,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.4 Added,
,2,23,TM-TYPE,4,ALPHA, , , , , , ,$,v5.1.9 Added,
,3,1,ALPHA, , , , , , , , ,$,v5.1.9 Added,
,3,2,TM, , , , , , , , ,$,v5.1.9 Added,
,2,24,ALPHA,1,0.15,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.01.9 Added,
,2,25,CPS-COMP-FLAG,4,YES, , , , , , ,$,v2.0.c_R05,
,3,1,YES, , , , , , , , ,$,v2.0.c_R05,
,3,2,NO, , , , , , , , ,$,v2.0.c_R05,
,2,26,CAVITY-WALL,4,YES, , , , , , ,$,v3.0.a_dev_R04_CD Added,
,3,1,YES, , , , , , , , ,$,v3.0.a_dev_R04_CD Added,
,3,2,NO, , , , , , , , ,$,v3.0.a_dev_R04_CD Added,
,2,27,PITCH-LOFT,4,YES, , , , , , ,$,v3.0.a_dev_R04_CD Added,
,3,1,YES, , , , , , , , ,$,v3.0.a_dev_R04_CD Added,
,3,2,NO, , , , , , , , ,$,v3.0.a_dev_R04_CD Added,
,2,28,TSC-SYSTEM,4,NO, , , , , , ,$,v2.0.d Added,
,3,1,YES, , , , , , , , ,$,v2.0.d Added,
,3,2,NO, , , , , , , , ,$,v2.0.d Added,
,2,29,TSC,3,EXP,TSC, , , , , ,$, ,
,2,30,TSC-AREA,1,EXP,0.00001,1000000000,0.1,5000,0,1000000000,m2,$, ,
,1,5,WALL-TM-BRIDGE,0,-6666,WALL,1, , , , ,$, ,
,2,1,PSI,1,2,-6666,-6666,0.01,2.5,-6666,-6666,W/mK,$,v4.3.0 Modified,
,2,2,L,1,10,-6666,-6666,0.01,1000,-6666,-6666,m,$, ,
,2,3,H-TR,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666, ,$, ,
,2,4,MULTIPLIER,1,1,1,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.4 Added,
,1,6,WINDOW,0,-6666,WALL,1, , , , ,$, ,
,2,1,AREA,1,REQ,0.00001,1000000000,0.1,5000,0,1000000000,m2,$, ,
,2,2,AREA-G,1,EXP,0,1000000000,0.1,5000,0,1000000000,m2,$,2010 1.0.0_dev_R01 Added,
,2,3,DEV-PROJ-RATIO,1,-6666,1,10,1,2,1,10, ,$,v3.4.b_dev_R14_a01 Modified,
,2,4,ROOF-LIGHT-COVER,1,1,1,4,1,4,1,4,ratio,$,v3.0.a_R23 Added,
,2,5,U-VALUE,1,EXP,0,10,0,10,0,10, ,$, ,
,2,6,U-VALUE-VERT,1,EXP,0,10,0,10,0,10, ,$, ,
,2,7,GLASS,3,REQ,GLASS, , , , , ,$, ,
,2,8,TOT-SOL-TRANS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,9,LIG-SOL-TRANS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,10,LIG-SOL-TRANS-ORI,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R03 Added,
,2,11,H-TR,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666, ,$, ,
,2,12,R,1,EXP,0,10,0,10,0,10, ,$, ,
,2,13,SHADING-SYSTEM,4,All other cases, , , , , , ,$,6.6.4.2 Table 11,
,3,1,Ext. solar protection. User moveable, , , , , , , , ,$, ,
,3,2,Ext. solar protection with automatic control, , , , , , , , ,$, ,
,3,3,All other cases, , , , , , , , ,$, ,
,2,14,Q-SUN-T,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.6.4.1 Is the solar heat gain through transparent construction in MJ,
,2,15,SHADING-FACTOR,1,1,0.1,1,0.1,1,-6666,-6666,NC,$,6.6.4.4 Is the shading correction factor… NEN5128 or NEN1068,
,2,16,DISP-WIND,4,NO, , , , , , ,$,v4.3.0 Added,

```
,3,1,YES, , , , , , , , ,$,v4.3.0 Added,
,3,2,NO, , , , , , , , ,$,v4.3.0 Added,
,2,17,FRAME-FACTOR,1,-6666,0,1,0,1,0,1,ratio,$,v3.4.b_dev_R13_a12 added,
,2,18,MULTIPLIER,1,1,1,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.4 Added,
,2,19,TM-TYPE,4,ALPHA, , , , , , ,$,v5.1.9 Added,
,3,1,ALPHA, , , , , , , , ,$,v5.1.9 Added,
,3,2,TM, , , , , , , , ,$,v5.1.9 Added,
,2,20,ALPHA,1,0.15,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.01.9 Added,
,2,21,CPS-COMP-FLAG,4,YES, , , , , , ,$,v2.0.c_R05,
,3,1,YES, , , , , , , , ,$,v2.0.c_R05,
,3,2,NO, , , , , , , , ,$,v2.0.c_R05,
,2,22,ASPECT-RATIO,1,-5555,-6666,-6666,-6666,-6666,-6666,-6666,ratio,$,v4.0.a_dev_R56_a01 Added,
,1,7,WINDOW-TM-BRIDGE,0,-6666,WINDOW,1, , , , , ,$, ,
,2,1,PSI,1,2,-6666,-6666,0.01,2.5,-6666,-6666,W/mK,$,v4.3.0 Modified,
,2,2,L,1,10,-6666,-6666,0.01,1000,-6666,-6666,m,$, ,
,2,3,H-TR,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666, ,$, ,
,2,4,MULTIPLIER,1,1,1,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.4 Added,
,1,8,SPACE,0,-6666,HVAC-SYSTEM,0, , , , ,$, ,
,2,1,TYPE,4,REQ, , , , , , ,$, ,
,2,2,AREA,1,REQ,0.00001,1000000000,0.001,1000000000,0,1000000000,m2,$, ,
,2,3,H-TR,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666, ,$, ,
,1,9,ZONE,1,-6666,HVAC-SYSTEM,0, , , , ,$, ,
,2,1,ISBEM-ID,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a Added_R21,
,2,2,TYPE,4,NC, , , , , , ,$,Type of zone,
,3,1,CONDITIONED, , , , , , , , ,$, ,
,3,2,UNCONDITIONED, , , , , , , , ,$, ,
,3,3,PLENUM, , , , , , , , ,$, ,
,2,3,ACTIVITY,1,0,0,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.1 Added,
,2,4,ACTIVITY-NAME,2,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.1 Added,
,2,5,SERVICE-TYPE,4,EITHER,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.3.c_dev_R12 Added,
,3,1,CONDITIONED, , , , , , , , ,$,v3.3.c_dev_R12 Added,
,3,2,UNCONDITIONED, , , , , , , , ,$,v3.3.c_dev_R12 Added,
,3,3,EITHER, , , , , , , , ,$,v3.3.c_dev_R12 Added,
,2,6,ACT-DBS-ACU,4,YES, , , , , , ,$,v1.1.a Added,
,3,1,YES, , , , , , , , ,$,v1.1.a Added,
,3,2,NO, , , , , , , , ,$,v1.1.a Added,
,2,7,AREA,1,REQ,0.00001,1000000000,1,10000,0,1000000000,m2,$,Is the useful area of the energy sector ,
,2,8,AREA-SUR-EXT,1,EXP,0,1000000000,0,1000000000,0,1000000000,m2,$,v1.2.a Added,
,2,9,AREA-WALL,1,EXP,0,1000000000,0,1000000000,0,1000000000,m2,$,2010 1.0.0_dev_R01 Added,
,2,10,AREA-ROOF,1,EXP,0,1000000000,0,1000000000,0,1000000000,m2,$,2010 1.0.0_dev_R01 Added,
,2,11,AREA-FLOOR,1,EXP,0,1000000000,0,1000000000,0,1000000000,m2,$,2010 1.0.0_dev_R01 Added,
,2,12,EXPOSED-FACADE-
WIDTH,1,EXP,0,1000000000,0,1000000000,0,1000000000,m,$,v4.0.a_dev_R23_a05 Added,
,2,13,HEIGHT,1,3,-6666,-6666,1.5,25,-6666,-6666,m ,$,v4.3.0 Added,
,2,14,SERVICE-STRATEGY,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.2.a Added,
,2,15,R-GEN-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.8,
,2,16,R-GEN-COOL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,10.7,
,2,17,R-SYS-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.7 Is the distribution system
efficency for distribution of heat,
,2,18,R-SYS-COOL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.7 Is the distribution system
efficency for distribution of cool,
,2,19,Q-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.3.2.1 is the energy consumption for
heating per year, in MJ,
,2,20,Q-HEAT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.3.2.2 Is the energy consumption for
heating in the month, in MJ,
,2,21,Q-DEM-HEAT-ROOM,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,22,Q-DEM-HEAT-M-ROOM,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.3.2.4 is the heat
demand in the month at room level, in MJ,
,2,23,HEAT-INTERMIT-FLAG,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.b_dev_R70_a02
Added,
,2,24,HEAT-INTERMIT-RED,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09_a05
```

Added,
,2,25,Q-COOL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,26,Q-COOL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,27,Q-DEM-COOL-ROOM,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,10.3.4,
,2,28,Q-DEM-COOL-M-ROOM,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,10.3.4,
,2,29,Q-DEM-COOL-ROOM-OH,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R05 Added,
,2,30,Q-DEM-COOL-M-ROOM-OH,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R05 Added,
,2,31,COOL-INTERMIT-RED,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09_a05 Added,
,2,32,Q-TR-HEAT,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.4.1 is the transmision heat loss per month in MJ,
,2,33,Q-TR-COOL,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,10.5.2 is the transmision heat loss per month in MJ,
,2,34,Q-TR-COOL-OH,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R05 Added,
,2,35,H-TR,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.4.2 is the transmision heat loss coefficient, in W/K,
,2,36,Q-VENT-HEAT,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.5.1 is the ventilation heat loss in MJ,
,2,37,Q-VENT-COOL,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,10.5.3 is the ventilation heat loss in MJ,
,2,38,Q-VENT-COOL-NC,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,MJ,$,v4.0.b_dev_R64_a01 added,
,2,39,NIGHT-COOLING,4,NO, , , , , , ,$,v4.0.b_dev_R64_a04 added,
,3,1,YES, , , , , , , , ,$,v4.0.b_dev_R64_a04 added,
,3,2,NO, , , , , , , , ,$,v4.0.b_dev_R64_a04 added,
,2,40,MAX-FLOW-NC,1,1,-6666,-6666,-6666,-6666,-6666,-6666,l/s.m2,$,v4.0.b_dev_R64_a01 added,
,2,41,MAX-HOURS-NC,1,200,-6666,-6666,-6666,-6666,-6666,-6666,hours/month,$,v4.0.b_dev_R64_a01 added,
,2,42,SFP-NC,1,2.5,-6666,-6666,-6666,-6666,-6666,-6666,W/(l/s),$,v4.0.b_dev_R64_a04 added,
,2,43,OPT-FLOW-NC,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,l/s,$,v4.0.b_dev_R64_a01 added,
,2,44,RATIO-OPT-MAX-NC,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,raio,$,v4.0.b_dev_R64_a01 added,
,2,45,Q-VENT-COOL-OH,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R05 Added,
,2,46,Q-INF-HEAT,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.1.9,
,2,47,Q-INF-COOL,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.1.9,
,2,48,Q-INF-COOL-OH,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R05 Added,
,2,49,F-VENT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,50,U-V-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Modified,
,2,51,U-V-N,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.2.a Added,
,2,52,H-VENT-HEAT,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.5.1 Is the ventilation heat loss coefficient tof the energy sector,
,2,53,H-VENT-COOL,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,10.5.3.1 Is the ventilation heat loss coefficient tof the energy sector,
,2,54,H-VENT-COOL-NC,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.b_dev_R64_a01 added,
,2,55,H-INF-HEAT,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.1.9,
,2,56,H-INF-COOL,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.1.9,
,2,57,Q-V-M-MAX,1,5,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,Revised v05.1.9: now is in ACH. 6.5.3.6 maximum supply capacity of the mechanical ventilation system as far as serving ventilation flows within the energy sectors simultaneously, in dm3/s (5.1 NEN 1087),
,2,58,Q-V-M-EXH,1,5,-6666,-6666,0.1,50,-6666,-6666,l/s.m2,$,v1.2.a Added,
,2,59,Q-V-TYPE,4,Mechanical, , , , , , ,$,v04.3.4 Modified ID1,
,3,1,Natural Supply, , , , , , , , ,$,v04.3.4 Modified ID1,
,3,2,Mechanical, , , , , , , , ,$,v04.3.4 Modified ID1,
,3,3,Not used, , , , , , , , ,$,v04.3.4 Modified ID1,
,2,60,Q-V-TYPE1,4,Other cases, , , , , , ,$,6.5.3.4 Computation values of the specific air flow due to mechanical ventialtion,
,3,1,Natural Supply and Mechanical exhaust, , , , , , , , ,$,In case of systems with natural supply and mechanical exhaust,
,3,2,Other cases, , , , , , , , ,$,All other cases,
,2,61,Q-V-FACILITY,4,None, , , , , , ,$,10.5.3.3 T026 Ventilation facility in cooling season.,

,3,1,None, , , , , , , , , ,$, ,
,3,2,Register, , , , , , , , , ,$, ,
,3,3,Tilt windows or top swing windows, , , , , , , , , ,$, ,
,3,4,Turn windows + tilt and turn windows, , , , , , , , , ,$, ,
,2,62,F-CTRL-VENT-TYPE,4,no recirculation, , , , , , ,$,v04.3.4 revised 6.5.3.7 Factor for the reduction of the supply of fresh air,
,3,1,no recirculation, , , , , , , , ,$,v04.3.4 Added ID2,
,3,2,less than 10%, , , , , , , , ,$,v04.3.4 Added ID2,
,3,3,10% to 20%, , , , , , , , ,$,v04.3.4 Added ID2,
,3,4,20% to 30%, , , , , , , , ,$,v04.3.4 Added ID2,
,3,5,30% to 40%, , , , , , , , ,$,v04.3.4 Added ID2,
,3,6,40% to 50%, , , , , , , , ,$,v04.3.4 Added ID2,
,3,7,50% to 60%, , , , , , , , ,$,v04.3.4 Added ID2,
,3,8,60% to 70%, , , , , , , , ,$,v04.3.4 Added ID2,
,3,9,70% to 80%, , , , , , , , ,$,v04.3.4 Added ID2,
,3,10,80% to 90%, , , , , , , , ,$,v04.3.4 Added ID2,
,3,11,more than 90%, , , , , , , , ,$,v04.3.4 Added ID2,
,2,63,F-CTRL-VENT-DEM,4,No demand-controlled ventilation, , , , , , ,$,cSBEM_v1.0.0_dev_R27_a02 modified,
,3,1,No demand-controlled ventilation, , , , , , , , ,$,cSBEM_v1.0.0_dev_R27_a02 modified,
,3,2,Demand control dependent on number of occupants, , , , , , , , ,$,cSBEM_v1.0.0_dev_R27_a02 modified,
,3,3,Demand control dependent on gas sensors, , , , , , , , ,$,cSBEM_v1.0.0_dev_R27_a02 modified,
,3,4,Enhanced ventilation, , , , , , , , ,$,cSBEM_v1.0.0_dev_R27_a02 modified,
,2,64,FLOW-REGU-TYPE,4,Damper control, , , , , , ,$,cSBEM_v1.0.0_dev_R27_a02 added,
,3,1,Damper control, , , , , , , , ,$,cSBEM_v1.0.0_dev_R27_a02 added,
,3,2,Speed control, , , , , , , , ,$,cSBEM_v1.0.0_dev_R27_a02 added,
,2,65,HEAT-REC-SYSTEM,4,No heat recovery, , , , , , ,$,v4.3.0 supersided - v04.3.4 - reused ID1,
,3,1,No heat recovery, , , , , , , , ,$,v4.3.0 supersided - v04.3.4 - reused ID1,
,3,2,Plate heat exchanger (Recuperator), , , , , , , , ,$,v4.3.0 supersided - v04.3.4 - reused ID1,
,3,3,Two-elements-system, , , , , , , , ,$,v4.3.0 supersided - v04.3.4 - reused ID1,
,3,4,Loading cold with air-conditioning, , , , , , , , ,$,v4.3.0 supersided - v04.3.4 - reused ID1,
,3,5,Heat-pipes, , , , , , , , ,$,v4.3.0 supersided - v04.3.4 - reused ID1,
,3,6,Thermal wheel, , , , , , , , ,$,v4.3.0 supersided - v04.3.4 - reused ID1,
,3,7,Run around coil, , , , , , , , ,$,v4.3.0 supersided - v04.3.4 - reused ID1,
,2,66,HEAT-REC-SEFF,1,0,0,1,0,1,0,1,NC,$,v2.0.a Added,
,2,67,HEAT-REC-VAR-EFF,4,NO, , , , , , ,$,v4.0.b_dev_R66_a01 added,
,3,1,YES, , , , , , , , ,$,v4.0.b_dev_R66_a01 added,
,3,2,NO, , , , , , , , ,$,v4.0.b_dev_R66_a01 added,
,2,68,INF-HEIGHT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,Table 4: Computation value for the specific air flow per m2… See figure 4. v4.3.0 supersided,
,2,69,Q-I,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.6.3 Is the internal heat production in MJ,
,2,70,Q-I-LI,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 equal to 6.6.3.2 Is the internal heat production by lighting in MJ per month,
,2,71,Q-PRIM-LI-MET,4,SPC-ELEC, , , , , , ,$,v4.3.0 Supersided,
,3,1,SPC-ELEC, , , , , , , , ,$,v4.3.0 Supersided,
,3,2,REAL-LOAD-70%, , , , , , , , ,$,v4.3.0 Supersided,
,3,3,REAL-LOAD, , , , , , , , ,$,v4.3.0 Supersided,
,2,72,LIGHT-CTRL-TYPE,4,ROOM, , , , , , ,$,v4.3.0 Supersided,
,3,1,SWEEPING-DAYLIGHT, , , , , , , , ,$,v4.3.0 Supersided,
,3,2,DAYLIGHT, , , , , , , , ,$,v4.3.0 Supersided,
,3,3,SWEEPING, , , , , , , , ,$,v4.3.0 Supersided,
,3,4,ROOM, , , , , , , , ,$,v4.3.0 Supersided,
,3,5,ROOM-FACADE, , , , , , , , ,$,v4.3.0 Supersided,
,3,6,CENTRAL, , , , , , , , ,$,v4.3.0 Supersided,
,2,73,Q-I-VENT,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.6.3.3 Is the internal heat production by fans in MJ per month,
,2,74,Q-PRIM-VENT,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,7.2 It has to be EXP, but not yet developed, so DEF to allow Q-I-VENT to be calculated before Q-PRIM-VENT,
,2,75,F-I-VENT-TYPE,4,OTHER, , , , , , ,$,6.6.3.3 Is a reduction factor for which has to be taken,
,3,1,M-EXH-ONLY, , , , , , , , ,$,T052,

,3,2,M-SUP-EXH, , , , , , , , ,$, ,
,3,3,REC-HEAT-REC, , , , , , , , ,$, ,
,3,4,M-SUP-FCAP, , , , , , , , ,$, ,
,3,5,OTHER, , , , , , , , ,$, ,
,2,76,Q-I-PERS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.6.3.1. Is the specific internal heat production in the considered energy sector by persons in W/m2. T008,
,2,77,Q-I-PERS-CLASS,4,I, , , , , , ,$,T008. v4.3.0 supersided,
,3,1,I, , , , , , , , ,$,v4.3.0 supersided,
,3,2,II, , , , , , , , ,$,v4.3.0 supersided,
,3,3,III, , , , , , , , ,$,v4.3.0 supersided,
,3,4,IV, , , , , , , , ,$,v4.3.0 supersided,
,3,5,V, , , , , , , , ,$,v4.3.0 supersided,
,2,78,Q-I-APP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.6.3.1. Is the specific internal heat production in the considered energy sector by the average capacity of other appliances in W/m2. T009,
,2,79,Q-SUN,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.6.4 Is the solar heat gain in MJ,
,2,80,Q-SUN-T,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.6.4.1 Is the solar heat gain through transparent construction in MJ,
,2,81,Q-SUN-NT,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,10.4.2 Is the solar heat gain through non transparent construction in MJ,
,2,82,Q-SUN-SES,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.6.4.2 Is the solar heat gain through solar energy systems in MJ,
,2,83,A-SES,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.6.4.3 Is the area in m2 of the smallest plane that connects the collector borders.,
,2,84,R-B-HEAT,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.6.5.2 Is the utilisation factor for heat gain,
,2,85,R-B-COOL,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,10.6.1 Is the utilisation factor for heat gain,
,2,86,A-FACTOR-COOL,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2.0.b_dev_R02 added,
,2,87,R-B-COOL-OH,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R05 Added,
,2,88,A-FACTOR-COOL-OH,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R05 Added,
,2,89,GAIN-LOSS-RATIO-H,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.6.5.2 Is the gain-loss ratio,
,2,90,GAIN-LOSS-RATIO-C,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,10.6.1 Is the gain-loss ratio,
,2,91,GAIN-LOSS-RATIO-C-OH,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R05 Added,
,2,92,Q-GAIN-HEAT,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,6.6. Is the utilized heat gain, in MJ;,
,2,93,Q-GAIN-COOL,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,10.4 Is the utilized heat gain, in MJ;,
,2,94,Q-LOSS-COOL,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,10.5 Is the heat loss per month for the cooling demand calculation, in MJ;,
,2,95,Q-LOSS-COOL-OH,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R05 Added,
,2,96,P-EFF,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,7.2.2 Is the computation value of the effective power in kW,
,2,97,P-EFF-AGR,1,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,98,COLD-GENERATOR,6,{ " " },COLD-GENERATOR, , , , , ,$, ,
,2,99,COLD-GEN-PREF,1,1,1,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,100,GEN-COOL-PROP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,101,Q-HUM,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,11,
,2,102,HUM-TYPE,4,ELEC, , , , , , ,$,11,
,3,1,ELEC, , , , , , , , ,$, ,
,3,2,NON-ELEC, , , , , , , , ,$, ,
,2,103,HUM-RECOV,4,NO, , , , , , ,$, ,
,3,1,YES, , , , , , , , ,$, ,
,3,2,NO, , , , , , , , ,$, ,
,2,104,HEAT-FLOW-TYPE,4,OTHER, , , , , , ,$,9,
,3,1,SPEED, , , , , , , , ,$, ,
,3,2,NO-PUMP, , , , , , , , ,$, ,
,3,3,OTHER, , , , , , , , ,$, ,
,2,105,COOL-FLOW-TYPE,4,OTHER, , , , , , ,$, ,

```
,3,1,SPEED, , , , , , , , ,$, ,
,3,2,NO-PUMP, , , , , , , , ,$, ,
,3,3,OTHER, , , , , , , , ,$, ,
,2,106,Q-PUMP-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,11,
,2,107,Q-PUMP-COOL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,11,
,2,108,DHW-GENERATOR,3," ",DHW-GENERATOR, , , , , ,$, ,
,2,109,Q-DHW,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,MJ,$,11,
,2,110,Q-DHW-CORR,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,MJ,$,v4.0.b_dev_R65_a01 addedd,
,2,111,DHW-SYS-DIST,4,CIRC-OR-UNKNOWN, , , , , , ,$,v04.3.3 Superseded,
,3,1,<3, , , , , , , , ,$,v04.3.3 Superseded,
,3,2,>3, , , , , , , , ,$,v04.3.3 Superseded,
,3,3,CIRC-OR-UNKNOWN, , , , , , , , ,$,v04.3.3 Superseded,
,2,112,Q-DHW-SYS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Superseded,
,2,113,DHW-DIST-LOSS,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,114,DHW-PIPE-LEN,1,0,0,-6666,0,5,-6666,-6666,m,$,v04.3.3 Added,
,2,115,PERS-HG-SEN,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,116,PERS-HG-LAT,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,117,EQUP-HG-SEN,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,118,EQUP-HG-LAT,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,119,LIGHT-LUX,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,120,LIGHT-DISP,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,121,HEAT-SET-POINT,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 REQ->NC,
,2,122,COOL-SET-POINT,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 REQ->NC,
,2,123,SB-SET-POINT,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Updated; v05.0.1 REQ-
>NC,
,2,124,HEAT-SET-AVE,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,125,HEAT-SET-POINT-ORI,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,126,COOL-SET-POINT-ORI,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,127,SB-SET-POINT-ORI,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,128,HEAT-SET-AVE-ORI,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,129,MAX-TIME-REDUCED,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added; v05.0.1
REQ->NC,
,2,130,MIN-TIME-REDUCED,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added; v05.0.1
REQ->NC,
,2,131,OA-FLOW-PERS,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,132,OA-FLOW-PERS-DES,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.2.a Added,
,2,133,OA-FLOW-PERS-MIN,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.2.a Added,
,2,134,DHW,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,135,Q50-INF,1,10,0,-6666,3,25,-6666,-6666,m3/h.m2,$, ,
,2,136,VENT-FRACT-ON,1,0.3,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,137,LIGHT-DF,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added,
,2,138,LIGHT-DF-F,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09 Added,
,2,139,LIGHT-DF-B,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09 Added,
,2,140,LIGHT-WIN/ROOF,4,NONE,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added,
,3,1,SIDE, , , , , , , , ,$,v4.3.0 Added,
,3,2,NONE, , , , , , , , ,$,v4.3.0 Added,
,3,3,OTHER, , , , , , , , ,$,v4.3.0 Added,
,2,141,LIGHT-T-START,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added; v05.0.1 REQ-
>NC,
,2,142,LIGHT-T-END,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added; v05.0.1 REQ-
>NC,
,2,143,OCC-T-START,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added; v05.0.1 REQ->NC,
,2,144,OCC-T-END,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added; v05.0.1 REQ->NC,
,2,145,OCC-T-HOURS,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.2.c Added,
,2,146,SYS-T-HOURS,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added; ID8,
,2,147,SYS-T-START,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added; ID8,
,2,148,SYS-T-END,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added; ID8,
,2,149,SYS-HEAT-T-HOURS,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.3 Added ,
,2,150,SYS-HEAT-T-START,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.3 Added ,
,2,151,SYS-HEAT-T-END,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.3 Added ,
,2,152,SYS-COOL-DAYS,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R02_a05 Added,
```

```
,2,153,LIGHT-PW,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.3 Added ,
,2,154,LIGHT-WHJ,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added,
,2,155,LIGHT-ILLUM,1,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added; v05.0.1 REQ->NC,
,2,156,BR-LIGHT-CHECK,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added; v05.0.1 REQ->NC,
,2,157,LIGHT-CASE,4,UNKNOWN,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added,
,3,1,UNKNOWN, , , , , , , , ,$,v4.3.0 Added,
,3,2,DESIGN, , , , , , , , ,$,v4.3.0 Added,
,3,3,CHOSEN, , , , , , , , ,$,v4.3.0 Added,
,2,158,LIGHT-TYPE,4,T12 Fluorescent - halophosphate - low frequency ballast,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R12_a01 Modified,
,3,1,NOT1, , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,3,2,NOT2, , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,3,3,REF1, , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,3,4,REF2, , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,3,5,TYP1, , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,3,6,TYP2, , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,3,7,LED, , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,3,8,Tungsten or Halogen, , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,3,9,Fluorescent - compact, , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,3,10,T12 Fluorescent - halophosphate - low frequency ballast, , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,3,11,T8 Fluorescent - halophosphate - low frequency ballast, , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,3,12,T8 Fluorescent - halophosphate - high frequency ballast, , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,3,13,T8 Fluorescent - triphosphor - high frequency ballast, , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,3,14,Metal Halide, , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,3,15,High Pressure Mercury, , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,3,16,High Pressure Sodium, , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,3,17,T5 Fluorescent - triphosphor-coated - high frequency ballast, , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,3,18,Fluorescent (no details) , , , , , , , , ,$,v4.0.a_dev_R12_a01 Modified,
,2,159,LIGHT-FACTOR-EQU,1,EXP,-6666,-6666,50,50000,-6666,-6666,W,$,2010 1.0.0_dev_R01 Added,
,2,160,LIGHT-ACT-WATT,1,-6666,-6666,-6666,50,50000,-6666,-6666,W,$,v4.3.0 Added,
,2,161,LIGHT-LUX-DESIGN,1,-6666,-6666,-6666,50,1500,-6666,-6666,lux,$,v1.2.a Added,
,2,162,LAMP-BALLAST-EFF,1,50,-6666,-6666,10,100,-6666,-6666,lumens/W,$,v4.3.0 Added,
,2,163,LIGHT-OUTPUT-RATIO,1,0.25,-6666,-6666,10,100,-6666,-6666,ratio,$,v4.3.0 Added,
,2,164,LIGHT-CONTROL,4,MANUAL,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added,
,3,1,MANUAL, , , , , , , , ,$,v4.3.0 Added,
,3,2,PHOTOELECTRIC, , , , , , , , ,$,v4.3.0 Added,
,3,3,BOTH, , , , , , , , ,$,v4.3.0 Added,
,3,4,NONE, , , , , , , , ,$,v4.3.0 Added,
,2,165,LIGHT-OCC-SENS,4,NO,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added,
,3,1,YES, , , , , , , , ,$,v4.3.0 Added,
,3,2,NO, , , , , , , , ,$,v4.3.0 Added,
,2,166,LIGHT-OCC-SENS-T,4,NONE,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added,
,3,1,MAN-ON-OFF+EXT, , , , , , , , ,$,v4.3.0 Added,
,3,2,AUTO-ON-DIMMED, , , , , , , , ,$,v4.3.0 Added,
,3,3,AUTO-ON-OFF, , , , , , , , ,$,v4.3.0 Added,
,3,4,MAN-ON-DIMMED, , , , , , , , ,$,v4.3.0 Added,
,3,5,MAN-ON-AUTO-OFF, , , , , , , , ,$,v4.3.0 Added,
,3,6,NONE, , , , , , , , ,$,v4.3.0 Added,
,2,167,LIGHT-EEXT1,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added,
,2,168,LIGHT-EEXT2,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added,
,2,169,LIGHT-PHOTO-TYPE,4,PH-SWITCHING,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added,
,3,1,PH-SWITCHING, , , , , , , , ,$,v4.3.0 Added,
,3,2,PH-DIMMING, , , , , , , , ,$,v4.3.0 Added,
,2,170,LIGHT-PH-BACK,4,NO,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added,
,3,1,YES, , , , , , , , ,$,v4.3.0 Added,
,3,2,NO, , , , , , , , ,$,v4.3.0 Added,
,2,171,LIGHT-DISPLAY-EFF,4,NO,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added,
```

```
,3,1,YES, , , , , , , , ,$,v4.3.0 Added,
,3,2,NO, , , , , , , , ,$,v4.3.0 Added,
,2,172,LIGHT-DISP-PW,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added - default modified v4.0.a_dev_R23_a02,
,2,173,LIGHT-DISP-PW-ORI,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.3 Added - default modified v4.0.a_dev_R23_a02,
,2,174,LAMP-BALLAST-EFF-DL,1,15,-6666,-6666,10,100,-6666,-6666,lumens/W,$,v4.3.0 Added - default modified v4.0.a_dev_R23_a02,
,2,175,LIGHT-PARAS-PW,1,-6666,0,1000,0.2,1.2,-6666,-6666,W/m2,$,v1.1.a Modified,
,2,176,LIGHT-PARAS-PW-OCC,1,0.3,0,1000,0.2,1.2,-6666,-6666,W/m2,$,v3.3.b_dev_R04_a01 Added,
,2,177,LIGHT-FOD,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added,
,2,178,LIGHT-TIME-SWITCH,4,NO,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R29_a03 Modified,
,3,1,YES, , , , , , , , ,$,v4.0.a_dev_R29_a03 Modified,
,3,2,NO, , , , , , , , ,$,v4.0.a_dev_R29_a03 Modified,
,2,179,LIGHT-DISP-HOFF,1,0,0,24,0,24,-6666,-6666,NC,$,v4.3.0 Added,
,2,180,LIGHT-DISP-F,1,1,0,1,0,1,-6666,-6666,NC,$,v4.3.0 Added - Fraction of display lighting that is switched off or dimmed if there is time control,
,2,181,LIGHT-GAIN-F,1,1,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added,
,2,182,LIGHT-GAIN,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added,
,2,183,LIGHT-ENERGY,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 Added,
,2,184,LIGHT-WALL/FLOOR-AREA,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0.2 Added,
,2,185,U-INF-WIND-ZONE-H,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,186,U-INF-STACK-ZONE-H,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,187,U-INF-TOT-ZONE-H,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,188,CM,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,189,TAU-HEAT,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,hours,$,v3.4.b_dev_R05_a02 Modified,
,2,190,TAU-COOL,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,hours,$,v3.4.b_dev_R05_a02 Modified,
,2,191,HEAT-MODE,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.3 Added; v05.0.6 Modified,
,2,192,COOL-MODE,5,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.3 Added; v05.0.6 Modified,
,2,193,IFA,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,194,RBA,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.0 Added,
,2,195,VENT-MECH-EXH,4,NO, , , , , , ,$,v04.3.4 Added ID1,
,3,1,YES, , , , , , , , ,$,v04.3.4 Added ID1,
,3,2,NO, , , , , , , , ,$,v04.3.4 Added ID1,
,2,196,VENT-ZONE-HVAC,4,HVAC, , , , , , ,$,v04.3.4 Added ID1,
,3,1,ZONE, , , , , , , , ,$,v04.3.4 Added ID1,
,3,2,HVAC, , , , , , , , ,$,v04.3.4 Added ID1,
,2,197,Q-AUX-ZONE,5,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.4 Added ID1,
,2,198,Q-AUX,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.2.a Added,
,2,199,Q-AUX-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.2.a Added,
,2,200,KWH-AUX,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.2.a Added,
,2,201,KWH-AUX-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.2.a Added,
,2,202,KWH-PUMP/EXH-AUX-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0.2 Added,
,2,203,SYSTEM-CORRECTION,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0.2 Added,
,2,204,VENT-SFP,1,3,-6666,-6666,0.1,3.5,-6666,-6666,W/l.s,$,v04.3.4 Added ID3,
,2,205,SFP-TU,1,0.8,-6666,-6666,0.1,3.5,-6666,-6666,W/l.s,$,2010 1.0.0.2 Added,
,2,206,VENT-SFP-EXH,1,3,-6666,-6666,0.1,3.5,-6666,-6666,W/l.s,$,v1.2.a Added,
,2,207,KG-CO2,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,208,IFCO2,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,209,RBCO2,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,210,DESTRAT-FAN,4,NO, , , , , , ,$,v05.0.1 Added ID9c,
,3,1,YES, , , , , , , , ,$,v05.0.1 Added ID9c,
,3,2,NO, , , , , , , , ,$,v05.0.1 Added ID9c,
,2,211,MULTIPLIER,1,1,1,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.4 Added,
,2,212,NOT-EFF-MECH-CORR,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,213,JNCT-ROOF-WALL-MC,1,0.6,0,-6666,0,-6666,0,-6666,NC,$,v1.1.a Added,
,2,214,JNCT-WALL-GRND-MC,1,1.15,0,-6666,0,-6666,0,-6666,NC,$,v1.1.a Added,
,2,215,JNCT-WALL-WALL-MC,1,0.25,0,-6666,0,-6666,0,-6666,NC,$,v1.1.a Added,
```

```
,2,216,JNCT-WALL-FLR-MC,1,0.07,0,-6666,0,-6666,0,-6666,NC,$,v1.1.a Added,
,2,217,JNCT-LINTEL-MC,1,1.27,0,-6666,0,-6666,0,-6666,NC,$,v1.1.a Added,
,2,218,JNCT-SILL-MC,1,1.27,0,-6666,0,-6666,0,-6666,NC,$,v1.1.a Added,
,2,219,JNCT-JAMB-MC,1,1.27,0,-6666,0,-6666,0,-6666,NC,$,v1.1.a Added,
,2,220,JNCT-OTHER-MC,1,1.27,0,-6666,0,-6666,0,-6666,NC,$,v1.1.a Added,
,2,221,JNCT-ROOF-WALL,1,0.12,0,-6666,0,-6666,0,-6666,NC,$,v1.1.a Added,
,2,222,JNCT-WALL-GRND,1,0.28,0,-6666,0,-6666,0,-6666,NC,$,v1.1.a Added,
,2,223,JNCT-WALL-WALL,1,0.09,0,-6666,0,-6666,0,-6666,NC,$,v1.1.a Added,
,2,224,JNCT-WALL-FLR,1,0.18,0,-6666,0,-6666,0,-6666,NC,$,v1.1.a Added,
,2,225,JNCT-LINTEL,1,0.53,0,-6666,0,-6666,0,-6666,NC,$,v1.1.a Added,
,2,226,JNCT-SILL,1,0.21,0,-6666,0,-6666,0,-6666,NC,$,v1.1.a Added,
,2,227,JNCT-JAMB,1,0.2,0,-6666,0,-6666,0,-6666,NC,$,v1.1.a Added,
,2,228,JNCT-OTHER,1,0.2,0,-6666,0,-6666,0,-6666,NC,$,v1.1.a Added,
,2,229,JNCT-ACCR-DETAIL,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R27
Added,
,2,230,JNCT-ACCR-DETAIL-MC,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v4.0.a_dev_R40_a02 Added,
,2,231,N-CORNERS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,232,JNCT-L-ROOF-WALL-MC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,233,JNCT-L-WALL-GRND-MC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,234,JNCT-L-WALL-WALL-MC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,235,JNCT-L-WALL-FLR-MC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,236,JNCT-L-LINTEL-MC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,237,JNCT-L-SILL-MC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,238,JNCT-L-JAMB-MC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,239,JNCT-L-OTHER-MC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,240,JNCT-L-ROOF-WALL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,241,JNCT-L-WALL-GRND,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,242,JNCT-L-WALL-WALL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,243,JNCT-L-WALL-FLR,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,244,JNCT-L-LINTEL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,245,JNCT-L-SILL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,246,JNCT-L-JAMB,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,247,JNCT-L-OTHER,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added,
,2,248,H-TR-TM,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.1.a Added. It is the transmision
heat loss coefficient due to TM, in W/K,
,2,249,VENT-MECH-EXH-NCM,4,Extract system serving a single room, , , , , , ,$,v2.0.a Added,
,3,1,Extract system serving multiple spaces, , , , , , , , ,$,v2.0.a Added,
,3,2,Extract system serving a single room, , , , , , , , ,$,v2.0.a Added,
,2,250,HEPA-FILTER,4,NO, , , , , , ,$,v2.0.a Added,
,3,1,YES, , , , , , , , ,$,v2.0.a Added,
,3,2,NO, , , , , , , , ,$,v2.0.a Added,
,2,251,Q-DEM-HEAT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,252,Q-DEM-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,253,Q-PRIM-HEAT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,254,Q-PRIM-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,255,KWH-HEAT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,256,KWH-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,257,Q-PRIM-HEAT-CHP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,258,Q-PRIM-HEAT-CHP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,259,KWH-HEAT-CHP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,260,KWH-HEAT-CHP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,261,Q-DEM-COOL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,262,Q-DEM-COOL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,263,Q-PRIM-COOL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,264,Q-PRIM-COOL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,265,KWH-COOL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,266,KWH-COOL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,267,Q-PRIM-EQUP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,268,Q-PRIM-EQUP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,269,KWH-EQUP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
```

,2,270,KWH-EQUP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,271,Q-PRIM-LI-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,272,Q-PRIM-LI,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,273,KWH-LI-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,274,KWH-LI,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,275,Q-NATGAS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,276,Q-NATGAS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,277,Q-LPG,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,278,Q-LPG-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,279,Q-BIOGAS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,280,Q-BIOGAS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,281,Q-OIL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,282,Q-OIL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,283,Q-COAL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,284,Q-COAL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,285,Q-ANTHRACITE,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,286,Q-ANTHRACITE-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,287,Q-SMOKELESS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,288,Q-SMOKELESS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,289,Q-DUELFUEL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,290,Q-DUELFUEL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,291,Q-BIOMASS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,292,Q-BIOMASS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,293,Q-SUPELEC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,294,Q-SUPELEC-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,295,Q-WASTEHEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,296,Q-WASTEHEAT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,297,Q-DISTRICT-HEATING,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,298,Q-DISTRICT-HEATING-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,299,KWH-NATGAS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,300,KWH-NATGAS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,301,KWH-LPG,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,302,KWH-LPG-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,303,KWH-BIOGAS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,304,KWH-BIOGAS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,305,KWH-OIL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,306,KWH-OIL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,307,KWH-COAL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,308,KWH-COAL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,309,KWH-ANTHRACITE,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,310,KWH-ANTHRACITE-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,311,KWH-SMOKELESS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,312,KWH-SMOKELESS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,313,KWH-DUELFUEL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,314,KWH-DUELFUEL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,315,KWH-BIOMASS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,316,KWH-BIOMASS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,317,KWH-SUPELEC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,318,KWH-SUPELEC-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,319,KWH-WASTEHEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,320,KWH-WASTEHEAT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,321,KWH-DISTRICT-HEATING,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,322,KWH-DISTRICT-HEATING-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,323,KWH-ACTUAL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R18 Added,
,2,324,KWH-ASSUMED,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R18 Added,
,2,325,KG-CO2-NATGAS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,326,KG-CO2-LPG,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,327,KG-CO2-BIOGAS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,328,KG-CO2-OIL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,329,KG-CO2-COAL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,330,KG-CO2-ANTHRACITE,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,

```
,2,331,KG-CO2-SMOKELESS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,332,KG-CO2-DUELFUEL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,333,KG-CO2-BIOMASS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,334,KG-CO2-SUPELEC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,335,KG-CO2-DISP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,336,KG-CO2-WASTEHEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,337,KG-CO2-DISTRICT-HEATING,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a Added,
,2,338,KWH-NATGAS-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,339,KWH-LPG-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,SBEM_v3.0.f_dev_R05_a01
Added,
,2,340,KWH-BIOGAS-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,341,KWH-OIL-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,SBEM_v3.0.f_dev_R05_a01
Added,
,2,342,KWH-COAL-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,343,KWH-ANTHRACITE-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,344,KWH-SMOKELESS-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,345,KWH-DUELFUEL-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,346,KWH-BIOMASS-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,347,KWH-SUPELEC-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,348,KWH-WASTEHEAT-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,349,KWH-DISTRICT-HEATING-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,350,KWH-HEATPUMP-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,351,CPS-COMP-FLAG,4,YES, , , , , , ,$,v2.0.c_R05,
,3,1,YES, , , , , , , , ,$,v2.0.c_R05,
,3,2,NO, , , , , , , , ,$,v2.0.c_R05,
,2,352,OVERHEATING-RISK,4,High risk, , , , , , ,$,Type of zone,
,3,1,No risk, , , , , , , , ,$,v3.0.a_dev_R24 - modified,
,3,2,Low risk, , , , , , , , ,$,v3.0.a_dev_R24 - modified,
,3,3,Moderate risk, , , , , , , , ,$,v3.0.a_dev_R24 - modified,
,3,4,Significant risk, , , , , , , , ,$,v3.0.a_dev_R24 - modified,
,3,5,High risk, , , , , , , , ,$,v3.0.a_dev_R24 - modified,
,2,353,LIGHT-DL-AUT-ZONE,4,YES, , , , , , ,$,v3.0.a_dev_R23 Added,
,3,1,YES, , , , , , , , ,$,v3.0.a_dev_R23 Added,
,3,2,NO, , , , , , , , ,$,v3.0.a_dev_R23 Added,
,2,354,LIGHT-DL-AREA-PERC,1,100,0,100,0,100,0,100,%,$,v3.0.a_dev_R22 Added,
,2,355,SOLAR-GAIN-CHECK,4,YES, , , , , , ,$,2010 1.0.0_dev_R01 Added,
,3,1,YES, , , , , , , , ,$,2010 1.0.0_dev_R01 Added,
,3,2,NO, , , , , , , , ,$,2010 1.0.0_dev_R01 Added,
,2,356,SOLAR-GAIN,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
,2,357,SOLAR-GAIN-OVER-LIMIT,4,EXP, , , , , , ,$,2010 1.0.0_dev_R01 Added,
,3,1,YES, , , , , , , , ,$,2010 1.0.0_dev_R01 Added,
,3,2,NO, , , , , , , , ,$,2010 1.0.0_dev_R01 Added,
,2,358,SOLAR-GAIN-LIMITS,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01
Added,
,2,359,DRIVER2A,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
,2,360,DRIVER2B,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
,2,361,DIMMING-NOT,4,NO, , , , , , ,$,v4.0.a_dev_R12_a03 Added,
,3,1,YES, , , , , , , , ,$,v4.0.a_dev_R12_a03 Added,
,3,2,NO, , , , , , , , ,$,v4.0.a_dev_R12_a03 Added,
,2,362,AE-PIG,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
```

```
,2,363,AE-PLG,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
,2,364,AE-PSG,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
,2,365,AE-PSC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
,2,366,AE-PFL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
,2,367,AE-PVL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
,2,368,AE-PSH,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
,2,369,AE-SCR,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
,2,370,AE-FAR,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
,2,371,AE-FPS1,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
,2,372,AE-FPS2,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
,2,373,AE-FPS3,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
,2,374,AE-FPS4,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
,2,375,AE-FPS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
,2,376,AE-FAN,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Added,
,2,377,NOT-CASE,1,1,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.0.0_dev_R09_a01 Added,
,2,378,TSC-SYSTEM,4,NO, , , , , , ,$,v2.0.d Added,
,3,1,YES, , , , , , , , ,$,v2.0.d Added,
,3,2,NO, , , , , , , , ,$,v2.0.d Added,
,2,379,TSC,6,{ " " },TSC, , , , , ,$,v2.0.d_dev_R22 Added,
,2,380,TSC-SUPPLY,5,EXP,0,100,0,100,0,100,%,$, ,
,2,381,Q-TSC-HEAT,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R46 Added,
,2,382,Q-TSC-HEAT-DIRECT,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R46 Added,
,2,383,Q-TSC-COOL,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R46 Added,
,2,384,TSC-KWH-AUX,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R62 Added,
,2,385,SHELL-ASSUMED,4,NO,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R18 Added,
,3,1,YES, , , , , , , , ,$,v4.0.a_dev_R18 Added,
,3,2,NO, , , , , , , , ,$,v4.0.a_dev_R18 Added,
,1,10,FAN,0,-6666,ZONE,1, , , , ,$,v4.3.0 supersided,
,2,1,FAN-TYPE,4,DIRECT,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 supersided,
,3,1,DIRECT, , , , , , , , ,$,v4.3.0 supersided,
,3,2,SINGLE-PHASE, , , , , , , , ,$,v4.3.0 supersided,
,3,3,ROTARY, , , , , , , , ,$,v4.3.0 supersided,
,2,2,FLOW-CTRL-TYPE,4,NO, , , , , , ,$,7.3.3.2.a Is the reduction factor for control of fan. T019. v4.3.0
supersided,
,3,1,NO, , , , , , , , ,$,v4.3.0 supersided,
,3,2,BLADE-ADJ, , , , , , , , ,$,v4.3.0 supersided,
,3,3,SPEED, , , , , , , , ,$,v4.3.0 supersided,
,2,3,P-SHAFT,1,REQ,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,7.3.3.3. v4.3.0 supersided,
,2,4,FAN-U-ELE,1,220,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 supersided,
,2,5,FAN-I,1,1,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 supersided,
,2,6,POWER-FACTOR,1,0.8,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 supersided,
,2,7,P-EFF-CONNECT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 supersided,
,2,8,MULTIPLIER,1,1,1,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.4 Added,
,1,11,COLD-GENERATOR,0,-6666, ,1, , , , ,$,v4.3.0 supersided,
,2,1,TYPE,4,CHILLER-COMP, , , , , , ,$,v4.3.0 supersided,
,3,1,CHILLER-COMP, , , , , , , , ,$,v4.3.0 supersided,
,3,2,CHILLER-ABS-DH, , , , , , , , ,$,v4.3.0 supersided,
,3,3,CHILLER-ABS-COG, , , , , , , , ,$,v4.3.0 supersided,
,3,4,COLD-STORAGE, , , , , , , , ,$,v4.3.0 supersided,
,3,5,HEAT-PUMP, , , , , , , , ,$,v4.3.0 supersided,
,2,2,DH-TYPE,4,GASMOTOR, , , , , , ,$,v4.3.0 supersided,
,3,1,GASMOTOR, , , , , , , , ,$,v4.3.0 supersided,
,3,2,HEAT-PUMP-UP, , , , , , , , ,$,v4.3.0 supersided,
,3,3,CCPP, , , , , , , , ,$,v4.3.0 supersided,
,3,4,IND, , , , , , , , ,$,v4.3.0 supersided,
,3,5,WASTE-COMB, , , , , , , , ,$,v4.3.0 supersided,
,2,3,GEN-CAP,1,REQ,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 supersided,
,2,4,GEN-CAP-NORM,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 supersided,
,2,5,GEN-COP,1,NC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 supersided,
,2,6,GEN-EFF,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 supersided,
,2,7,AREA-COOLED,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 supersided,
```

```
,2,8,CS-WATER-FLOW,1,0,0,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.0 supersided,
,2,9,MULTIPLIER,1,1,1,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.4 Added,
,1,12,DHW-GENERATOR,0,-6666, ,0, , , , , ,$, ,
,2,1,ISBEM-ID,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.3.a Added v3.0.a_R21 Mod,
,2,2,Q-PRIM-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,3,Q-PRIM,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Modified,
,2,4,KWH-DHW-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,5,KWH-DHW,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,6,KG-CO2-DHW,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R01_CD Added ,
,2,7,Q-DEM,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,8,Q-DEM-TOT,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.1.0 Added,
,2,9,Q-DEM-TOT-CORR,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.b_dev_R65_a01 Added,
,2,10,Q-SES-DHW,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,11,Q-CHP-DHW,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.4.b_dev_R13_a11 Added,
,2,12,R-GEN-DHW,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,13,SOURCE-TYPE,4,Electric, , , , , , ,$,v04.3.3 Superseded,
,3,1,Electric, , , , , , , , ,$,v04.3.3 Superseded,
,3,2,VR Boiler, , , , , , , , ,$,v04.3.3 Superseded,
,3,3,Gas Boiler, , , , , , , , ,$,v04.3.3 Superseded,
,3,4,Cogeneration, , , , , , , , ,$,v04.3.3 Superseded,
,3,5,District heating, , , , , , , , ,$,v04.3.3 Superseded,
,3,6,Heat pump, , , , , , , , ,$,v04.3.3 Superseded,
,3,7,Steam, , , , , , , , ,$,v04.3.3 Superseded,
,2,14,DH-TYPE,4,GASMOTOR, , , , , , ,$,v04.3.3 Superseded,
,3,1,GASMOTOR, , , , , , , , ,$,v04.3.3 Superseded,
,3,2,HEAT-PUMP-UP, , , , , , , , ,$,v04.3.3 Superseded,
,3,3,CCPP, , , , , , , , ,$,v04.3.3 Superseded,
,3,4,IND, , , , , , , , ,$,v04.3.3 Superseded,
,3,5,WASTE-COMB, , , , , , , , ,$,v04.3.3 Superseded,
,2,15,AREA-SERVED,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.3 Added,
,2,16,DIST-LOSS,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.3 Added,
,2,17,STORE-VOL,1,-6666,-6666,-6666,0,10000,-6666,-6666,litres,$,v4.3.3 Added,
,2,18,STORE-VOL-DEFAULT,1,0,-6666,-6666,0,10000,-6666,-6666,litres,$,v2.0.d_dev_R22 Added,
,2,19,STORE-VOL-SES,1,0,-6666,-6666,0,10000,-6666,-6666,litres,$,v3.4.b_dev_R19_a05 Added,
,2,20,STORE-LOSS,1,-6666,-6666,-6666,0,1000,-6666,-6666,MJ/month,$,v4.3.3 Added,
,2,21,STORE-LOSS-DEFAULT,1,0,-6666,-6666,0,1000,-6666,-6666,MJ/month,$,v2.0.d_dev_R22 Added,
,2,22,STORE-LOSS-DAY,1,-6666,-6666,-6666,0,1000,-6666,-6666,kWh/litre.day,$,v4.0.a_dev_R01_a01
Added,
,2,23,STORE-INSULAT-TYPE,4,Uninsulated, , , , , , ,$,v3.4.b_dev_R13_a10 Added,
,3,1,Uninsulated, , , , , , , , ,$,v3.4.b_dev_R13_a10 Added,
,3,2,Loose jacket, , , , , , , , ,$,v3.4.b_dev_R13_a10 Added,
,3,3,Factory insulated, , , , , , , , ,$,v3.4.b_dev_R13_a10 Added,
,2,24,STORE-INSULAT-THICK,1,0,0,500,0,200,-6666,-6666,mm,$,v3.4.b_dev_R13_a10 Added,
,2,25,SEC-LOOP-LENGTH,1,-6666,-6666,-6666,0,1000,-6666,-6666,m,$,v4.3.3 Added,
,2,26,SEC-CIRC-LOSS/M,1,-6666,-6666,-6666,0,1000,-6666,-6666,W/m,$,v4.3.3 Added,
,2,27,SEC-CIRC-LOSS,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.3 Added,
,2,28,SEC-CIRC-TS,4,NO,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.c_R07,
,3,1,YES, , , , , , , , ,$,v2.0.c_R07,
,3,2,NO, , , , , , , , ,$,v2.0.c_R07,
,2,29,HOURS,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.3 Added,
,2,30,DIST-EFF,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.3 Added,
,2,31,SEC-PUMP-POWER,1,-6666,-6666,-6666,0,10000,-6666,-6666,W,$,v4.3.3 Added,
,2,32,SEC-PUMP-ENERGY,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.3 Added,
,2,33,THERM-EFF,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.3 Added,
,2,34,HEAT-GEN-EFF,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.3.3 Added,
,2,35,HEAT-GEN-EFF-ACT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.2.a Added,
,2,36,HEAT-GEN-TYPE,4,Dedicated hot water boiler, , , , , , ,$,v4.3.3 Added,
,3,1,Using Central Heating boiler, , , , , , , , ,$,v4.3.3 Added,
,3,2,Dedicated hot water boiler, , , , , , , , ,$,v4.3.3 Added,
,3,3,Stand-alone water heater, , , , , , , , ,$,v4.3.3 Added,
,3,4,Instantaneous hot water only, , , , , , , , ,$,v4.3.3 Added,
```

```
,3,5,Instantaneous combi, , , , , , , , ,$,v4.3.3 Added,
,3,6,Direct (or off-peak) electricity, , , , , , , , ,$,v4.3.3 Added,
,3,7,Heat pump, , , , , , , , ,$,v4.3.3 Added,
,3,8,NOT, , , , , , , , ,$,v4.3.4 Added,
,2,37,HEAT-SOURCE,4,LTHW boiler,1,16, , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,1,LTHW boiler, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,2,MTHW boiler, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,3,HTHW boiler, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,4,Steam boiler (hidden by now), , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,5,delete now, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,6,Air heater, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,7,Unflued gas warm air heater, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,8,Unitary radiant heater, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,9,Radiant heater, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,10,Unflued radiant heater, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,11,Direct or storage electric heater, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,12,Heat pump (gas/oil): air source, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,13,Heat pump (gas/oil): ground or water source, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,14,Room heater, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,15,District heating, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,16,CHP, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,17,Direct gas firing, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,18,Heat pump (electric): air source, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,3,19,Heat pump (electric): ground or water source, , , , , , , , ,$,v3.2.c_dev_R04_a02 Added,
,2,38,HVAC-SYSTEM-EFF,1,0.6175,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.6 Added,
,2,39,DHW-GEN-SEFF,1,0.5,0,-6666,0.4,1,-6666,-6666,NC,$,v1.2.a Added,
,2,40,HEAT-GEN-POST98,4,NO, , , , , , ,$,v4.3.3 Added,
,3,1,YES, , , , , , , , ,$,v4.3.3 Added,
,3,2,NO, , , , , , , , ,$,v4.3.3 Added,
,2,41,STORE-SYSTEM,4,NO, , , , , , ,$,v4.3.4 Added,
,3,1,YES, , , , , , , , ,$,v4.3.4 Added,
,3,2,NO, , , , , , , , ,$,v4.3.4 Added,
,2,42,SEC-CIRCULATION,4,NO, , , , , , ,$,v4.3.4 Added,
,3,1,YES, , , , , , , , ,$,v4.3.4 Added,
,3,2,NO, , , , , , , , ,$,v4.3.4 Added,
,2,43,FUEL-TYPE,4,Natural gas,1,12, , , , ,$,v4.3.4 Added ID2,
,3,1,Natural Gas, , , , , , , , ,$,v4.3.4 Added ID2,
,3,2,LPG, , , , , , , , ,$,v4.3.4 Added ID2,
,3,3,Biogas, , , , , , , , ,$,v4.3.4 Added ID2,
,3,4,Oil, , , , , , , , ,$,v4.3.4 Added ID2,
,3,5,Coal, , , , , , , , ,$,v4.3.4 Added ID2,
,3,6,Anthracite, , , , , , , , ,$,v5.0.1 Added ID7,
,3,7,Smokeless Fuel (inc Coke), , , , , , , , ,$,v1.1 BUG from 1.0 corrected,
,3,8,Dual Fuel Appliances (Mineral + Wood), , , , , , , , ,$,v1.1 BUG from 1.0 corrected,
,3,9,Biomass, , , , , , , , ,$,v4.3.4 Added ID2,
,3,10,Grid Supplied Electricity, , , , , , , , ,$,v4.3.4 Added ID2,
,3,11,Grid Displaced Electricity, , , , , , , , ,$,v4.3.4 Added ID2,
,3,12,Waste Heat, , , , , , , , ,$,v4.3.4 Added ID2,
,3,13,District Heating, , , , , , , , ,$,v3.0.b_dev_R01 Added,
,2,44,MULTIPLIER,1,1,1,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.4 Added,
,2,45,CHP,4,NO, , , , , , ,$,v05.0.8 Added,
,3,1,YES, , , , , , , , ,$,v05.0.8 Added,
,3,2,NO, , , , , , , , ,$,v05.0.8 Added,
,2,46,ACTIVATED,4,NO, , , , , , ,$,v1.1.b Added (Not include in manuals),
,3,1,YES, , , , , , , , ,$,v1.1.b Added (Not include in manuals),
,3,2,NO, , , , , , , , ,$,v1.1.b Added (Not include in manuals),
,2,47,HVACGUIDE-DHW,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.2.a Added,
,2,48,REC-PERFORMANCE,4,Not applicable, , , , , , ,$,v2.0.d_dev_R22 Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22 Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22 Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22 Added,
```

,3,4,Not applicable, , , , , , , , ,$,v2.0.d_dev_R22 Added,
,2,49,EPC-W1,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added,
,2,50,EPC-W3,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added,
,2,51,EPC-W4,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added,
,2,52,EPC-W2,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R39_CD Added _R47 Modified,
,2,53,SC-ACTUAL-RATIO,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R18 Added,
,2,54,BH-HEAT-SOURCE,7,{ "-5555" },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R29 Added,
,2,55,BH-FUEL-TYPE,7,{ "-5555" },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R29 Added,
,2,56,BH-HEAT-GEN-TYPE,7,{ "-5555" },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R29_a03 Added,
,2,57,BH-DHW-GEN-SEFF,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R29_a03 Added,
,2,58,BH-HVAC-SYSTEM-EFF,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R29 Added,
,2,59,BH-LOAD,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R29 Added,
,1,13,SES,0,-6666,DHW-GENERATOR,1, , , , , ,$, ,
,2,1,Q-SES-HW-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.4.b_dev_R19_a02 renamed from Q-SES,
,2,2,Q-RAD-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,3,Q-RAD,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,4,AREA,1,REQ,0.00001,-6666,0.1,500,-6666,-6666,m2,$, ,
,2,5,SES-ORIENTATION,4,South, , , , , , ,$, ,
,3,1,North, , , , , , , , ,$, ,
,3,2,North-East, , , , , , , , ,$, ,
,3,3,East, , , , , , , , ,$, ,
,3,4,South-East, , , , , , , , ,$, ,
,3,5,South, , , , , , , , ,$, ,
,3,6,South-West, , , , , , , , ,$, ,
,3,7,West, , , , , , , , ,$, ,
,3,8,North-West, , , , , , , , ,$, ,
,2,6,SES-INCLINATION,4,60, , , , , , ,$, ,
,3,1,0, , , , , , , , ,$, ,
,3,2,15, , , , , , , , ,$, ,
,3,3,30, , , , , , , , ,$, ,
,3,4,45, , , , , , , , ,$, ,
,3,5,60, , , , , , , , ,$, ,
,3,6,75, , , , , , , , ,$, ,
,3,7,90, , , , , , , , ,$, ,
,2,7,MULTIPLIER,1,1,1,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.4 Added,
,2,8,COLLECTOR-PAR-KNOWN,4,NO, , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,1,YES, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,2,NO, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,2,9,COLLECTOR-TYPE,4,Flat Panel, , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,1,Unglazed, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,2,Flat Panel, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,3,Evacuated tube, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,2,10,SIGMA-ZERO,1,0.6,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.4.b_dev_R19_a02 Added,
,2,11,A1,1,20,-6666,-6666,-6666,-6666,-6666,-6666,W/m2K,$,v3.4.b_dev_R19_a02 Added,
,2,12,A2,1,0,-6666,-6666,-6666,-6666,-6666,-6666,W/m2K,$,v3.4.b_dev_R19_a02 Added,
,2,13,IAM,1,1,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.4.b_dev_R19_a02 Added,
,2,14,SES-STORE-VOL,1,REQ,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.4.b_dev_R19_a05 Added,
,2,15,SES-PRE-HEAT-TYPE,4,Combined cylinder, , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,1,Separate solar cylinder, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,2,Combined cylinder, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,2,16,STORE-LOSS,1,EXP,-6666,-6666,0,1000,-6666,-6666,MJ/month,$,v3.4.b_dev_R19_a05 Added,
,2,17,STORE-INSULAT-TYPE,4,Uninsulated, , , , , , ,$,v3.4.b_dev_R19_a05 Added,
,3,1,Uninsulated, , , , , , , , ,$,v3.4.b_dev_R19_a05 Added,
,3,2,Loose jacket, , , , , , , , ,$,v3.4.b_dev_R19_a05 Added,
,3,3,Factory insulated, , , , , , , , ,$,v3.4.b_dev_R19_a05 Added,

```
,2,18,STORE-INSULAT-THICK,1,0,0,500,0,200,-6666,-6666,mm,$,v3.4.b_dev_R19_a05 Added,
,2,19,HEAT-EXCHANGE,4,YES, , , , , , ,$,v3.4.b_dev_R21_a04 Added,
,3,1,YES, , , , , , , , ,$,v3.4.b_dev_R21_a04 Added,
,3,2,NO, , , , , , , , ,$,v3.4.b_dev_R21_a04 Added,
,2,20,HEAT-EXCH-LOSS-KNOWN,4,NO, , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,1,YES, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,2,NO, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,2,21,HEAT-EXCH-LOSS,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.4.b_dev_R19_a02 Added,
,2,22,U-LOOP-P-KNOWN,4,NO, , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,1,YES, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,2,NO, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,2,23,U-LOOP-P,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.4.b_dev_R19_a02 Added,
,2,24,SES-CIRC-SYS,4,thermosiphon system, , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,1,thermosiphon system, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,2,forced circulation system with PV, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,3,forced circulation system with no PV, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,2,25,AUX-POWER-KNOWN,4,NO, , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,1,YES, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,2,NO, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,2,26,AUX-POWER,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.4.b_dev_R19_a02 Added,
,2,27,PIPE-INSULAT,4,NO, , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,1,YES, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,3,2,NO, , , , , , , , ,$,v3.4.b_dev_R19_a02 Added,
,2,28,KWH-SES-AUX-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.4.b_dev_R19_a02 Added,
,2,29,AREA-SERVED,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.4.b_dev_R19_a02 Added,
,1,14,PVS,0,-6666, ,1, , , , ,$, ,
,2,1,Q-SES,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,2,Q-RAD-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,3,Q-RAD,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,4,Q-DISP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,5,KWH-DISP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,6,KG-CO2-DISP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,7,Q-DISP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,8,KWH-DISP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,9,KG-CO2-DISP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$, ,
,2,10,AREA,1,REQ,0.00001,-6666,0.1,500,-6666,-6666,m2,$, ,
,2,11,WATT-PEAK-TYPE,4,Other thin films, , , , , , ,$, ,
,3,1,Monocrystalline silicon, , , , , , , , ,$, ,
,3,2,Policrystalline silicon, , , , , , , , ,$, ,
,3,3,Amorphous silicon, , , , , , , , ,$, ,
,3,4,Other thin films, , , , , , , , ,$, ,
,2,12,SES-ORIENTATION,4,South, , , , , , ,$, ,
,3,1,North, , , , , , , , ,$, ,
,3,2,North-East, , , , , , , , ,$, ,
,3,3,East, , , , , , , , ,$, ,
,3,4,South-East, , , , , , , , ,$, ,
,3,5,South, , , , , , , , ,$, ,
,3,6,South-West, , , , , , , , ,$, ,
,3,7,West, , , , , , , , ,$, ,
,3,8,North-West, , , , , , , , ,$, ,
,2,13,SES-INCLINATION,4,60, , , , , , ,$, ,
,3,1,0, , , , , , , , ,$, ,
,3,2,15, , , , , , , , ,$, ,
,3,3,30, , , , , , , , ,$, ,
,3,4,45, , , , , , , , ,$, ,
,3,5,60, , , , , , , , ,$, ,
,3,6,75, , , , , , , , ,$, ,
,3,7,90, , , , , , , , ,$, ,
,2,14,MULTIPLIER,1,1,1,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.4 Added,
,1,15,GENERAL,1,1, ,0, , , , ,$,v4.3.0 Added,
,2,1,P-NAME,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.2
```

Added,
,2,2,UPRN,2,"000000000000",12,12,12,12,12,12,NC,$,v3.0.a_dev_R16 Added,
,2,3,MPRN,2,"000000000000",12,12,12,12,12,12,NC,$,v3.4.b_dev_R13_a0 Added,
,2,4,B-INSP-DATE,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R17 Added _R35 Mod,
,2,5,B-CONS-DATE,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.4.b_dev_R13_a09 Added,
,2,6,B-TYPE,4,B1 Offices and Workshop businesses,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,2010 1.0.0_dev_R01 Modified,
,3,1,A1/A2 Retail and Financial/Professional services, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,2,A3/A4/A5 Restaurant and Cafes/Drinking Establishments and Hot Food takeaways, , , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,3,B1 Offices and Workshop businesses, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,4,B2 to B7 General Industrial and Special Industrial Groups, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,5,B8 Storage or Distribution, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,6,C1 Hotels, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,7,C2 Residential Institutions - Hospitals and Care Homes, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,8,C2 Residential Institutions - Residential schools, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,9,C2 Residential Institutions - Universities and colleges, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,10,C2A Secure Residential Institutions, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,11,Residential spaces, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,12,D1 Non-residential Institutions - Community/Day Centre, , , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,13,D1 Non-residential Institutions - Libraries Museums and Galleries, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,14,D1 Non-residential Institutions - Education, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,15,D1 Non-residential Institutions - Primary Health Care Building, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,16,D1 Non-residential Institutions - Crown and County Courts, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,17,D2 General Assembly and Leisure plus Night Clubs and Theatres, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,18,Others - Passenger terminals, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,19,Others - Emergency services, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,20,Others -Telephone exchanges, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,21,Others - Miscellaneous 24hr activities, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,22,Others - Car Parks 24 hrs, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,3,23,Others - Stand alone utility block, , , , , , , , ,$,2010 1.0.0_dev_R01 Modified,
,2,7,B-ADDRESS-0,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.e_dev_R02 Added,
,2,8,B-ADDRESS-1,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.e_dev_R02 Added,
,2,9,B-ADDRESS-2,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.e_dev_R02 Added,
,2,10,B-ADDRESS-3,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.e_dev_R02 Added,
,2,11,B-ADDRESS,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.2 Added v3.0.e_dev_R02 Mod,
,2,12,B-CITY,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.2 Added v3.0.e_dev_R02 Mod,
,2,13,B-POSTCODE,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.2 Added v3.0.e_dev_R02 Mod,
,2,14,B-COUNTY,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.1.b_dev_R08_a02 Added ,
,2,15,B-LOC-DESCRIPTION,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R61 Added,
,2,16,BUILDING-AREA,1,-6666,-6666,-6666,10,100000,-6666,-6666,m2,$,v1.1.a Added,
,2,17,BUILDING-VOLUME,1,-6666,-6666,-6666,10,100000,-6666,-6666,m3,$,v3.1.b_dev_R04_a03 Added,
,2,18,BUILDING-AREA-CALC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.3.a Added,
,2,19,O-NAME,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.2 Added,
,2,20,O-TELEPHONE,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.2 Added,
,2,21,O-ADDRESS,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.2 Added,
,2,22,O-CITY,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.2 Added,
,2,23,O-POSTCODE,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.2 Added,

,2,24,O-COUNTY,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.1.b_dev_R08_a02 Added ,
,2,25,C-NAME,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.2 Added,
,2,26,C-TELEPHONE,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.2 Added,
,2,27,C-ADDRESS,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.2 Added,
,2,28,C-CITY,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.2 Added,
,2,29,C-POSTCODE,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.2 Added,
,2,30,C-COUNTY,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.1.b_dev_R08_a02 Added ,
,2,31,C-ACCRED-SCHEME,2,"Information not provided by the user", , , , , , ,$,v2.0..d_dev_R08 Added _3.0.a_R17 Mod,
,2,32,C-REG-NUMBER,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0..d_dev_R08 Added,
,2,33,C-REG-NUMBER-S,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.2.a_dev_R07 Added,
,2,34,C-REG-NUMBER-IR,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.2.a_dev_R07 Added,
,2,35,C-COMPANY,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0..d_dev_R08 Added,
,2,36,C-EMP-TRAD-NAME,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a R17 Added,
,2,37,C-EMP-TRAD-ADDRESS,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a R17 Added,
,2,38,C-EMP-TRAD-NUMBER,2,"Information not provided by the user",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.2.a_dev_R07 Added,
,2,39,C-REL-PART-DISC,4,Not recorded,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.4.b_dev_R11_a15 Modified,
,3,1,Not related to the owner, , , , , , , , ,$,v3.4.b_dev_R11_a15 Modified,
,3,2,Relative of the owner, , , , , , , , ,$,v3.4.b_dev_R11_a15 Modified,
,3,3,Relative of the professional dealing with the property transaction, , , , , , , , ,$,v3.4.b_dev_R11_a15 Modified,
,3,4,Indirect relation to the owner (e.g. relative employed/contracted by owner), , , , , , , , , ,$,v3.4.b_dev_R11_a15 Modified,
,3,5,Occupier of the property, , , , , , , , ,$,v3.4.b_dev_R11_a15 Modified,
,3,6,Owner or Director of the organisation dealing with the property transaction, , , , , , , , , ,$,v3.4.b_dev_R11_a15 Modified,
,3,7,Employed by the organisation dealing with the property transaction, , , , , , , , ,$,v3.4.b_dev_R11_a15 Modified,
,3,8,Financial interest in the property, , , , , , , , ,$,v3.4.b_dev_R11_a15 Modified,
,3,9,Employed by the owner, , , , , , , , ,$,v3.4.b_dev_R11_a15 Modified,
,3,10,Contracted by the owner to provide other Energy Assessment services, , , , , , , , , ,$,v3.4.b_dev_R11_a15 Modified,
,3,11,Contracted by the owner to provide other (non Energy Assessment) services, , , , , , , , , ,$,v3.4.b_dev_R11_a15 Modified,
,3,12,Previous relation to the owner (e.g. employee/contractor), , , , , , , , , ,$,v3.4.b_dev_R11_a15 Modified,
,3,13,Not recorded, , , , , , , , ,$,v3.4.b_dev_R14_a01 Added,
,2,40,C-INSURER,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a R39 Added,
,2,41,C-INS-POL-NUMBER,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a R39 Added,
,2,42,C-INS-EFF-DATE,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a R39 Added,
,2,43,C-INS-EXP-DATE,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a R39 Added,
,2,44,C-INS-PI-LIMIT,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a R39 Added,
,2,45,NOS-LEVEL,4,Level 3,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R29 Added,
,3,1,Level 3, , , , , , , , ,$,v3.0.a_dev_R29 Added,
,3,2,Level 4, , , , , , , , ,$,v3.0.a_dev_R29 Added,
,3,3,Level 5, , , , , , , , ,$,v3.0.a_dev_R29 Added,
,2,46,C-QUALIFICATIONS,4,NOS3,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R61 Added,
,3,1,NOS3, , , , , , , , ,$,v3.0.a_dev_R61 Added,

```
,3,2,NOS4, , , , , , , , ,$,v3.0.a_dev_R61 Added,
,3,3,NOS5, , , , , , , , ,$,v3.0.a_dev_R61 Added,
,2,47,TRANSACTION-TYPE,4,Not recorded, , , , , , ,$,v3.4.b_dev_R11_a15 Added,
,3,1,Mandatory issue (Marketed sale), , , , , , , , ,$,v3.4.b_dev_R11_a15 Added,
,3,2,Mandatory issue (Non-marketed sale), , , , , , , , ,$,v3.4.b_dev_R11_a15 Added,
,3,3,Mandatory issue (Property on construction), , , , , , , , ,$,v3.4.b_dev_R11_a15 Added,
,3,4,Mandatory issue (Property to let), , , , , , , , ,$,v3.4.b_dev_R11_a15 Added,
,3,5,Voluntary re-issue (A valid EPC is already lodged), , , , , , , , ,$,v3.4.b_dev_R11_a15 Added,
,3,6,Voluntary (No legal requirement for an EPC), , , , , , , , ,$,v3.4.b_dev_R11_a15 Added,
,3,7,Not recorded, , , , , , , , ,$,v3.4.b_dev_R14_a01 Added,
,2,48,INTERFACE-VAL,4,Not Approved,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R15
Added,
,3,1,Not Approved, , , , , , , , ,$,v2.0.d_dev_R15 Added,
,3,2,iSBEM, , , , , , , , ,$,v2.0.d_dev_R15 Added,
,3,3,CARBON-CHECKER, , , , , , , , ,$,v2.0.d_dev_R15 Added,
,3,4,HEVACOMP, , , , , , , , ,$,v2.0.d_dev_R15 Added,
,3,5,IES, , , , , , , , ,$,v2.0.d_dev_R50 Added,
,3,6,DESIGN-BUILDER, , , , , , , , ,$,v2.0.d_dev_R15 Added,
,3,7,DIGITALENERGY, , , , , , , , ,$,v3.0.a_dev_R60 Added,
,3,8,ECAT, , , , , , , , ,$,v3.0.a_dev_R60 Added,
,2,49,INTERFACE-VAL-OK,4,NO,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0..d_dev_R15 Added,
,3,1,YES, , , , , , , , ,$,v2.0..d_dev_R15 Added,
,3,2,NO, , , , , , , , ,$,v2.0..d_dev_R15 Added,
,2,50,INTERFACE,2,REQ,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v06.0.3 Added,
,2,51,INTERFACE-VERSION,2,REQ,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v06.0.3 Added,
,2,52,SOFT-COMP-NAME,2,REQ,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.4.b_dev_R15 Added,
,2,53,PATH-FILE-INTERFACE,2,"-5555",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a R56 Added,
,2,54,PATH-FILE-ENGINE,2,"-5555",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a R56 Added,
,2,55,ACT-NOT,4,REQ, , , , , , ,$,v4.3.0 Added,
,3,1,ACT, , , , , , , , ,$,v4.3.0 Added,
,3,2,NOT, , , , , , , , ,$,v4.3.0 Added,
,2,56,NOT-TYPE,4,NOT, , , , , , ,$,v2.0.d_dev_R05 Added,
,3,1,NOT, , , , , , , , ,$,v2.0.d_dev_R05 Added,
,3,2,REF, , , , , , , , ,$,v2.0.d_dev_R05 Added,
,3,3,TYP, , , , , , , , ,$,v2.0.d_dev_R05 Added,
,2,57,NOT-CASE,1,1,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.3.c_dev_R23 Added,
,2,58,WEATHER,4,REQ, , , , , , ,$,v4.3.0 Added,
,3,1,BEL, , , , , , , , ,$,v4.3.0 Added,
,3,2,BIR, , , , , , , , ,$,v4.3.0 Added,
,3,3,CAR, , , , , , , , ,$,v4.3.0 Added,
,3,4,DUB, , , , , , , , ,$,v1.1.a Added,
,3,5,EDI, , , , , , , , ,$,v1.1.a Added,
,3,6,GLA, , , , , , , , ,$,v1.1.a Added,
,3,7,LEE, , , , , , , , ,$,v1.1.a Added,
,3,8,LON, , , , , , , , ,$,v1.1.a Added,
,3,9,MAN, , , , , , , , ,$,v1.1.a Added,
,3,10,NEW, , , , , , , , ,$,v1.1.a Added,
,3,11,NOR, , , , , , , , ,$,v1.1.a Added,
,3,12,NOT, , , , , , , , ,$,v1.1.a Added,
,3,13,PLY, , , , , , , , ,$,v1.1.a Added,
,3,14,SOU, , , , , , , , ,$,v1.1.a Added,
,3,15,SWI, , , , , , , , ,$,v1.1.a Added,
,3,16,JER, , , , , , , , ,$,v4.3.0 Added,
,2,59,ANALYSIS-TYPE,4,BUILT, , , , , , ,$,v5.0.3 Added,
,3,1,DESIGN, , , , , , , , ,$,v5.0.3 Added,
,3,2,BUILT, , , , , , , , ,$,v5.0.3 Added,
,3,3,OTHER, , , , , , , , ,$,v5.0.3 Added,
,2,60,AREA,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,61,H-STACK,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.a modified,
,2,62,CHP-POS,1,-1,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.08 Added,
,2,63,Q-NATGAS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
```

```
,2,64,Q-NATGAS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,65,Q-LPG,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,66,Q-LPG-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,67,Q-BIOGAS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,68,Q-BIOGAS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,69,Q-OIL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,70,Q-OIL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,71,Q-COAL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,72,Q-COAL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,73,Q-ANTHRACITE,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,74,Q-ANTHRACITE-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,75,Q-SMOKELESS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,76,Q-SMOKELESS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,77,Q-DUELFUEL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,78,Q-DUELFUEL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,79,Q-BIOMASS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,80,Q-BIOMASS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,81,Q-SUPELEC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,82,Q-SUPELEC-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,83,Q-DISP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,84,Q-DISP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,85,Q-WASTEHEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,86,Q-WASTEHEAT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,87,Q-DISTRICT-HEATING,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,88,Q-DISTRICT-HEATING-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,89,Q-NATGAS-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,SBEM_v3.0.f_dev_R05_a01
Added,
,2,90,Q-LPG-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,SBEM_v3.0.f_dev_R05_a01
Added,
,2,91,Q-BIOGAS-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,SBEM_v3.0.f_dev_R05_a01
Added,
,2,92,Q-OIL-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,SBEM_v3.0.f_dev_R05_a01
Added,
,2,93,Q-COAL-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,SBEM_v3.0.f_dev_R05_a01
Added,
,2,94,Q-ANTHRACITE-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,95,Q-SMOKELESS-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,96,Q-DUELFUEL-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,97,Q-BIOMASS-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,SBEM_v3.0.f_dev_R05_a01
Added,
,2,98,Q-SUPELEC-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,SBEM_v3.0.f_dev_R05_a01
Added,
,2,99,Q-WASTEHEAT-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,100,Q-DISTRICT-HEATING-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,101,Q-HEATPUMP-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,SBEM_v3.0.f_dev_R05_a01 Added,
,2,102,KWH-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,103,KG-CO2-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R01_CD Added ,
,2,104,KWH-HEAT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,105,KWH-COOL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,106,KG-CO2-COOL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R01_CD Added ,
,2,107,KWH-COOL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,108,KWH-AUX,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,109,KG-CO2-AUX,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R01_CD Added ,
,2,110,KWH-AUX-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,111,KWH-EQUP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
```

```
,2,112,KWH-EQUP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,113,KWH-LI,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,114,KG-CO2-LI,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R01_CD Added ,
,2,115,KWH-LI-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,116,KWH-DHW,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,117,KG-CO2-DHW,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R01_CD Added ,
,2,118,KWH-DHW-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,119,Q-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,120,Q-HEAT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,121,Q-COOL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,122,Q-COOL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,123,Q-AUX,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,124,Q-AUX-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,125,Q-EQUP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,126,Q-EQUP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,127,Q-LI,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,128,Q-LI-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,129,Q-DHW,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,130,Q-DHW-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,131,KWH-TOTAL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added,
,2,132,KWH-JER,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R63_public_a03 Added,
,2,133,Q-TOTAL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added,
,2,134,Q-TOTAL-P-WEIGHTED,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R09_CD
Added,
,2,135,KWH-NATGAS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,136,KWH-NATGAS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,137,KWH-LPG,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,138,KWH-LPG-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,139,KWH-BIOGAS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,140,KWH-BIOGAS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,141,KWH-OIL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,142,KWH-OIL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,143,KWH-COAL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,144,KWH-COAL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,145,KWH-ANTHRACITE,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,146,KWH-ANTHRACITE-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,147,KWH-SMOKELESS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,148,KWH-SMOKELESS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,149,KWH-DUELFUEL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,150,KWH-DUELFUEL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,151,KWH-BIOMASS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,152,KWH-BIOMASS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,153,KWH-SUPELEC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,154,KWH-SUPELEC-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,155,KWH-DISP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,156,KWH-DISP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,157,KWH-WASTEHEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,158,KWH-WASTEHEAT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,159,KWH-DISTRICT-HEATING,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,160,KWH-DISTRICT-HEATING-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,161,KWH-PVS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R46 Added,
,2,162,KWH-WIND,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R46 Added,
,2,163,KWH-CHP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R46 Added,
,2,164,KWH-SES,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R46 Added,
,2,165,KWH-HEAT-PUMP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R31 Added,
,2,166,KG-CO2-NATGAS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,167,KG-CO2-LPG,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,168,KG-CO2-BIOGAS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,169,KG-CO2-OIL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,170,KG-CO2-COAL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,171,KG-CO2-ANTHRACITE,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
```

```
,2,172,KG-CO2-SMOKELESS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,173,KG-CO2-DUELFUEL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.1 Added ID7,
,2,174,KG-CO2-BIOMASS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,175,KG-CO2-SUPELEC,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,176,KG-CO2-DISP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,177,KG-CO2-WASTEHEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,178,KG-CO2-DISTRICT-HEATING,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3
Added,
,2,179,KG-CO2,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,180,KG-CO2-EXCL-RENEWABLES,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,2,181,KG-CO2-DISP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,182,KG-CO2-CHP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.9 Added,
,2,183,KG-CO2-CHP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.9 Added,
,2,184,KWH-PRIM-IR,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.1.b_dev_R03_a02 Added,
,2,185,KWH-PRIM-JER,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.3.c_dev_R14 Added,
,2,186,ELEC-POWER-FACTOR,4,<0.9, , , , , , ,$,v1.0.b Added,
,3,1,<0.9, , , , , , , , ,$, ,
,3,2,0.9 to 0.95, , , , , , , , ,$, ,
,3,3,>0.95, , , , , , , , ,$, ,
,2,187,LIGHT-METERING,1,1,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.0.b Added,
,2,188,NOT-FUEL,4,OIL, , , , , , ,$,v1.1.a Added,
,3,1,OIL, , , , , , , , ,$,v1.1.a Added,
,3,2,GAS, , , , , , , , ,$,v1.1.a Added,
,2,189,BUILD-ORIENTATION,4,0, , , , , , ,$,v1.1.a Added,
,3,1,0, , , , , , , , ,$,v1.1.a Added,
,3,2,45, , , , , , , , ,$,v1.1.a Added,
,3,3,90, , , , , , , , ,$,v1.1.a Added,
,3,4,135, , , , , , , , ,$,v1.1.a Added,
,3,5,180, , , , , , , , ,$,v1.1.a Added,
,3,6,225, , , , , , , , ,$,v1.1.a Added,
,3,7,270, , , , , , , , ,$,v1.1.a Added,
,3,8,315, , , , , , , , ,$,v1.1.a Added,
,2,190,GAS-AVAILABILITY,4,YES, , , , , , ,$,v1.2.a Added,
,3,1,YES, , , , , , , , ,$,v1.2.a Added,
,3,2,NO, , , , , , , , ,$,v1.2.a Added,
,2,191,BUILDING-W/K,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v1.2.a Added,
,2,192,BUILDING-ALPHA,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a Added,
,2,193,COOL-SET-POINT-MODE,4,NORMAL, , , , , , ,$,v1.1.a Added,
,3,1,NORMAL, , , , , , , , ,$,v1.1.a Added,
,3,2,OVERHEATING, , , , , , , , ,$,v1.1.a Added,
,3,3,MIXED-MODE, , , , , , , , ,$,v1.1.a Added,
,2,194,PAUSE-KNOWN-ERRORS,4,YES,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R16
Added,
,3,1,YES, , , , , , , , ,$,v3.0.a_dev_R16 Added,
,3,2,NO, , , , , , , , ,$,v3.0.a_dev_R16 Added,
,2,195,MAIN-FUEL-TYPE,4,Other,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.3.c_dev_R10 Mod,
,3,1,Natural Gas, , , , , , , , ,$,v3.0.a_dev_R29 Added,
,3,2,LPG, , , , , , , , ,$,v3.0.a_dev_R29 Added,
,3,3,Biogas, , , , , , , , ,$,v3.0.a_dev_R29 Added,
,3,4,Oil, , , , , , , , ,$,v3.0.a_dev_R29 Added,
,3,5,Coal, , , , , , , , ,$,v3.0.a_dev_R29 Added,
,3,6,Anthracite, , , , , , , , ,$,v3.0.a_dev_R29 Added,
,3,7,Smokeless Fuel (inc Coke), , , , , , , , ,$,v3.0.a_dev_R29 Added,
,3,8,Dual Fuel Appliances (Mineral + Wood), , , , , , , , ,$,v3.0.a_dev_R29 Added,
,3,9,Biomass, , , , , , , , ,$,v3.0.a_dev_R29 Added,
,3,10,Grid Supplied Electricity, , , , , , , , ,$,v3.0.a_dev_R29 Added,
,3,11,Grid Displaced Electricity, , , , , , , , ,$,v3.0.a_dev_R29 Added,
,3,12,Waste Heat, , , , , , , , ,$,v3.0.a_dev_R29 Added,
,3,13,District Heating, , , , , , , , ,$,v3.0.b_dev_R01 Added,
,3,14,Other, , , , , , , , ,$,v3.3.c_dev_R10 Mod,
```

```
,2,196,BUILDING-ENVIRONMENT,4,Unconditioned,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v3.3.c_dev_R10 Mod,
,3,1,Heating and Natural Ventilation, , , , , , , , ,$,v3.0.a_dev_R30 Added,
,3,2,Heating and Mechanical Ventilation, , , , , , , , ,$,v3.0.a_dev_R30 Added,
,3,3,Air Conditioning, , , , , , , , ,$,v3.0.a_dev_R30 Added,
,3,4,Mixed-mode with Natural Ventilation, , , , , , , , ,$,v3.0.a_dev_R30 Added,
,3,5,Mixed-mode with Mechanical Ventilation, , , , , , , , ,$,v3.0.a_dev_R30 Added,
,3,6,Unconditioned, , , , , , , , ,$,v3.3.c_dev_R10 Mod,
,2,197,RENEWABLE-MAIN,4,From calculation,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v4.0.a_dev_R29_a05 Modified,
,3,1,From calculation, , , , , , , , ,$,v4.0.a_dev_R29_a05 Modified,
,3,2,Other non-calculated, , , , , , , , ,$,v4.0.a_dev_R29_a05 Modified,
,2,198,DH-CO2F,1,0.293,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.b_dev_R01 Added,
,2,199,DH-PRIMF,1,1.2,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.3.c_dev_R25 Added,
,2,200,PRODUCE-MESSAGES,4,YES,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.2.c_dev_R015
Added,
,3,1,YES, , , , , , , , ,$,v3.2.c_dev_R015 Added,
,3,2,NO, , , , , , , , ,$,v3.2.c_dev_R015 Added,
,2,201,HTML-DR-REPORTS,4,NO,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.1.b_dev_R04_a01
Added,
,3,1,YES, , , , , , , , ,$,v3.1.b_dev_R04_a01 Added,
,3,2,NO, , , , , , , , ,$,v3.1.b_dev_R04_a01 Added,
,2,202,HTTP-PROXY-SERVER,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.2.c_dev R11 Added,
,2,203,HTTP-PROXY-PORT,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.2.c_dev R11 Added,
,2,204,REA-DATA,2,"",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.3.c_dev_R38 Added,
,2,205,MAX-STOREY,1,1,1,-6666,1,-6666,1,-6666, ,$,v4.0.a_dev_R32 Added,
,2,206,INTERMIT-CALC,4,CEN2008, , , , , , ,$,v4.0.b_dev_R70_a02 Added,
,3,1,CEN2004, , , , , , , , ,$,v4.0.b_dev_R70_a02 Added,
,3,2,CEN2008, , , , , , , , ,$,v4.0.b_dev_R70_a02 Added,
,2,207,LANDMARK-ENVIRONMENT,4,www, , , , , , ,$,v4.0.c_dev_R03 Added,
,3,1,www, , , , , , , , ,$,v4.0.c_dev_R03 Added,
,3,2,uat, , , , , , , , ,$,v4.0.c_dev_R03 Added,
,3,3,ote, , , , , , , , ,$,v4.0.c_dev_R03 Added,
,1,16,COMPLIANCE,1,1, ,0, , , , ,$,v5.0.0 Added,
,2,1,BR-TYPE,4,None, , , , , , ,$,v1.2.a Modified 3.0.a_R17 Modified,
,3,1,None, , , , , , , , ,$,v1.2.a Modified 3.0.a_R17 Modified,
,3,2,England and Wales Building Regulations Part L 2010, , , , , , , , ,$,v1.2.a Modified 3.0.a_R17 Modified,
,3,3,Scottish Building Regulations 2010, , , , , , , , ,$,v1.2.a Modified 3.0.a_R17 Modified,
,3,4,Northern Ireland Building Regulations Part F 2010, , , , , , , , ,$,v1.2.a Modified 3.0.a_R17 Modified,
,3,5,Republic of Ireland: Building Regulations Part L, , , , , , , , ,$,v3.1.b Added,
,3,6,Building Bye-laws Jersey Part 11, , , , , , , , ,$,v3.3.c_dev_R04 Added,
,2,2,EPC-TYPE,4,None, , , , , , ,$,v1.2.a Modified 3.0.a_R17 Modified,
,3,1,None, , , , , , , , ,$,v1.2.a Modified 3.0.a_R17 Modified,
,3,2,EPC England, , , , , , , , ,$,v1.2.a Modified 3.0.a_R17 Modified,
,3,3,EPC Wales, , , , , , , , ,$,v1.2.a Modified 3.0.a_R17 Modified,
,3,4,EPC Scotland, , , , , , , , ,$,v1.2.a Modified 3.0.a_R17 Modified,
,3,5,EPC Northern Ireland, , , , , , , , ,$,v1.2.a Modified 3.0.a_R17 Modified,
,3,6,Republic of Ireland: Building Energy Rating, , , , , , , , ,$,v3.1.b Added,
,2,3,EPC-LANGUAGE,4,ENGLISH, , , , , , ,$,v3.0.a_R17 Added,
,3,1,ENGLISH, , , , , , , , ,$,v3.0.a_R17 Added,
,3,2,WELSH, , , , , , , , ,$,v3.0.a_R17 Added,
,2,4,EPC+BR-CHECK,4,NO, , , , , , ,$,v3.0.a_R17 Added,
,3,1,YES, , , , , , , , ,$,v3.0.a_R17 Added,
,3,2,NO, , , , , , , , ,$,v3.0.a_R17 Added,
,2,5,S-EPC-PER,1,-7777,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_R19 Added,
,2,6,BR-S6-BTYPE,4,Other buildings, , , , , , ,$,v2.0.a Added,
,3,1,Other buildings, , , , , , , , ,$,v2.0.a Added,
,3,2,Shell buildings, , , , , , , , ,$,v2.0.a Added,
,3,3,Extensions to the insulation envelope, , , , , , , , ,$,v2.0.a Added,
,2,7,BR-S6-ACCRED-CONST,4,NO, , , , , , ,$,v2.0.a Added,
,3,1,YES, , , , , , , , ,$,v2.0.a Added,
```

```
,3,2,NO, , , , , , , , , ,$,v2.0.a Added,
,2,8,BR-S6-NV-CREDIT,4,NO, , , , , , , ,$,v2.0.a Added,
,3,1,YES, , , , , , , , , ,$,v2.0.a Added,
,3,2,NO, , , , , , , , , ,$,v2.0.a Added,
,2,9,BR-STAGE,4,As built, , , , , , , ,$,v1.2.a Added - v4.0.a_dev_R16 Modified,
,3,1,As designed, , , , , , , , , ,$,v1.2.a Added,
,3,2,As built, , , , , , , , , ,$,v1.2.a Added,
,2,10,BER-STAGE,4,New Building - Provisional, , , , , , , ,$,v3.4.b_dev_R13_a05 Modified,
,3,1,Existing Building - Final, , , , , , , , , ,$,v3.4.b_dev_R13_a05 Modified,
,3,2,New Building - Final, , , , , , , , , ,$,v3.4.b_dev_R13_a05 Modified,
,3,3,New Building - Provisional, , , , , , , , , ,$,v3.4.b_dev_R13_a05 Modified,
,2,11,ROOF-U-MAX,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,12,WALL-U-MAX,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,13,FLOOR-U-MAX,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,14,WINDOW-U-MAX,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,15,P-DOOR-U-MAX,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,16,V-DOOR-U-MAX,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,17,H-DOOR-U-MAX,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,18,S-VENT-U-MAX,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,19,ROOFLIGHT-U-MAX,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,20,WINROOF-U-MAX,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,21,ROOF-U-MIN,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09_a02 Added,
,2,22,WALL-U-MIN,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09_a02 Added,
,2,23,FLOOR-U-MIN,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09_a02 Added,
,2,24,P-DOOR-U-MIN,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09_a02 Added,
,2,25,V-DOOR-U-MIN,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09_a02 Added,
,2,26,H-DOOR-U-MIN,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09_a02 Added,
,2,27,WINROOF-U-MIN,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09_a02 Added,
,2,28,Q50-INF-AVE,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,29,Q50-INF-CHECK,4,YES, , , , , , ,$,v1.3.a Added,
,3,1,YES, , , , , , , , , ,$,v1.3.a Added,
,3,2,NO, , , , , , , , , ,$,v1.3.a Added,
,2,30,ROOF-U-AVE,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,31,WALL-U-AVE,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,32,FLOOR-U-AVE,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,33,WINDOW-U-AVE,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,34,P-DOOR-U-AVE,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,35,V-DOOR-U-AVE,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,36,H-DOOR-U-AVE,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,37,S-VENT-U-AVE,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,38,ROOFLIGHT-U-AVE,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,39,WINROOF-U-AVE,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,40,BUILD-U-AVE,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,41,ROOF-WIN-U-AVE,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,42,WALL-WIN-U-AVE,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,43,ROOF-AREA,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,44,WALL-AREA,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,45,FLOOR-AREA,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,46,WINDOW-AREA,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,47,ROOFLIGHT-AREA,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.c_dev_R09 Added,
,2,48,P-DOOR-AREA,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,49,V-DOOR-AREA,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,50,H-DOOR-AREA,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,51,S-VENT-AREA,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,52,ROOF-WIN-AREA,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,53,WALL-WIN-AREA,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,54,WALL-WIN-U-MAX,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,55,ROOF-WIN-U-MAX,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,56,IF,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.0 Moved to compliance,
,2,57,RB,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.0 Moved to compliance,
,2,58,TER,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.2 Moved to compliance,
```

```
,2,59,KG-CO2-REF,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R05 Added,
,2,60,KG-CO2-REF-IFRB,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R05 Added,
,2,61,KG-CO2-TYP,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R05 Added,
,2,62,PASS-CODE,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.0 Moved to compliance,
,2,63,PASS-CODE-CO2-IR,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.1.b_dev_R03_a02
Added,
,2,64,PASS-CODE-ENERGY-IR,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.1.b_dev_R03_a02
Added,
,2,65,PASS-CODE-CO2-JER,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.3.c_dev_R15 Added,
,2,66,PASS-CODE-PRIM-JER,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v4.0.a_dev_R63_public_a03 Added,
,2,67,PASS-CODE-ENERGY-JER,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.3.c_dev_R15
Added,
,2,68,CPC-IR,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.1.b_dev_R03_a02 Added,
,2,69,EPC-IR,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.1.b_dev_R03_a02 Added,
,2,70,CPT-JER,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.1.b_dev_R03_a02 Added,
,2,71,EPT-PRIM-JER,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R63_public_a03 Added,
,2,72,EPT-JER,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.1.b_dev_R03_a02 Added,
,2,73,KG-CO2-NOT,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v04.3.3 Added,
,2,74,KWH-PRIM-IR-NOT,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.1.b_dev_R03_a02 Added,
,2,75,KWH-PRIM-IR-REF,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.1.b_dev_R04_a02 Added,
,2,76,KWH-PRIM-JER-NOT,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.3.c_dev_R14 Added,
,2,77,KWH-JER-NOT,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R63_public_a03 Added,
,2,78,KG-CO2-NOT-AGGREGATE,1,0,-6666,-6666,-6666,-6666,-6666,-6666,kg,$,2010 1.0.0_dev_R01
Added,
,2,79,WALL-MAX,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,80,FLOOR-MAX,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,81,ROOF-MAX,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,82,WINROOF-MAX,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.0 Added,
,2,83,P-DOOR-MAX,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.1 Added ID9c,
,2,84,H-DOOR-MAX,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.1 Added ID9c,
,2,85,V-DOOR-MAX,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.1 Added ID9c,
,2,86,S-VENT-MAX,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.1 Added ID9c,
,2,87,WALL-MIN,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09_a02 Added,
,2,88,FLOOR-MIN,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09_a02 Added,
,2,89,ROOF-MIN,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09_a02 Added,
,2,90,WINROOF-MIN,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09_a02 Added,
,2,91,P-DOOR-MIN,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09_a02 Added,
,2,92,H-DOOR-MIN,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09_a02 Added,
,2,93,V-DOOR-MIN,1,-6666,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.c_dev_R09_a02 Added,
,2,94,OVERHEAT-RISK,4,NO,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R04_AN added,
,3,1,YES, , , , , , , , ,$,v3.0.a_dev_R04_AN added,
,3,2,NO, , , , , , , , ,$,v3.0.a_dev_R04_AN added,
,2,95,ENG-HERITAGE,4,NO,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R32 Added,
,3,1,YES, , , , , , , , ,$,v3.0.a_dev_R32 Added,
,3,2,NO, , , , , , , , ,$,v3.0.a_dev_R32 Added,
,2,96,BUILDING-
W/M2K,1,EXP,0,1000000000,0,1000000000,0,1000000000,W/m2K,$,v3.1.b_dev_R04_a03 Added,
,2,97,BUILDING-EXT-
AREA,1,EXP,0,1000000000,0,1000000000,0,1000000000,m2,$,v3.1.b_dev_R04_a03 Added,
,2,98,AIR-CON-INSTALLED,4,Unknown,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.3.c_dev_R11
Added,
,3,1,Yes, , , , , , , , ,$,v3.3.c_dev_R11 Added,
,3,2,No, , , , , , , , ,$,v3.3.c_dev_R11 Added,
,3,3,Unknown, , , , , , , , ,$,v3.3.c_dev_R11 Added,
,2,99,AIR-CON-KW,4,Greater than 250kW,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.3.c_dev_R11
Added,
,3,1,Below 12kW, , , , , , , , ,$,v3.3.c_dev_R11 Added,
,3,2,Between 12kW and 250kW, , , , , , , , ,$,v3.3.c_dev_R11 Added,
,3,3,Greater than 250kW, , , , , , , , ,$,v3.3.c_dev_R11 Added,
,3,4,EXACT, , , , , , , , ,$,v3.3.c_dev_R11 Added,
```

```
,2,100,AIR-CON-INSPECTION,4,Don't know,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.3.c_dev_R11 Added,
,3,1,Yes - inspection completed, , , , , , , , ,$,v3.3.c_dev_R11 Added,
,3,2,Yes - inspection commissioned, , , , , , , , ,$,v3.3.c_dev_R11 Added,
,3,3,No inspection completed or commissioned, , , , , , , , ,$,v3.3.c_dev_R11 Added,
,3,4,Not relevant, , , , , , , , ,$,v3.3.c_dev_R11 Added,
,3,5,Don't know, , , , , , , , ,$,v3.3.c_dev_R11 Added,
,2,101,AIR-CON-KW-EXACT,1,0,0,-6666,0,-6666,0,-6666,NC,$,v3.3.c_dev_R11 Added,
,2,102,SOLAR-GAIN-CHECK,4,NO, , , , , , , ,$,2010 1.0.0_dev_R01 Added - Changed on R07 after release,
,3,1,YES, , , , , , , , ,$,2010 1.0.0_dev_R01 Added,
,3,2,NO, , , , , , , , ,$,2010 1.0.0_dev_R01 Added,
,2,103,SOLAR-GAIN-OVER-LIMIT,4,NO, , , , , , , ,$,2010 1.0.0_dev_R01 Added,
,3,1,YES, , , , , , , , ,$,2010 1.0.0_dev_R01 Added,
,3,2,NO, , , , , , , , ,$,2010 1.0.0_dev_R01 Added,
,2,104,SHELL-AND-CORE,4,NO,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R18 Added,
,3,1,YES, , , , , , , , ,$,v4.0.a_dev_R18 Added,
,3,2,NO, , , , , , , , ,$,v4.0.a_dev_R18 Added,
,2,105,SC-ACTUAL-RATIO,1,EXP,0,1,0,1,0,1,ratio,$,v4.0.a_dev_R18 Added,
,2,106,MODULAR-PORTABLE,4,NO,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R19 Added,
,3,1,YES, , , , , , , , ,$,v4.0.a_dev_R19 Added,
,3,2,NO, , , , , , , , ,$,v4.0.a_dev_R19 Added,
,2,107,MODULAR-YEARS,4,More than 2 years,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R19 Added,
,3,1,Less than 2 years, , , , , , , , ,$,v4.0.a_dev_R19 Added,
,3,2,More than 2 years, , , , , , , , ,$,v4.0.a_dev_R19 Added,
,2,108,MODULAR-CORRECTION,4,NO,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R19 Added,
,3,1,YES, , , , , , , , ,$,v4.0.a_dev_R19 Added,
,3,2,NO, , , , , , , , ,$,v4.0.a_dev_R19 Added,
,2,109,MODULAR-ADL2-T3,4,Post October 2010,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R19 Added,
,3,1,Post October 2010, , , , , , , , ,$,v4.0.a_dev_R19 Added,
,3,2,April 2006 - September 2010, , , , , , , , ,$,v4.0.a_dev_R19 Added,
,3,3,April 2002 - March 2006, , , , , , , , ,$,v4.0.a_dev_R19 Added,
,3,4,Pre April 2002, , , , , , , , ,$,v4.0.a_dev_R19 Added,
,2,110,MODULAR-TER-FACTOR,1,1,0,-6666,0,-6666,0,-6666,factor,$,v4.0.a_dev_R23_a03 Added,
,2,111,ACCRED-CONST-DETAILS,4,NO, , , , , , ,$,v4.0.c_dev_R09_a02 Added,
,3,1,YES, , , , , , , , ,$,v4.0.c_dev_R09_a02 Added,
,3,2,NO, , , , , , , , ,$,v4.0.c_dev_R09_a02 Added,
,1,17,DOOR,0,-6666,WALL,1, , , , ,$,v5.0.1 Added,
,2,1,TYPE,4,Personnel Doors,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.1 Added,
,3,1,Personnel Doors, , , , , , , , ,$,v5.0.1 Added,
,3,2,Vehicle Access Doors, , , , , , , , ,$,v5.0.1 Added,
,3,3,High Usage Entrance Doors, , , , , , , , ,$,v5.0.1 Added,
,2,2,AREA,1,REQ,0.00001,1000000000,0.1,5000,0,1000000000,m2,$,v5.0.1 Added ,
,2,3,U-VALUE,1,EXP,0,10,0,10,0,10, ,$,v5.0.1 Added,
,2,4,CM,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.1 Added,
,2,5,CONSTRUCTION,3,REQ,CONSTRUCTION, , , , , ,$,v5.0.1 Added,
,2,6,H-TR,1,EXP,0,1000000000,0,1000000000,0,1000000000, ,$,v5.0.1 Added,
,2,7,Q-SUN-NT,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.1 Added,
,2,8,MULTIPLIER,1,1,1,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.4 Added,
,2,9,TM-TYPE,4,ALPHA, , , , , , ,$,v5.1.9 Added,
,3,1,ALPHA, , , , , , , , ,$,v5.1.9 Added,
,3,2,TM, , , , , , , , ,$,v5.1.9 Added,
,2,10,ALPHA,1,0.15,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.01.9 Added,
,2,11,CPS-COMP-FLAG,4,YES, , , , , , ,$,v2.0.c_R05,
,3,1,YES, , , , , , , , ,$,v2.0.c_R05,
,3,2,NO, , , , , , , , ,$,v2.0.c_R05,
,1,18,DOOR-TM-BRIDGE,0,-6666,DOOR,1, , , , ,$,v05.0.4.Added,
```

```
,2,1,PSI,1,2,-6666,-6666,0.01,2.5,-6666,-6666,W/mK,$,v05.0.4.Added,
,2,2,L,1,10,-6666,-6666,0.01,1000,-6666,-6666,m,$,v05.0.4.Added,
,2,3,H-TR,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666, ,$,v05.0.4.Added,
,2,4,MULTIPLIER,1,1,1,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.4 Added,
,1,19,SMOKE-VENT,0,-6666,WALL,1, , , , ,$,v5.0.1 Added,
,2,1,AREA,1,REQ,0.00001,1000000000,0.1,1000000000,0,1000000000,m2,$,v5.0.1 Added ID9c,
,2,2,U-VALUE,1,REQ,0,10,0,10,0,10, ,$,v5.0.1 Added ID9c,
,2,3,H-TR,1,EXP,0,1000000000,0,1000000000,0,1000000000, ,$,v5.0.1 Added ID9c,
,2,4,Q-SUN-NT,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v5.0.1 Added ID9c,
,2,5,MULTIPLIER,1,1,1,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.4 Added,
,2,6,TM-TYPE,4,ALPHA, , , , , , ,$,v5.1.9 Added,
,3,1,ALPHA, , , , , , , , ,$,v5.1.9 Added,
,3,2,TM, , , , , , , , ,$,v5.1.9 Added,
,2,7,ALPHA,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.01.9 Added,
,1,20,CCHP-GENERATOR,0,1, ,0, , , , ,$,v05.0.8 Added,
,2,1,TRIGENERATION,4,NO, , , , , , ,$,v3.2.c_R19 Added,
,3,1,YES, , , , , , , , ,$,v2.0.c_R05,
,3,2,NO, , , , , , , , ,$,v2.0.c_R05,
,2,2,HEAT-SEER,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added v3.2.c_R19 Mod,
,2,3,COOL-SEER,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.2.c_R19 Added,
,2,4,CHPQA_QI,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.2.c_R19 Added,
,2,5,BUILD-CHP-SHEAT,1,0,0,1,0,1,-6666,-6666,NC,$,v05.0.8 Added v3.0.a_R50 Mod,
,2,6,BUILD-CHP-DHEAT,1,0,0,1,0,1,-6666,-6666,NC,$,v05.0.8 Added v3.0.a_R50 Mod,
,2,7,BUILD-CHP-SCOOL,1,0,0,1,0,1,-6666,-6666,NC,$,v3.2.c_R19 Added,
,2,8,HEAT-POWER-RATIO,1,1.5,0,20,1,10,0,20,NC,$,v05.0.8 Added,
,2,9,Q-DEM-DHW-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,10,Q-DEM-DHW,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,11,Q-DEM-HEAT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,12,Q-DEM-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,13,Q-DEM-HEATCOOL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.2.c_R19 Added,
,2,14,Q-DEM-HEATCOOL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.2.c_R19 Added,
,2,15,KWH-DHW-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,16,KWH-DHW,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,17,KWH-HEAT-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.2.c_R19 Added,
,2,18,KWH-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.2.c_R19 Added,
,2,19,KWH-HEATCOOL-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,20,KWH-HEATCOOL,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,21,KWH-DISP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,22,KWH-DISP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,23,KG-CO2-DISP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,24,KG-CO2-DISP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,25,KG-CO2-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,26,KG-CO2,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,27,FUEL-TYPE,4,Natural gas,1,12, , , , ,$,v05.0.8 Added,
,3,1,Natural Gas, , , , , , , , ,$,v05.0.8 Added,
,3,2,LPG, , , , , , , , ,$,v05.0.8 Added,
,3,3,Biogas, , , , , , , , ,$,v05.0.8 Added,
,3,4,Oil, , , , , , , , ,$,v05.0.8 Added,
,3,5,Coal, , , , , , , , ,$,v05.0.8 Added,
,3,6,Anthracite, , , , , , , , ,$,v05.0.8 Added,
,3,7,Smokeless Fuel (inc Coke), , , , , , , , ,$,v1.1 BUG from 1.0 corrected,
,3,8,Dual Fuel Appliances (Mineral + Wood), , , , , , , , ,$,v1.1 BUG from 1.0 corrected,
,3,9,Biomass, , , , , , , , ,$,v05.0.8 Added,
,3,10,Grid Supplied Electricity, , , , , , , , ,$,v05.0.8 Added,
,3,11,Grid Displaced Electricity, , , , , , , , ,$,v05.0.8 Added,
,3,12,Waste Heat, , , , , , , , ,$,v05.0.8 Added,
,3,13,District Heating, , , , , , , , ,$,v3.0.b_dev_R01 Added,
,1,21,WIND-GENERATOR,0,-6666, ,1, , , , ,$,v05.0.8 Added,
,2,1,DIAMETER,1,EXP,0.2,50,0.2,25,0.2,10,m,$,v05.0.8 Added v3.0.a_R14 Mod,
,2,2,HUB-HEIGHT,1,10,0,-6666,1,100,-6666,-6666,m,$,v05.0.8 Added,
,2,3,POWER,1,50,0,4000,0.5,1000,0,4000,kW,$,v05.0.8 Added,
```

```
,2,4,TERRAIN-TYPE,4,Suburban or industrial area,1,6, , , , ,$,v1.1.a Modified,
,3,1,Smooth flat country (no obstacles), , , , , , , , ,$,v1.1.a Modified,
,3,2,Farm land with boundary hedges, , , , , , , , ,$,v1.1.a Modified,
,3,3,Suburban or industrial area, , , , , , , , ,$,v1.1.a Modified,
,3,4,Urban with average building height > 15m, , , , , , , , ,$,v1.1.a Modified,
,2,5,MULTIPLIER,1,1,1,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,6,AREA,1,REQ,0.1,1963,0.1,491,0.1,79,m2,$,v05.0.8 Added v3.0.a_R14 Mod,
,2,7,SPEED-CORR-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,8,AVG-POWER-DENS-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,9,EFF-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,10,KWH-DISP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,11,KWH-DISP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,12,KG-CO2-DISP-M,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,2,13,KG-CO2-DISP,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v05.0.8 Added,
,1,22,REC-USER,0,-6666, ,0, , , , ,$,v2.0.d_dev_R21 Added,
,2,1,CATEGORY,4,OTHER,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R21 Added _R40
Mod,
,3,1,HEATING, , , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,2,COOLING, , , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,3,HOT-WATER, , , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,4,LIGHTING, , , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,5,RENEWABLES, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,6,OVERHEATING, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,7,ENVELOPE, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,8,FUEL-SWITCHING, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,9,AUXILIARY, , , , , , , , ,$,v3.0.a_dev_R01_CD Added ,
,3,10,OTHER, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,2,2,CODE,4,REQ,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R21 Added _R40 Mod,
,3,1,USER, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,2,EPC-C1, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,3,EPC-C2, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,4,EPC-C3, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,5,EPC-W1, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,6,EPC-W3, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,7,EPC-W4, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,8,EPC-W2, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,9,EPC-E1, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,10,EPC-E2, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,11,EPC-E3, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,12,EPC-E4, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,13,EPC-E5, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,14,EPC-E6, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,15,EPC-E7, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,16,EPC-E8, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,17,EPC-F1, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,18,EPC-F2, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,19,EPC-F3, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,20,EPC-F4, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,21,EPC-F5, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,22,EPC-F6, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,23,EPC-H2, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,24,EPC-H5, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,25,EPC-H6, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,26,EPC-H7, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,27,EPC-H8, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,28,EPC-H1, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,29,EPC-H3, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,30,EPC-H4, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,31,EPC-L1, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,32,EPC-L2, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,33,EPC-L3, , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
```

```
,3,34,EPC-L4, , , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,35,EPC-L5, , , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,36,EPC-L6, , , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,37,EPC-L7, , , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,38,EPC-V1, , , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,39,EPC-R1, , , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,40,EPC-R2, , , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,41,EPC-R3, , , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,42,EPC-R4, , , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,3,43,EPC-R5, , , , , , , , , ,$,v2.0.d_dev_R40 Added _R47 Modified,
,2,3,USER-IN-OUT,4,IN,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R41 Added,
,3,1,IN, , , , , , , , , ,$,v2.0.d_dev_R41 Added,
,3,2,OUT, , , , , , , , , ,$,v2.0.d_dev_R41 Added,
,2,4,REC-TEXT,2,"The assessor must introduce the text for the recommendation here",-6666,-6666,-
6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R40 Modified,
,2,5,REC-COMMENT,2," ",-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R40 Added,
,2,6,ENERGY-IMPACT,4,UNKNOWN,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R40
Added,
,3,1,HIGH, , , , , , , , , ,$,v2.0.d_dev_R40 Added,
,3,2,MEDIUM, , , , , , , , , ,$,v2.0.d_dev_R40 Added,
,3,3,LOW, , , , , , , , , ,$,v2.0.d_dev_R40 Added,
,3,4,UNKNOWN, , , , , , , , , ,$,v2.0.d_dev_R40 Added,
,2,7,CO2-IMPACT,4,UNKNOWN,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R40 Added,
,3,1,HIGH, , , , , , , , , ,$,v2.0.d_dev_R40 Added,
,3,2,MEDIUM, , , , , , , , , ,$,v2.0.d_dev_R40 Added,
,3,3,LOW, , , , , , , , , ,$,v2.0.d_dev_R40 Added,
,3,4,UNKNOWN, , , , , , , , , ,$,v2.0.d_dev_R40 Added,
,2,8,PAY-BACK,4,UNKNOWN,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R40 Added,
,3,1,LONG, , , , , , , , , ,$,v2.0.d_dev_R40 Added,
,3,2,MEDIUM, , , , , , , , , ,$,v2.0.d_dev_R40 Added,
,3,3,SHORT, , , , , , , , , ,$,v2.0.d_dev_R40 Added,
,3,4,UNKNOWN, , , , , , , , , ,$,v2.0.d_dev_R40 Added,
,2,9,PAY-BACK-NUM,1,-5555,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R48 Added,
,2,10,PAY-BACK-SC,4,CALC,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R48 Added,
,3,1,USER, , , , , , , , , ,$,v2.0.d_dev_R48 Added,
,3,2,CALC, , , , , , , , , ,$,v2.0.d_dev_R48 Added,
,2,11,CO2-SAVE-POUND,4,UNKNOWN,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R40
Added,
,3,1,GOOD, , , , , , , , , ,$,v2.0.d_dev_R40 Added,
,3,2,FAIR, , , , , , , , , ,$,v2.0.d_dev_R40 Added,
,3,3,POOR, , , , , , , , , ,$,v2.0.d_dev_R40 Added,
,3,4,UNKNOWN, , , , , , , , , ,$,v2.0.d_dev_R40 Added,
,2,12,REF-OBJECT-TYPE,4,WHOLE-BUILDING,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R38 Added,
,3,1,WHOLE-BUILDING, , , , , , , , , ,$,v2.0.d_dev_R38 Added_v3.0.a_R21 Mod,
,3,2,HVAC-SYSTEM, , , , , , , , , ,$,v2.0.d_dev_R38 Added,
,3,3,DHW-GENERATOR, , , , , , , , , ,$,v2.0.d_dev_R38 Added_v3.0.a_R21 Mod v3.2.c_R01 mod,
,2,13,REF-OBJECT,3,EXP,REF-OBJECT-TYPE,K, , , , ,$,v2.0.d_dev_R38 Added,
,2,14,REF-OBJ-ISBEM-ID,1,0,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_R21 Added,
,2,15,STATE,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R40 Added,
,1,23,REC-PROJECT,0,1, ,0, , , , ,$,v2.0.d_dev_R21 Added,
,2,1,PERF-DHW-ENERGY,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,2,PERF-DHW-CO2,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
```

```
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,3,PERF-DHW-SYSTEM-ENERGY,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,4,PERF-DHW-SYSTEM-CO2,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,5,DHW-PERCENT-CO2,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD
Added,
,2,6,PERF-HEAT-ENERGY,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,7,PERF-HEAT-CO2,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,8,PERF-HEAT-SYSTEM-ENERGY,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,9,PERF-HEAT-SYSTEM-CO2,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,10,HEAT-PERCENT-CO2,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD
Added,
,2,11,HEAT-PERCENT-ENERGY,1,EXP,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v3.0.a_dev_R04_CD Added,
,2,12,F-PAYBACK-HEAT,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.0.a_dev_R04_CD Added,
,2,13,PERF-COOL-ENERGY,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,14,PERF-COOL-CO2,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,15,PERF-COOL-SYSTEM-ENERGY,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
```

```
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,16,PERF-COOL-SYSTEM-CO2,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,17,COOL-PERCENT-CO2,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD
Added,
,2,18,PERF-LIGHT-SYSTEM-ENERGY,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,19,PERF-LIGHT-SYSTEM-CO2,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,20,PERF-LIGHT-ENERGY-WORST,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,21,LIGHT-PERCENT-CO2,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD
Added,
,2,22,PERF-AUX-SYSTEM-ENERGY,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,23,PERF-AUX-SYSTEM-CO2,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,24,AUX-PERCENT-CO2,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD
Added,
,2,25,CHP-PERCENT-CO2,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v3.4.b_dev_R13_a03
Added,
,2,26,PERF-WINDOWS,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,27,PERF-WALLS,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,28,PERF-ROOFS,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
```

```
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,29,PERF-FLOORS,4,NOT APPLICABLE,-6666,-6666,-6666,-6666,-6666,-
6666,NC,$,v2.0.d_dev_R22_CD Added,
,3,1,GOOD, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,2,FAIR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,3,POOR, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,3,4,NOT APPLICABLE, , , , , , , , ,$,v2.0.d_dev_R22_CD Added,
,2,30,OVERHEATING-RISK,4,High risk, , , , , , ,$,v3.0.a_dev_R01_CD Added ,
,3,1,No risk, , , , , , , , ,$,v3.0.a_dev_R01_CD Added ,
,3,2,Low risk, , , , , , , , ,$,v3.0.a_dev_R01_CD Added ,
,3,3,Moderate risk, , , , , , , , ,$,v3.0.a_dev_R01_CD Added ,
,3,4,Significant risk, , , , , , , , ,$,v3.0.a_dev_R01_CD Added ,
,3,5,High risk, , , , , , , , ,$,v3.0.a_dev_R01_CD Added ,
,2,31,SOLAR-GAIN-OVER-LIMIT,4,NO, , , , , , ,$,v4.0.a_dev_R12_a04 Added,
,3,1,YES, , , , , , , , ,$,v4.0.a_dev_R12_a04 Added,
,3,2,NO, , , , , , , , ,$,v4.0.a_dev_R12_a04 Added,
,2,32,EPC-C1,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,33,EPC-C2,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,34,EPC-C3,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,35,EPC-W1,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,36,EPC-W3,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,37,EPC-W4,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,38,EPC-W2,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,39,EPC-E1,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,40,EPC-E2,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,41,EPC-E3,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,42,EPC-E4,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,43,EPC-E5,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,44,EPC-E6,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,45,EPC-E7,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,46,EPC-E8,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,47,EPC-F1,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,48,EPC-F2,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,49,EPC-F3,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,50,EPC-F4,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,51,EPC-F5,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,52,EPC-F6,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
Modified,
,2,53,EPC-H2,5,{ 2 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD Added _R47
```

```
Modified,
,2,54,EPC-H5,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,55,EPC-H6,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,56,EPC-H7,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,57,EPC-H8,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,58,EPC-H1,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,59,EPC-H3,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,60,EPC-H4,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,61,EPC-L1,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,62,EPC-L2,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,63,EPC-L3,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,64,EPC-L4,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,65,EPC-L5,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,66,EPC-L6,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,67,EPC-L7,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,68,EPC-V1,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,69,EPC-R1,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,70,EPC-R2,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,71,EPC-R3,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,72,EPC-R4,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,2,73,EPC-R5,5,{  2  },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v2.0.d_dev_R22_CD  Added  _R47
Modified,
,1,24,TSC,0,-6666, ,1, , , , , ,$,v4.0.a_dev_R02 Added,
,2,1,TYPE,4,Standard operation, , , , , , ,$,v4.0.a_dev_R02 Added,
,3,1,Standard operation, , , , , , , , ,$,v4.0.a_dev_R02 Added,
,3,2,High temperature rise, , , , , , , , ,$,v4.0.a_dev_R02 Added,
,3,3,High air volume, , , , , , , , ,$,v4.0.a_dev_R02 Added,
,2,2,OPERATION-TYPE,4,Constant flow, , , , , , ,$,v4.0.a_dev_R02 Added,
,3,1,Constant flow, , , , , , , , ,$,v4.0.a_dev_R02 Added,
,3,2,Variable flow, , , , , , , , ,$,v4.0.a_dev_R02 Added,
,2,3,MULTIPLIER,1,1,1,1,1,1,1,1,1,$,v4.0.a_dev_R48 Added,
,2,4,ABSORTIVITY,4,Good, , , , , , ,$,v4.0.a_dev_R02 Added,
,3,1,Very high, , , , , , , , ,$,v4.0.a_dev_R02 Added,
,3,2,High, , , , , , , , ,$,v4.0.a_dev_R02 Added,
,3,3,Good, , , , , , , , ,$,v4.0.a_dev_R02 Added,
,3,4,Moderate, , , , , , , , ,$,v4.0.a_dev_R02 Added,
,3,5,Low, , , , , , , , ,$,v4.0.a_dev_R02 Added,
,2,5,FAN-INDEPENDENT,4,YES, , , , , , ,$,v4.0.a_dev_R02 Added,
,3,1,YES, , , , , , , , ,$,v4.0.a_dev_R02 Added,
,3,2,NO, , , , , , , , ,$,v4.0.a_dev_R02 Added,
,2,6,SFP,1,2,0,-6666,0,-6666,0,-6666,W/ls,$,v4.0.a_dev_R02 Added,
,2,7,SHADING-FACTOR,1,REQ,0,1,0,1,0,1,ratio,$,v4.0.a_dev_R02 Added,
```

```
,2,8,THICKNESS,1,0.2,0,1,0,1,0,1,ratio,$,v4.0.a_dev_R02 Added,
,2,9,AIR-FLOW-RATE-DES,1,REQ,0,-6666,0,-6666,0,-6666,l/s,$,v4.0.a_dev_R02 Added,
,2,10,COLLECTOR-EFFICIENCY,5,EXP,0,100,0,100,0,100, ,$,v4.0.a_dev_R02 Added,
,2,11,WALL-ID,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R02 Added,
,2,12,INWALL-AREA,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R02 Added,
,2,13,WALL-AREA,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R02 Added,
,2,14,ORIENTATION,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R02 Added,
,2,15,INCLINATION,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R02 Added,
,2,16,AREA,1,EXP,0,-6666,0,-6666,0,-6666,W/ls,$,v4.0.a_dev_R40 Added,
,2,17,WALL-N,1,0,0,-6666,0,-6666,0,-6666,W/ls,$,v4.0.a_dev_R40 Added,
,2,18,REP-WIND-SPEED,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R40 Added,
,2,19,REP-SOLAR-GAIN,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R40 Added,
,2,20,REP-HC,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R40 Added,
,2,21,REP-T,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R40 Added,
,2,22,REP-HR,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R40 Added,
,2,23,REP-CEFF,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R40 Added,
,2,24,REP-AIR-FLOW-RATE,5,{     -5555     },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R41
Mod,
,2,25,AIR-FLOW-RATE,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R41 Added,
,2,26,REP-VS,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R40 Added,
,2,27,REP-EHX,1,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R40 Added,
,2,28,REP-INC-TDEL,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R40 Added,
,2,29,Q-TSC,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R40 Added,
,2,30,REP-Q-TSC,5,EXP,0,-6666,0,-6666,0,-6666,W/ls,$,v4.0.a_dev_R47 Added,
,2,31,KWH-AUX,5,EXP,-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R40 Added,
,2,32,REP-TACT,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R43 Added,
,2,33,REP-TREC,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R43 Added,
,2,34,ZONE-N,1,0,0,-6666,0,-6666,0,-6666,W/ls,$,v4.0.a_dev_R43 Added,
,2,35,ZONE-SUPPLY,1,0,0,-6666,0,-6666,0,-6666,W/ls,$,v4.0.a_dev_R43 Added,
,2,36,ZONE-SET-POINT,5,{ -5555 },-6666,-6666,-6666,-6666,-6666,-6666,NC,$,v4.0.a_dev_R43 Added,
,2,37,Q-HEAT-REC,5,EXP,0,-6666,0,-6666,0,-6666,W/ls,$,v4.0.a_dev_R45 Added,
,2,38,Q-TOT,5,EXP,0,-6666,0,-6666,0,-6666,W/ls,$,v4.0.a_dev_R45 Added,
,2,39,Q-TOT-DIRECT,5,EXP,0,-6666,0,-6666,0,-6666,W/ls,$,v4.0.a_dev_R49 Added,
,2,40,CONTROL-TYPE,4,Only manual, , , , , , ,$,v4.0.a_dev_R48 Added,
,3,1,Only manual, , , , , , , , ,$,v4.0.a_dev_R48 Added,
,3,2,Automatic, , , , , , , , ,$,v4.0.a_dev_R48 Added,
```

BEEMLibDef

Coded file

```
$Config file for defaults values in SBEM
$File created on: 12/11/2010 10:51:5512/11/2010 10:51:55
 1 1 D001 1.234
 1 2 D002 330
 1 3 D003 65
 1 4 D004 1.234
 1 5 D005 12
 1 6 D006 1.234
 1 7 D007 2500
 1 8 D008 0.0864
 1 9 D008ACU 0.0036
 1 10 D009 1.234
 1 11 D010 1.234
 1 12 D011 1
 1 13 D012 1.5
 1 14 D013 2.5
 1 15 D014 1.2
 1 16 D015 1.234
 1 17 D016 1.234
 1 18 D017 1.234
 1 19 D018 1
```

```
1 20 D019 1.234
1 21 D020 0.75
1 22 D021 0.5
1 23 D022 1.234
1 24 D023 1.234
1 25 D024 1.234
1 26 D025 1.234
1 27 D026 8
1 28 D027 1.234
1 29 D028 1.234
1 30 D029 2800
1 31 D030 0.5
1 32 D031 8760
1 33 D032 1.234
1 34 D033 1
1 35 D034 0.3
1 36 D035 1.234
1 37 D036 1.234
1 38 D037 1.234
1 39 D038 1
1 40 D039 0.045
1 41 D040 24
1 42 D041 1.234
1 43 D042 1.234
1 44 D043 4
1 45 D044 41800
1 46 D045 11
1 47 D046 1.234
1 48 D047 2800
1 49 D048 0.5
1 50 D049 0.5
1 51 D050 1.234
1 52 D051 0.65
1 53 D052 1.234
1 54 D053 4
1 55 D054 1.234
1 56 D055 1.234
1 57 D056 1.234
1 58 D057 1.234
1 59 D058 1.234
1 60 D059 1.234
1 61 D060 1.234
1 62 LIGHT_GAIN_F 0.9
1 63 DF_WINDOW 45
1 64 DF_ROOFLIGHT 90
1 65 LIGHT_EExt_T3max 10000
1 66 LIGHT_1 100
1 67 LIGHT_2 3.75
1 68 LIGHT_3 5.2
1 69 LIGHT_4 50
1 70 LIGHT_5 15
1 71 LIGHT_6 0.09
1 72 LIGHT_7 0.45
1 73 LIGHT_8 0.2
1 74 LIGHT_RW 0.33
1 75 LIGHT_RF 0.125
1 76 D029_LON 2369
1 77 D029_MAN 2155
1 78 D029_EDI 2299
1 79 D047_LON 2369
1 80 D047_MAN 2155
```

```
1 81 D047_EDI 2299
1 82 LIGHT_CEN_AREA 10000
1 83 LIGHT_EEXT1_MAX 70
1 84 LIGHT_EEXT2_MAX 120
1 85 T3_Eext_ini 6.5
1 86 T3_Eext_end 19.5
1 87 LIGHT_Ein_Eext 2.6
1 88 Earth_Cond 1.5
1 89 Wall_Width 0.3
1 90 dU_INF1 0.0769
1 91 dU_INF2 0.667
1 92 dU_INF3 0.75
1 93 dU_INF4 0.0146
1 94 dU_INF5 0.7
1 95 dU_INF6 0.667
1 96 dU_INF7 0.14
1 97 INT_HEAT_1 3
1 98 INT_HEAT_2 0.2
1 99 INT_HEAT_3 3
1 100 DHW_1 3
1 101 DHW_2 0.17
1 102 DHW_3 18
1 103 DHW_4 0.1425
1 104 DHW_5 4
1 105 DHW_6 15
1 106 DHW_7 0.25
1 107 DHW_8 42
1 108 DHW_9 500
1 109 DHW_10 0.24657
1 110 DHW_11 0.005
1 111 DHW_12 1.76
1 112 DHW_13 12.8
1 113 DHW_14 0.55
1 114 DHW_15 4
1 115 dMIN_DHW_LOSS 0.008354
1 116 DHW_HVAC_SYS_RED 0.95
1 117 DACH 3
1 118 dELEC_CODID 10
1 119 dGAS_CODID 1
1 120 dBIOMASS_CODID 9
1 121 dOIL_CODID 4
1 122 dLPG_CODID 2
1 123 dCOOL_SSEER 1.12
1 124 dNO_COOL_SSEER 1000000
1 125 dHEAT_SSEFF 0.48
1 126 dNO_HEAT_SSEFF 1000000
1 127 dAUX_ENERGY 35.91
1 128 dNO_AUX_ENERGY 0
1 129 dDESTRAT_AUX 1
1 130 dF_Vent 0.9
1 131 dTEMP_GRAD1 2
1 132 dTEMP_GRAD2 1.5
1 133 dAUX_HOURS_ACT 3255
1 134 dAUX_HOURS_NOT 3255
1 135 dCM_GROUND_F 36
1 136 dCM_EXT_W 11.7
1 137 dHS_FT_1_1 1
1 138 dHS_FT_1_2 4
1 139 dHS_FT_1_3 46
1 140 dHS_FT_1_4 7
1 141 dHS_FT_1_5 46
```

```
1 142 dHS_FT_1_6 46
1 143 dHS_FT_1_7 46
1 144 dHS_FT_1_8 46
1 145 dHS_FT_1_9 10
1 146 dHS_FT_1_12 46
1 147 dHS_FT_2_1 2
1 148 dHS_FT_2_2 5
1 149 dHS_FT_2_3 46
1 150 dHS_FT_2_4 8
1 151 dHS_FT_2_5 46
1 152 dHS_FT_2_6 46
1 153 dHS_FT_2_7 46
1 154 dHS_FT_2_8 46
1 155 dHS_FT_2_9 10
1 156 dHS_FT_2_12 46
1 157 dHS_FT_3_1 3
1 158 dHS_FT_3_2 6
1 159 dHS_FT_3_3 46
1 160 dHS_FT_3_4 9
1 161 dHS_FT_3_5 46
1 162 dHS_FT_3_6 46
1 163 dHS_FT_3_7 46
1 164 dHS_FT_3_8 46
1 165 dHS_FT_3_9 10
1 166 dHS_FT_3_12 46
1 167 dHS_FT_4_1 46
1 168 dHS_FT_4_2 46
1 169 dHS_FT_4_3 46
1 170 dHS_FT_4_4 46
1 171 dHS_FT_4_5 46
1 172 dHS_FT_4_6 46
1 173 dHS_FT_4_7 46
1 174 dHS_FT_4_8 46
1 175 dHS_FT_4_9 46
1 176 dHS_FT_4_12 46
1 177 dHS_FT_5_1 21
1 178 dHS_FT_5_2 46
1 179 dHS_FT_5_3 46
1 180 dHS_FT_5_4 24
1 181 dHS_FT_5_10 36
1 182 dHS_FT_5_11 46
1 183 dHS_FT_5_12 46
1 184 dHS_FT_6_1 21
1 185 dHS_FT_6_2 46
1 186 dHS_FT_6_3 46
1 187 dHS_FT_6_4 24
1 188 dHS_FT_6_10 36
1 189 dHS_FT_6_11 46
1 190 dHS_FT_6_12 46
1 191 dHS_FT_7_1 46
1 192 dHS_FT_7_10 46
1 193 dHS_FT_7_11 46
1 194 dHS_FT_8_1 46
1 195 dHS_FT_8_2 46
1 196 dHS_FT_8_3 46
1 197 dHS_FT_8_4 46
1 198 dHS_FT_8_10 46
1 199 dHS_FT_8_11 46
1 200 dHS_FT_9_1 46
1 201 dHS_FT_9_2 46
1 202 dHS_FT_9_3 46
```

```
1 203 dHS_FT_9_4 46
1 204 dHS_FT_9_10 46
1 205 dHS_FT_9_11 46
1 206 dHS_FT_10_1 25
1 207 dHS_FT_10_10 46
1 208 dHS_FT_10_11 46
1 209 dHS_FT_11_10 46
1 210 dHS_FT_11_11 46
1 211 dHS_FT_12_1 46
1 212 dHS_FT_12_2 46
1 213 dHS_FT_12_3 46
1 214 dHS_FT_12_4 46
1 215 dHS_FT_12_10 46
1 216 dHS_FT_12_11 46
1 217 dHS_FT_13_1 46
1 218 dHS_FT_13_2 46
1 219 dHS_FT_13_3 46
1 220 dHS_FT_13_4 46
1 221 dHS_FT_13_10 46
1 222 dHS_FT_13_11 46
1 223 dHS_FT_14_1 46
1 224 dHS_FT_14_2 46
1 225 dHS_FT_14_3 46
1 226 dHS_FT_14_4 46
1 227 dHS_FT_14_5 46
1 228 dHS_FT_14_6 46
1 229 dHS_FT_14_7 46
1 230 dHS_FT_14_8 46
1 231 dHS_FT_14_9 46
1 232 dHS_FT_14_10 46
1 233 dHS_FT_14_11 46
1 234 dHS_FT_15_1 46
1 235 dHS_FT_15_2 46
1 236 dHS_FT_15_3 46
1 237 dHS_FT_15_4 46
1 238 dHS_FT_15_5 46
1 239 dHS_FT_15_6 46
1 240 dHS_FT_15_7 46
1 241 dHS_FT_15_8 46
1 242 dHS_FT_15_9 46
1 243 dHS_FT_15_12 46
1 244 dHS_FT_16_1 34
1 245 dHS_FT_16_2 34
1 246 dHS_FT_16_3 34
1 247 dHS_FT_16_4 34
1 248 dHS_FT_16_5 34
1 249 dHS_FT_16_6 34
1 250 dHS_FT_16_7 34
1 251 dHS_FT_16_8 34
1 252 dHS_FT_16_9 34
1 253 dHS_FT_16_12 34
1 254 dU_VALUE_SV 6
1 255 dGRID_DISP_CO2 0.568
1 256 dWind_height 10
1 257 dWind_airdens 1.225
1 258 dPVS_LOSS 0.25
1 259 dSES_LOSS 0.15
1 260 dSES_A2_DEFAULT 0
1 261 dSES_SIGMA_LOOP 0.85
1 262 dSES_HW_TEMP 40
1 263 dSES_VOL_REF 75
```

```
1 264 dSES_PUMP_TIME 2000
1 265 dSES_VOL_RATIO 0.666667
1 266 dSES_U_LOOP_1 5
1 267 dSES_U_LOOP_2 0.5
1 268 dSES_U_LOOP_3 40
1 269 dSES_TEMP_REF_1 11.6
1 270 dSES_TEMP_REF_2 1.18
1 271 dSES_TEMP_REF_3 3.86
1 272 dSES_TEMP_REF_4 1.32
1 273 dSES_SES_VOL_1 5
1 274 dExhFan 1.5
1 275 dALPHA 0.15
1 276 dHG_TYPE 46
1 277 dChiller_def 0.8
1 278 dWind_size 80
1 279 dWindow_ratio 0.7
1 280 dDoor_ratio 0.4
1 281 dSFP_NOT 2
1 282 dFloor_ratio 1.5
1 283 dParasiticPower 0.57
1 284 dParasiticPowerOcc 0.3
1 285 dABSORB_COEFF 0.9
1 286 dEXT_SURF_RES 0.05
1 287 dAbsorbCoeff 0.9
1 288 dSurfaceResistance 0.123
1 289 drad_factor1 2
1 290 drad_factor2 0.33
1 291 dNOT_SFP 2
1 292 dNOT_SFP_EXH 1.8
1 293 dGlazingGfactor 0.9
1 294 dGlazing_L_Solar 0.76
1 295 DWin_Cor_Flat 0.3
1 296 DWin_Cor_Pitch 0.2
1 297 Q50_INF_AREA 500
1 298 Q50_INF_MAX2 15
1 299 Q50_INF_MAX1 10
1 300 dHEAT_ONLY_RED 0.95
1 301 Not_Rooflight_T_factor 0.6
1 302 Not_Window_T_factor 0.8
1 303 In_Res_Floor 0.17
1 304 Ex_Res_Floor 0.04
1 305 Floor_1 0.457
1 306 DOHTemp 27
1 307 DMMTemp 27
1 308 REC_DEF1 0.8
1 309 REC_DEF1a 0.79
1 310 REC_DEF2 0.7
1 311 REC_DEF2a 0.75
1 312 REC_DEF3 0.88
1 313 REC_DEF4 0.7
1 314 REC_DEF5 2.5
1 315 REC_DEF5a 2.4
1 316 REC_DEF6 2
1 317 REC_DEF7 2
1 318 REC_DEF8 0.04
1 319 REC_DEF9 0.005
1 320 REC_DEF10 1.67
1 321 REC_DEF11 2.33
1 322 REC_DEF12 0.9
1 323 REC_DEF13 0.5
1 324 REC_DEF14 0.3
```

```
1 325 REC_DEF15 0.81
1 326 REC_DEF16 3.1
1 327 REC_DEF17 2.2
1 328 REC_DEF18 0.85
1 329 REC_DEF19 1.5
1 330 REC_DEF20 0.89
1 331 REC_DEF21 1.01
1 332 REC_DEF22 1.02
1 333 REC_DEF23 1.015
1 334 REC_DEF24 3.5
1 335 dEPC_C1 3
1 336 dEPC_C2 3.5
1 337 dEPC_C3 7.5
1 338 dEPC_W1 4.15
1 339 dEPC_W3 3.8
1 340 dEPC_W4 4.5
1 341 dEPC_W2 8
1 342 dEPC_E1 15
1 343 dEPC_E2 25
1 344 dEPC_E3 6.5
1 345 dEPC_E4 3.7
1 346 dEPC_E5 4.6
1 347 dEPC_E6 5.6
1 348 dEPC_E7 7
1 349 dEPC_E8 9.3
1 350 dEPC_F1 1.08
1 351 dEPC_F2 3.75
1 352 dEPC_F3 3.81
1 353 dEPC_F4 6.7
1 354 dEPC_F5 6.72
1 355 dEPC_F6 8.4
1 356 dEPC_H2 1.8
1 357 dEPC_H5 5.8
1 358 dEPC_H6 4.8
1 359 dEPC_H7 2.5
1 360 dEPC_H8 5
1 361 dEPC_H1 2.3
1 362 dEPC_H3 6.6
1 363 dEPC_H4 3
1 364 dEPC_L1 0.6
1 365 dEPC_L2 0.85
1 366 dEPC_L3 1.8
1 367 dEPC_L4 2.5
1 368 dEPC_L5 2.8
1 369 dEPC_L6 3.5
1 370 dEPC_L7 5.7
1 371 dEPC_V1 1.7
1 372 dEPC_R1 11.7
1 373 dEPC_R2 15.9
1 374 dEPC_R3 20.2
1 375 dEPC_R4 44.7
1 376 dEPC_R5 9.8
1 377 dCO2Save_Good_Fair 3
1 378 dCO2Save_Fair_Poor 5
1 379 DL_ROOFHEIGHT_GAIN 0.5
1 380 DL_A_O45 12
1 381 DL_A_O90 36
1 382 DL_P_O45 0.4
1 383 DL_P_O90 1
1 384 DL_Depth 6
1 385 MPCPC 1
```

```
1 386 MPEPC 1
1 387 dFrame_Fac_Win 0.1
1 388 dFrame_Fac_Roof 0.3
1 389 dDev_Proj_Win 1
1 390 dDev_Proj_Roof 1.3
1 391 LIGHT_WALL_FLOOR_AREA_LIMIT 8
1 392 dNOT_DISTRICT_HEAT1 0.15
1 393 dNOT_DISTRICT_HEAT2 0.2
1 394 dNOT_COOL1 0.9
1 395 dNOT_HEAT_GAS 1.2
1 396 dNOT_HEAT_OIL 1.09
1 397 dLUMIN_EFF_NOT 55
1 398 dPARAS_PWR_NOT1 0.3
1 399 dPARAS_PWR_NOT2 0.03
1 400 dNOT_WIN_HEIGHT 1.5
1 401 dSOL_WIN_HEIGHT 1
1 402 dSOL_ROOF_AREA1 0.1
1 403 dSOL_ROOF_AREA2 0.2
1 404 dSOL_ZONE_HEIGHT 6
1 405 dDEM_CONT_VENT1 0.6
1 406 dDISP_LIGHT_SWITCH 0.2
1 407 dTSC_TE 0.9
1 408 dTSC_AirDens 1.23
1 409 dTSC_Cp 1006
1 410 dTSC_sb 0.000000056704
1 411 dTSC_thickness 0.008
1 412 dTSC_Offset 3
1 413 dTSC_WindCorr 0.5
```

BEEMLibTab

Coded file

```
$Config file for defaults tables in SBEM
$File created on: 10/11/2010 10:16:46
 1 1 T001 5 2 1 1 1 1 1.1 1 1.1 1 1.1 1
 1 2 T002 10 10 19 19 19 19 19 19 19 19 19 13 13 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19
19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19
19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19
 1 3 T004 7 3 0.03 0.05 0.09 0.07 0.1 0.18 0.11 0.16 0.3 0.15 0.22 0.42 0.2 0.29 0.54 0.26 0.38 0.72 0.5 0.75
1.35
 1 4 T005 1 3 0.3 0.3 0.8
 1 5 Teff_HR_H 1 7 0 0.65 0.6 0.4 0.6 0.65 0.5
 1 6 Teff_HR_C 1 7 0 0.65 0.6 0.4 0.6 0.65 0.5
 1 7 T007 11 2 0 3.4 0.3 3.8 0.5 4.2 0.7 4.8 1 5.4 1.4 6 1.8 6.8 2.2 7.6 2.6 8.4 3 9.2 3 10
 1 8 T008 1 5 15 10 5 3 2
 1 9 T009 10 10 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 1 10 T011 3 2 0.5 0.5 0.5 0.35 1 1
 1 11 T012 20 10 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0.015
0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015
0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015
0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015
0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015
0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015
0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015
 1 12 T013 3 2 55 55 110 180 180 360
 1 13 T014 12 3 0 0.08 0 0 0.25 0 0.04 0.13 0.06 0 0.13 0.06 0 0.25 0.06 0.04 0.13 0.07 0 0 0 0 0.3 0 0.1 0.05
0.06 0 0 0.01 0 0.35 0.01 0.1 0.05 0.07
 1 14 T015 7 3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3
 1 15 T016 4 6 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3
```

```
 1 16 T017 17 4 0.5 0.5 0.5 0.5 3.4 3.1 2.8 1 6.1 5.1 4.4 1 4.7 4.2 3.6 1 4.1 3.7 3.3 1 1.6 1.5 1.4 1 2.6 2.2 2 1
2.1 1.9 1.8 1 1.9 1.8 1.7 1 1 1 1 1 1 1 1 1 0.8 0.8 0.8 0.8 0.75 0.75 0.75 0.75 0.875 0.875 0.875 0.85 0.9 0.9 0.9
0.875 0.925 0.925 0.925 0.9 0.5 0.5 0.5 0.5
 1 17 T018 11 3 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33
 1 18 T019 12 3 1 0.7 0.6 1 0.7 0.6 1 0.7 0.6 1 0.7 0.6 1 0.7 0.6 1 0.7 0.6 1 0.7 0.6 1 0.7 0.6 1 0.7 0.6 1 0.7 0.6
1 0.7 0.6 1 0.7 0.6
 1 19 T020 1 8 1 2 3 4 5 6 7 8
 1 20 T021 10 10 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40
40
 1 21 T022 1 2 0.6 0.1
 1 22 T023 6 2 0.55 0.7 0.6 0.8 0.75 0.75 0.9 0.9 0.75 0.9 1 1
 1 23 T024 20 10 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81
0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81
0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81
0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81
0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.015 0.015 0.015 0.015 0.015
0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015
0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015
0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015
0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015
0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015
0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015 0.015
 1 24 T025 1 12 16 16 16 16 16 17 18.5 18.3 16 16 16 16
 1 25 T026 1 4 0 0.5 1.5 2
 1 26 T027 20 10 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5
1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5
1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5
1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 0.025 0.025 0.025 0.025 0.025 0.025 0.025
0.025 0.025 0.037 0.037 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025
0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025
0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025
0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025
0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025
0.025 0.025 0.025 0.025 0.025 0.025 0.025 0.025
 1 27 T028 1 5 0.1 0.2 0.5 0.8 1
 1 28 T029 1 5 4 0.7 1 12 5
 1 29 T030 10 10 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 1 30 T031 2 2 4 10 6 14
 1 31 T032 10 10 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
 1 32 T034 1 7 0.75 0.45 0.55 0.9 0.9 1.4 0.45
 1 33 T050 1 11 1 1 0.9 0.8 0.7 0.6 0.5 0.4 0.3 0.2 0.1
 1 34 T051 1 3 0.3 0.5 1
 1 35 T052 1 5 0 0.6 0.8 0.3 0.5
 1 36 T053 1 4 1.2 0.8 2 3
 1 37 T100 10 10 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 38 T101 10 10 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 39 T102 1 2 40 115
 1 40 T102_PVS 1 4 0.15 0.12 0.06 0.08
 1 41 T102_SES 1 2 0.45 0.45
 1 42 tSES_SIGMA_ZERO_DEFAULT 1 3 0.9 0.75 0.65
 1 43 T102_SES_A1 1 3 20 6 3
 1 44 T102_SES_IAM 1 3 1 0.94 0.97
 1 45 T102_SES_CORREL_FACTORS 2 6 1.029 -0.065 -0.245 0.0018 0.0215 0 0.863 -0.147 -0.263 0.008
0.029 0.025
 1 46 T102_SES_PIPE_LOSSES 1 2 0.05 0.02
```

```
1 47 T103 1 3 0 1 0.7
1 48 T0F1 2 5 0.26 0.57 0.27 0.54 0.32 0.5 0.35 0.44 0.36 0.4
1 49 LIGHT_FD_0 1 12 0 0 0 0 0 0 0 0 0 0 0 0
1 50 LIGHT_FD_5 1 12 0.472 0.291 0.206 0.16 0.13 0.183 0.159 0.168 0.197 0.282 0.384 0.496
1 51 LIGHT_FD_10 1 12 0.772 0.57 0.389 0.338 0.26 0.3 0.279 0.281 0.334 0.449 0.649 0.828
1 52 LIGHT_FD_15 1 12 0.946 0.776 0.594 0.521 0.367 0.397 0.373 0.389 0.493 0.666 0.895 0.985
1 53 LIGHT_FD_20 1 12 0.995 0.916 0.736 0.617 0.571 0.466 0.436 0.559 0.631 0.833 0.984 0.999
1 54 LIGHT_FD_25 1 12 1 0.977 0.876 0.738 0.654 0.622 0.594 0.646 0.77 0.94 0.998 1
1 55 LIGHT_FD_30 1 12 1 0.998 0.95 0.847 0.749 0.731 0.706 0.752 0.883 0.987 1 1
1 56 LIGHT_FD_35 1 12 1 1 0.989 0.928 0.838 0.822 0.804 0.85 0.96 0.998 1 1
1 57 LIGHT_FD_40 1 12 1 1 0.997 0.974 0.915 0.896 0.877 0.917 0.99 1 1 1
1 58 LIGHT_FD_45 1 12 1 1 0.999 0.991 0.956 0.945 0.938 0.966 0.997 1 1 1
1 59 LIGHT_FD_50 1 12 1 1 1 0.996 0.984 0.973 0.974 0.99 0.999 1 1 1
1 60 LIGHT_FD_55 1 12 1 1 1 1 0.996 0.989 0.993 0.998 1 1 1 1
1 61 LIGHT_FD_60 1 12 1 1 1 1 0.998 0.997 0.997 1 1 1 1 1
1 62 LIGHT_FD_65 1 12 1 1 1 1 1 0.999 0.999 1 1 1 1 1
1 63 LIGHT_FD_70 1 12 1 1 1 1 1 1 1 1 1 1 1 1
1 64 IDEAL_DIMMER_0 1 12 1 1 1 1 1 1 1 1 1 1 1 1
1 65 IDEAL_DIMMER_5 1 12 0.764 0.855 0.897 0.92 0.935 0.909 0.92 0.916 0.902 0.859 0.808 0.752
1 66 IDEAL_DIMMER_10 1 12 0.571 0.712 0.8 0.835 0.87 0.834 0.851 0.846 0.818 0.747 0.646 0.545
1 67 IDEAL_DIMMER_15 1 12 0.428 0.584 0.703 0.747 0.809 0.773 0.792 0.785 0.741 0.646 0.507 0.394
1 68 IDEAL_DIMMER_20 1 12 0.328 0.476 0.611 0.668 0.74 0.722 0.743 0.72 0.665 0.547 0.395 0.298
1 69 IDEAL_DIMMER_25 1 12 0.263 0.392 0.527 0.599 0.669 0.669 0.691 0.656 0.592 0.46 0.318 0.238
1 70 IDEAL_DIMMER_30 1 12 0.219 0.328 0.454 0.534 0.607 0.611 0.634 0.597 0.522 0.389 0.265 0.199
1 71 IDEAL_DIMMER_35 1 12 0.188 0.282 0.393 0.474 0.55 0.556 0.579 0.54 0.459 0.335 0.227 0.17
1 72 IDEAL_DIMMER_40 1 12 0.164 0.246 0.345 0.421 0.497 0.504 0.526 0.487 0.405 0.293 0.199 0.149
1 73 IDEAL_DIMMER_45 1 12 0.146 0.219 0.307 0.376 0.449 0.457 0.478 0.439 0.36 0.261 0.177 0.132
1 74 IDEAL_DIMMER_50 1 12 0.131 0.197 0.276 0.339 0.407 0.415 0.435 0.398 0.325 0.234 0.159 0.119
1 75 IDEAL_DIMMER_55 1 12 0.12 0.179 0.251 0.308 0.371 0.379 0.397 0.362 0.295 0.213 0.145 0.108
1 76 IDEAL_DIMMER_60 1 12 0.11 0.164 0.23 0.282 0.34 0.348 0.364 0.332 0.27 0.195 0.133 0.099
1 77 IDEAL_DIMMER_65 1 12 0.101 0.152 0.213 0.261 0.314 0.322 0.336 0.306 0.25 0.18 0.122 0.092
1 78 IDEAL_DIMMER_70 1 12 0.094 0.141 0.197 0.242 0.292 0.299 0.312 0.285 0.232 0.167 0.114 0.085
1 79 IDEAL_DIMMER_75 1 12 0.088 0.131 0.184 0.226 0.272 0.279 0.291 0.266 0.216 0.156 0.106 0.079
1 80 IDEAL_DIMMER_80 1 12 0.082 0.123 0.173 0.212 0.255 0.261 0.273 0.249 0.203 0.147 0.099 0.074
1 81 IDEAL_DIMMER_85 1 12 0.077 0.116 0.163 0.199 0.24 0.246 0.257 0.234 0.191 0.138 0.094 0.07
1 82 IDEAL_DIMMER_90 1 12 0.073 0.11 0.154 0.188 0.227 0.232 0.243 0.221 0.18 0.13 0.088 0.066
1 83 IDEAL_DIMMER_95 1 12 0.069 0.104 0.145 0.178 0.215 0.22 0.23 0.21 0.171 0.123 0.084 0.063
1 84 IDEAL_DIMMER_100 1 12 0.066 0.099 0.138 0.169 0.204 0.209 0.218 0.199 0.162 0.117 0.08 0.06
1 85 IDEAL_DIMMER_105 1 12 0.063 0.094 0.132 0.161 0.194 0.199 0.208 0.19 0.155 0.112 0.076 0.057
1 86 IDEAL_DIMMER_110 1 12 0.06 0.09 0.126 0.154 0.186 0.19 0.199 0.181 0.148 0.107 0.072 0.054
1 87 IDEAL_DIMMER_115 1 12 0.057 0.086 0.12 0.147 0.177 0.182 0.19 0.173 0.141 0.102 0.069 0.052
1 88 IDEAL_DIMMER_120 1 12 0.055 0.082 0.115 0.141 0.17 0.174 0.182 0.166 0.135 0.098 0.066 0.05
1 89 T3_Eext_630 1 12 0 0.2 2.2 2.1 6.8 9 7.4 3.7 0.7 0 0 0
1 90 T3_Eext_730 1 12 0.3 2 7.3 7.3 13 15.1 13.9 9.9 4.5 0.7 0.7 0.1
1 91 T3_Eext_830 1 12 2.2 6.5 12.5 12.6 19.3 20.9 20 16.6 11 4.2 3.8 1.6
1 92 T3_Eext_930 1 12 5.8 10.6 17.1 18.2 24.7 26 26.1 22.6 16.9 9.4 7.8 4.7
1 93 T3_Eext_1030 1 12 8.7 14 20.7 22.7 28.7 30.6 31.1 26.9 22.2 13.8 10.9 7.6
1 94 T3_Eext_1130 1 12 10.2 15.3 22.5 26.1 31 32.6 34.9 30.6 25 17.1 12.6 9
1 95 T3_Eext_1230 1 12 10.1 15.9 22.4 27.7 33.6 34.8 36.3 32.9 25.9 18.7 12.6 9.1
1 96 T3_Eext_1330 1 12 8.9 13.7 20.4 27.6 33.8 35.4 35.9 33.1 25.4 19 11 7.7
1 97 T3_Eext_1430 1 12 6 10.9 16.8 26.6 32.6 34 34.2 31.8 24.5 17.1 8.2 4.9
1 98 T3_Eext_1530 1 12 2.5 6.7 12.5 24 29.1 30.2 31.1 28.3 21.1 14 3.9 1.6
1 99 T3_Eext_1630 1 12 0.3 2 7.4 18.7 24.4 25.6 26.6 23.1 16.2 9.8 0.6 0.1
1 100 T3_Eext_1730 1 12 0 0.2 2.3 13.4 18.9 20.5 20.7 17 10.5 4.2 0 0
1 101 T3_Eext_1830 1 12 0 0 0.3 7.6 13.2 14.8 14.6 10.5 4.3 0.7 0 0
1 102 T3_Eext_1930 1 12 0 0 0 2.1 6.8 9.1 8.1 3.8 0.7 0 0 0
2 1 tCIBSEA230_headings 1 13 "0.270833333333333" "0.3125" "0.354166666666667"
"0.395833333333333" "0.4375" "0.479166666666667" "0.520833333333333" "0.5625"
"0.604166666666667" "0.645833333333333" "0.6875" "0.729166666666667" "0.770833333333333"
1 103 tCIBSEA230 9 13 41 196 375 525 620 669 689 632 517 357 176 41 13 354 577 708 739 685 581 445
254 113 79 55 31 12 477 651 672 577 415 215 131 109 95 77 55 31 12 340 377 286 126 123 118 118 109 95
```

```
77 55 31 12 36 58 76 99 115 118 118 109 95 77 59 37 49 30 54 76 99 115 118 118 116 122 275 327 291 134
30 54 76 99 115 132 217 420 569 634 555 406 162 30 54 78 118 256 435 597 699 729 669 494 305 108 117
263 409 530 605 642 657 613 522 393 239 111 31
 1 104 LIGHT_FOC 1 5 0.95 0.95 0.9 0.9 0.82
 1 105 tLUMINOUS-EFFICACY 3 18 0 0 0 0 0 0 27.5 7.5 22.5 25 27.5 32.5 36.25 25 22.5 35 37.5 22.5 0 0
0 0 0 0 27.5 7.5 22.5 25 27.5 32.5 36.25 25 22.5 35 37.5 22.5 0 0 0 0 0 0 0 33 9 27 30 33 39 43.5 39 27 42 45
27
 1 106 ACT_PDC 3 3 0.069 0.069 0.069 0.008 0.008 0.008 2.11 2.11 2.11
 1 107 NJ 1 12 31 28 31 30 31 30 31 31 30 31 30 31
 1 108 NJ_weekday 1 12 22 21 21 21 20 21 23 20 22 23 20 21
 1 109 tn50Q4 1 3 0.2 0.2 0.2
 1 110 tU_INF3 1 3 0.75 0.55 0.35
 1 111 tTCheat 2 2 1 15 0.8 70
 1 112 tTCcool 2 2 1 15 1 15
 1 113 DHW_GEN_EFF 2 8 0.45 0.7 0.75 0.7 0.7 1 1.5 0.45 0.45 0.65 0.7 0.65 0.65 1 1.5 0.45
 1 114 tDHW_TEMP 1 2 60 10
 1 115 tFUEL_TYPE_COOL 1 5 1 2 3 4 10
 1 116 tKGCO2 1 13 0.198 0.245 0.018 0.297 0.35 0.318 0.344 0.206 0.013 0.517 0.529 0.058 1
 1 117 tKGCO2_IR 1 13 0.203 0.232 0.025 0.272 0.361 0.361 0.392 0.289 0.025 0.643 0.643 0.018 1
 1 118 tPRIM_IR 1 13 1.1 1.1 1.1 1.1 1.1 1.1 1.2 1.1 1.1 2.7 2.7 1.05 1
 1 119 tKGCO2_JER 1 13 0.194 0.234 0.025 0.265 0.291 0.317 0.392 0.187 0.025 0.08 0.08 0.018 1
 1 120 tPRIM_JER 1 13 1.1 1.1 1.1 1.1 1.1 1.1 1.2 1.1 1.1 2.7 2.7 1.05 1
 1 121 tVENT_STRGY 1 2 1 2
 1 122 tMIXED_MODE 1 30 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
 1 123 tSERVICE_STRGY 1 30 4 1 1 1 1 1 1 1 1 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 4
 1 124 tSYS_HEAT 1 30 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0
 1 125 tSYS_COOL 1 30 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
 1 126 tSYS_VENT 1 30 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 0 0
 1 127 tSYS_AUX 1 30 0 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0
 1 128 tDUCT_LEAK_CORR 2 30 0 0 0 0 0 0 0 0 0 0 0 0 5 5 5 2 2 5 5 5 5 5 0.5 0.5 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 10 10 10 4 4 10 10 10 10 10 1 1 0 0 0 0 0 0 0
 1 129 tK 1 30 0 0.14 0 0.33 0 0 0.25 0.25 0.25 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 130 tGRAD 1 30 0 1.5 2.3 0 0 2 0.3 0.3 0.3 2.3 2.3 0 0 0 0.1 0.1 0 0 0 0 0 0 0 0.1 0.1 0.1 0.1 0.1 2 0
 1 131 tRadiant_Prop 1 29 0 0.25 0 0.6 0 0 0.6 0.6 0.6 0 0 0 0 0 0 0 0 0 0 0 0 0 0.4 0 0 0 0 0 0 0 0
 1 132 tRad_Tot_Eff 1 29 0 0.56 0 0.55 0 0 -6666 -6666 -6666 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.55 0 0 0 0 0 0 0
 1 133 tFPStype 1 29 0 0 0 0 0 2 0 0 0 0 0 0 0 2 2 1 1 3 2 2 2 2 2 2 4 3 2 0 0 0 0 0 0
 1 134 tPump_type 1 29 0 1 1 1 0 0 0 0 0 0 0 0 0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0 0 0 0 0 0
 1 135 tNot_Pump_power 1 29 0 0.3 0.3 0.3 0 0 0 0 0 0 0 0 0.9 0.9 0.9 0.9 0.9 0.9 0.9 0.9 0.9 0.9 0.9 0.9 0.9 0
0 0 0 0
 1 136 tWind_Shear 1 6 0.14 0.19 0.24 0.29 0.31 0.43
 1 137 tWind_EFF 8 2 0 0 0.2 0.36 0.2 0.35 0.19 0.33 0.16 0.29 0.15 0.26 0.14 0.23 0.14 0.23
 1 138 HVAC_tables 27 32 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0.58 3.26 1 0 0 1 0
0 2 0.95 0.9 1 1 0 0 1 0 0 0.65 0.95 0.9 1 1 0 0 1 0 0 0.95 0.9 1 0 0.58 3.26 1 0 0 1 0 0 2 0.95 0.9 1 1 0 0 1 0 0
0.65 0.95 0.9 1 1 0 0 1 0 0 0.95 0.9 1 0 0.58 3.26 1 0 0 1 0 0 2 0.95 0.9 1 1 0 0 1 0 0 0.65 0.95 0.9 1 1 0 0 1 0
0 0.95 0.9 1 0 0.7 91.14 1 0 0 1 0 0 2 0.95 0.9 1 0.678 0.107 3 0.837 0.816 0.2 0.65 0.95 0.9 1 0.036 0.321 3
0.839 0.804 0.2 0.95 0.9 1 0 0.6 8.1 1 0 0 1 0 0 2 0.95 0.9 1 1 0 0 1 0 0 0.75 0.95 0.9 1 1 0 0 1 0 0 0.95 0.9 1 0
0.7 0 1 0 0 1 0 0 2 0.95 0.9 1 1 0 0 1 0 0 0.75 0.95 0.9 1 1 0 0 1 0 0 0.95 0.9 1 0 0.6 0.33 1 0 0 1 0 0 2 0.95 0.9
1 1 0 0 1 0 0 0.75 0.95 0.9 1 1 0 0 1 0 0 0.95 0.9 1 0 0.65 0.33 1 0 0 1 0 0 2 0.95 0.9 1 1 0 0 1 0 0 0.75 0.95
0.9 1 1 0 0 1 0 0 0.95 0.9 1 0 0.65 8.1 1 0 0 1 0 0 2 0.95 0.9 1 1 0 0 1 0 0 0.75 0.95 0.9 1 1 0 0 1 0 0 0.95 0.9 1
0 0.75 8.1 1 0 0 1 0 0 2 0.95 0.9 1 1 0 0 1 0 0 0.75 0.95 0.9 1 1 0 0 1 0 0 0.95 0.9 1 1.12 0.48 35.91 1.182 -
0.061 3 0.853 0.736 0.2 2 0.95 0.9 1 0.872 0.043 3 0.809 0.954 0.2 0.65 0.95 0.9 1 0.333 0.333 3 0.833 0.836
0.2 0.95 0.9 1 0.99 0.45 35.91 1.182 -0.061 3 0.853 0.736 0.2 2 0.95 0.9 1 0.872 0.043 3 0.809 0.954 0.2 0.65
0.95 0.9 1 0.333 0.333 3 0.833 0.836 0.2 0.95 0.9 1 1.43 0.54 20.35 1.182 -0.061 3 1  0.2 2 0.95 0.9 1 0.872
0.043 3 1 0 0.2 0.65 0.95 0.9 1 0.333 0.333 3 1 0 0.2 0.95 0.9 1.17 0.6 43.88 1.226 -0.078 2.9 0.98 0.098 0.2 2
0.95 0.9 1 0.835 0.057 2.9 0.971 0.147 0.2 0.65 0.95 0.9 1 0.077 0.318 2.9 0.846 0.77 0.2 0.95 0.9 1 1.2 0.63
43.23 1.224 -0.056 4 0.956 0.22 0.2 2 0.95 0.9 1 0.829 0.043 4 0.939 0.305 0.2 0.65 0.95 0.9 1 0.077 0.231 4
0.845 0.774 0.2 0.95 0.9 1 0.73 0.65 119.7 1.675 -0.225 3 0.876 0.621 0.2 2 0.95 0.9 1 0.697 0.101 3 0.737
1.315 0.2 0.65 0.95 0.9 1 0 0.333 3 0.846 0.77 0.2 0.95 0.9 1 0.84 0.65 119.7 1.675 -0.225 3 0.876 0.621 0.2 2
0.95 0.9 1 0.697 0.101 3 0.737 1.315 0.2 0.65 0.95 0.9 1 0 0.333 3 0.846 0.77 0.2 0.95 0.9 1 0.83 1.02 119.7
1.924 -0.308 3 0.84 0.8 0.2 2 0.95 0.9 1 0.645 0.118 3 0.829 0.853 0.2 0.65 0.95 0.9 1 0 0.333 3 0.833 0.833
0.2 0.95 0.9 1 0.48 0.43 119.7 1.304 -0.101 3 0.828 0.859 0.2 2 0.95 0.9 1 0.648 0.117 3 0.833 0.833 0.2 0.65
```

```
0.95 0.9 1 0 0.333 3 0.833 0.833 0.2 0.95 0.9 1 0.48 0.43 119.7 1.304 -0.101 3 0.86 0.7 0.2 2 0.95 0.9 1 0.743
0.086 3 0.767 1.167 0.2 0.65 0.95 0.9 1 0 0.333 3 0.834 0.832 0.2 0.95 0.9 1 1.68 0.56 15.97 1.068 -0.023 3
0.981 0.095 0.2 2 0.95 0.9 1 0.938 0.021 3 0.977 0.113 0.2 0.65 0.95 0.9 1 0.25 0.25 3 0.875 0.625 0.2 0.95
0.9 1 1.53 0.56 16.97 1.064 -0.021 3 0.98 0.1 0.2 2 0.95 0.9 1 0.938 0.021 3 0.977 0.113 0.2 0.65 0.95 0.9 1
0.295 0.235 3 0.882 0.588 0.2 0.95 0.9 1 1.09 0.65 45.45 1.209 -0.07 3 0.939 0.303 0.2 2 0.95 0.9 1 0.826
0.058 3 0.914 0.429 0.2 0.65 0.95 0.9 1 0.078 0.307 3 0.847 0.763 0.2 0.95 0.9 1 1.81 3.16 23.94 1.114 -0.038
3 0.962 0.192 0.2 3 0.95 0.9 1 0.883 0.039 3 0.951 0.247 0.2 3.2 0.95 0.9 1 0 0.333 3 0.836 0.822 0.2 0.95 0.9
1 1.42 2.05 1 1 2 0.95 0.9 1 1 0 0 1 0 0 2.2 0.95 0.9 1 0 0 0 1 1.21 2.17 23.94 1.114 -0.038 3 0.962 0.192
0.2 2 0.95 0.9 1 0.882 0.039 3 0.95 0.248 0.2 2.2 0.95 0.9 1 0 0.333 3 0.833 0.834 0.2 0.95 0.9 1
 1 139 DUCT_leakage 1 3 0.02 0.06 0.15
 1 140 AHU_leakage 1 4 0.001 0.007 0.02 0.06
 1 141 KEY_iSBEM 1 27 23 1 24 25 2 3 26 27 28 29 30 7 31 32 4 14 5 6 16 17 15 11 12 13 10 8 9
 1 142 tPOWER_FACTOR 1 3 0 0.01 0.025
 1 143 tTerrainFactorKR 1 4 0.17 0.19 0.22 0.24
 1 144 tRoughnessZo 1 4 0.01 0.05 0.3 1
 1 145 tDem_Cont_Vent_Coeff 1 4 0 0.85 0.95 0.5
 1 146 t2010_TYP 1 10 1 0.75 0.75 0.7 0.75 1
 1 147 tG_Value_Adjust 15 9 0.87 0.87 0.87 0.87 0.87 0.87 0.87 0.87 0.87 0.87 0.87 0.87 0.87 0.87 0.87
0.87 0.87 1.28 1.27 1.35 1.52 1.68 1.52 1.36 1.29 0.87 1.28 1.27 1.35 1.52 1.68 1.52 1.36 1.29 0.87
1.28111111111111 1.26888888888889 1.34111111111111 1.49555555555556 1.63666666666667
1.49444444444444 1.35 1.28777777777778 0.87 1.28333333333333 1.26666666666667 1.32333333333333
1.44666666666667 1.55 1.44333333333333 1.33 1.28333333333333 0.87 1.28555555555556
1.26444444444444 1.30555555555556 1.39777777777778 1.46333333333333 1.39222222222222 1.31
1.27888888888889 0.87 1.28777777777778 1.26222222222222 1.28777777777778 1.34888888888889
1.37666666666667 1.34111111111111 1.29 1.27444444444444 0.87 1.29 1.26 1.27 1.3 1.29 1.29 1.27 1.27
0.87 1.25888888888889 1.23111111111111 1.23666666666667 1.25333333333333 1.23888888888889
1.24555555555556 1.23444444444444 1.23888888888889 0.87 1.22777777777778 1.20222222222222
1.20333333333333 1.20666666666667 1.18777777777778 1.20111111111111 1.19888888888889
1.20777777777778 0.87 1.19666666666667 1.17333333333333 1.17 1.16 1.13666666666667
1.15666666666667 1.16333333333333 1.17666666666667 0.87 1.16555555555556 1.14444444444444
1.13666666666667 1.11333333333333 1.08555555555556 1.11222222222222 1.12777777777778
1.14555555555556 0.87 1.12111111111111 1.10555555555556 1.09666666666667 1.06777777777778 1.04
1.06666666666667 1.08666666666667 1.10444444444444 0.87 0.89 0.91 0.91 0.89 0.88 0.88 0.9 0.9 0.87
 1 148 tModular_TER_Factor 2 4 1 1.33 1.75 2.35 1 1.33 1.75 1.75
 1 149 tModular_TER_Factor_Scot 2 4 1 1.43 1.87 1.87 1 1.43 1.87 1.87
 1 150 tVRP_Type 4 3 0 0.5 1.5 0 0.4 1.1 0 0.3 0.9 0 0.6 1.8
 1 151 tTSC_absortivity 1 5 0.95 0.85 0.75 0.65 0.45
 1 152 tTSC_use 1 12 1 1 1 1 0.8 0.05 0.05 0.05 0.8 1 1 1
 1 153 tTSC_HeatDirect 1 12 0 0 0 0 -1 -1 -1 -1 -1 0 0 0
 1 154 tTSC_HeatDirect_back 1 12 0.3 0.3 0.2 0.1 0 -1 -1 -1 0 0.1 0.2 0.3
 1 155 tTSC_CoolCtrl 1 2 0.5 0
 1 156 tPSI_106 1 11 0.12 0.16 0.09 0.07 0.3 0.04 0.05 0
 1 157 tPSI_106_MC 1 11 0.28 1.15 0.25 0 1.27 1.27 1.27 0
 1 158 tPSI_factors 2 2 1.5 0.04 1.25 0.02
 1 159 tPSI_106max 1 11 0.18 0.24 0.135 0.11 0.45 0.08 0.09 0.04
 1 160 tPSI_106_MCmax 1 11 0.42 1.725 0.375 0.04 1.905 1.905 1.905 0.04
 1 161 tNIGHT_COOL_MONTH 1 12 0 0 0 0 1 1 1 1 1 0 0 0
 2 2 Not_Buildings 1 11 "EW_NOT_2002" "EW_REF_2002" "EW_TYP_2002" "S6_NOT_2010"
"RoI_REF_2002" "RoI_NOT_2002" "RoI_TYP_2002" "JER_NOT_2002" "EW_NOT_2010_Not_Lit"
"EW_NOT_2010_Side_Lit" "EW_NOT_2010_Top_Lit"
 1 162 Not_Buildings_CASE 1 11 1 2 3 4 5 6 7 8 9 10 11
 1 163 Not_U_Values 10 11 0.25 0.25 0.45 0.16 0.16 0.16 0.45 0.25 0.18 0.18 0.18 0.25 0.25 0.45 0.25 0.16
0.16 0.45 0.25 0.18 0.18 0.18 0.35 0.35 0.45 0.3 0.27 0.27 0.45 0.35 0.26 0.26 0.26 0.25 0.25 0.45 0.25 0.25
0.25 0.45 0.25 0.22 0.22 0.22 2.2 2.2 3.3 2.2 2.2 2.2 3.3 2.2 2.2 2.2 2.2 6 6 6 2.2 6 6 6 6 2.2 2.2 2.2 1.5 1.5 1.5
1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.5 6 6 6 6 6 6 6 6 6 6 6 2 2 2 2 2 2 2 2 1.8 1.8 1.8 1.24 1.24 1.24 1.24 1.24 1.24
1.24 1.24 1 1 1
 1 164 Not_U_Value_Factor 1 11 1.1 1.1 1.1 1 1.16 1.16 1.16 1.1 1 1 1
 1 165 Not_CM_Values 10 11 12 12 12 12 12 12 12 12 21.8 21.8 21.8 12 12 12 12 12 12 12 12 21.8 21.8
21.8 11.7 11.7 11.7 11.7 11.7 11.7 11.7 11.7 88.3 88.3 88.3 36 36 36 36 36 36 36 36 77.7 77.7 77.7 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 11.9 11.9 11.9 11.9 11.9 11.9 11.9
11.9 8.8 8.8 8.8 8.6 8.6 8.6 8.6 8.6 8.6 8.6 8.6 69.2 69.2 69.2
```

```
 1 166 Not_CM_Values_MC 10 11 12 12 12 12 12 12 12 12 1.4 1.4 1.4 12 12 12 12 12 12 12 12 1.4 1.4 1.4
11.7 11.7 11.7 11.7 11.7 11.7 11.7 11.7 1.4 1.4 1.4 36 36 36 36 36 36 36 36 77.7 77.7 77.7 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 11.9 11.9 11.9 11.9 11.9 11.9 11.9 11.9 8.8
8.8 8.8 8.6 8.6 8.6 8.6 8.6 8.6 8.6 8.6 69.2 69.2 69.2
 1 167 NotGlass_U_Value 2 11 2.2 2.2 3.3 2.2 2.2 2.2 3.3 2.2 1.8 1.8 1.8 5.7 5.7 5.7 5.7 5.7 5.7 5.7 5.7 0 0 0
 1 168 NotGlass_G_Value 2 11 0.72 0.72 0.72 0.72 0.72 0.72 0.72 0.72 0.4 0.4 0.43 0.77 0.77 0.77 0.72 0.77
0.77 0.77 0.72 0 0 0
 1 169 NotGlass_L_Value 2 11 0.76 0.76 0.76 0.76 0.76 0.76 0.76 0.76 0.71 0.71 0.67 0.87 0.87 0.87 0 0.87
0.87 0.87 0 0 0 0
 1 170 Not_Frame_Factor 3 11 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 1 0.1 1 0.3 0.3 0.3 0.3 0.3 0.3 0.3 0.3 1 1 0.15 1
1 1 1 1 1 1 1 1 1 1
 1 171 Not_Rooflight_Cover 1 11 4 4 4 4 1 1 1 4 1 1 4
 1 172 Not_Psi 8 11 0 0 0 0.06 0 0 0 0 0.12 0.12 0.12 0 0 0 0.16 0 0 0 0 0.16 0.16 0.16 0 0 0 0.09 0 0 0 0 0.09
0.09 0.09 0 0 0 0.07 0 0 0 0 0.07 0.07 0.07 0 0 0 0.3 0 0 0 0 0.3 0.3 0.3 0 0 0 0.04 0 0 0 0 0.04 0.04 0.04 0 0 0
0.05 0 0 0 0 0.05 0.05 0.05 0 0 0 0 0 0 0 0 0 0 0
 1 173 Not_Psi_MC 8 11 0 0 0 0 0 0 0 0 0.28 0.28 0.28 0 0 0 0 0 0 0 0 1.15 1.15 1.15 0 0 0 0 0 0 0 0 0.25
0.25 0.25 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1.27 1.27 1.27 0 0 0 0 0 0 0 0 1.27 1.27 1.27 0 0 0 0 0 0 0 0
1.27 1.27 1.27 0 0 0 0 0 0 0 0 0 0 0
 1 174 Not_Win_Door_Area 9 11 30 30 30 30 30 30 30 30 0 0 0 40 40 40 40 40 40 40 40 0 0 0 15 15 15 15
15 15 15 15 0 0 0 20 20 20 0 20 20 20 20 0 0 0 20 20 20 0 20 20 20 20 0 0 0 20 20 20 10 20 20 20 20 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 175 Not_Glazing_Percent 3 11 0 0 0 0 0 0 0 0 0 40 0 0 0 0 0 0 0 0 0 0 0 12 0 0 0 0 0 0 0 0 0 0 0
 1 176 Not_Inf 1 11 10 10 15 10 10 10 15 10 5 5 5
 1 177 DHW_NOT_EFF 1 11 0.45 0.45 0.35 0.45 0.45 0.45 0.35 0.45 0 0 0
 1 178 Not_Water_Heat_SEff_1 1 11 0 0 0 0 0 0 0 0 0.665 0.665 0.665
 1 179 Not_Water_Heat_SEff_2 1 11 0 0 0 0 0 0 0 0 0.836 0.836 0.855
 1 180 Not_Water_Heat_SEff_3 1 11 0 0 0 0 0 0 0 0 1.33 1.33 1.33
 1 181 Not_Water_Heat_SEff_4 1 11 0 0 0 0 0 0 0 0 2.565 2.565 2.565
 1 182 tNOT_SFP 5 11 2 2 2 2 2 2 2 2 1.8 1.8 1.8 1.2 1.2 1.2 1.2 1.2 1.2 1.2 1.2 0.9 0.9 0.9 0.8 0.8 0.8 0.8 0.8
0.8 0.8 0.8 0.6 0.6 0.6 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.4 0.4 0.4 0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.8 0.5 0.5 0.5
 1 183 Not_HVAC_Eff 15 11 1000000 1000000 1000000 1000000 1000000 1000000 1000000 1000000 0 0
0 1.67 1.67 1.17 1.67 1.67 1.67 1.17 1.67 0 0 0 1000000 1000000 1000000 1000000 1000000 1000000
1000000 1000000 0 0 0 2.25 2.25 1.57 2.25 2.25 2.25 1.57 2.25 0 0 0 2.25 2.25 1.57 2.25 2.25 2.25 1.57 2.25
0 0 0 0.73 0.73 0.55 0.73 0.73 0.73 0.55 0.73 0 0 0 0.83 0.83 0.55 0.83 0.83 0.83 0.55 0.83 0 0 0 0.78 0.78
0.55 0.78 0.78 0.78 0.55 0.78 0 0 0 0.73 0.73 0.55 0.73 0.73 0.73 0.55 0.73 0 0 0 0.78 0.73 0.55 0.78 0.78 0.73
0.55 0.73 0 0 0 0.61 0.61 1.23 0.61 0.61 0.61 1.23 0.61 0 0 0 8.5 8.5 14 8.5 8.5 8.5 14 8.5 0 0 0 0.61 0.61 1.23
0.61 0.61 0.61 1.23 0.61 0 0 0 0.61 0.61 1.23 0.61 0.61 0.61 1.23 0.61 0 0 0 0.61 0.61 1.23 0.61 0.61 0.61 1.23
0.61 0 0 0
 1 184 Not_TEMP_GRAD 2 11 0 0 0 0 0 0 0 0 1.5 1.5 0.3 0 0 0 0 0 0 0 0 0 0 0
 1 185 Not_Space_Heat_SEff_1 1 11 0 0 0 0 0 0 0 0 0.63 0.63 0.63
 1 186 Not_Space_Heat_SEff_2 1 11 0 0 0 0 0 0 0 0 0.792 0.792 0.81
 1 187 Not_Space_Heat_SEff_3 1 11 0 0 0 0 0 0 0 0 1.26 1.26 1.26
 1 188 Not_Space_Heat_SEff_4 1 11 0 0 0 0 0 0 0 0 2.43 2.43 2.43
 1 189 Not_Space_Heat_SEff_5 1 11 0 0 0 0 0 0 0 0 0 0 0.86
 1 190 Not_Radiant_Heat_Seff 2 11 0 0 0 0 0 0 0 0 0 0 0.65 0 0 0 0 0 0 0 0 0 0 0.86
 1 191 Not_Space_Cool_SEER 2 11 0 0 0 0 0 0 0 0 3.6 3.6 3.6 0 0 0 0 0 0 0 0 2.7 2.7 2.7
 2 3 Not_Heat_Recovery_Sys 1 11 "No heat recovery" "No heat recovery" "No heat recovery" "No heat
recovery" "No heat recovery" "No heat recovery" "No heat recovery" "No heat recovery" "Plate heat
exchanger (Recuperator)" "Plate heat exchanger (Recuperator)" "Plate heat exchanger (Recuperator)"
 1 192 Not_Heat_Recovery_Seff 1 11 0 0 0 0 0 0 0 0 0.7 0.7 0.7
 2 4 Not_Var_Heat_Recovery 1 11 "NO" "NO" "NO" "NO" "NO" "NO" "NO" "NO" "YES" "YES"
"YES"
 1 193 Not_HVAC_Metering 1 11 1 1 1 1 1 1 1 1 0.95 0.95 0.95
 2 5 Not_Elec_Power_Fac 1 11 "<0.9" "<0.9" "<0.9" "<0.9" "<0.9" "<0.9" "<0.9" "<0.9" ">0.95"
">0.95" ">0.95"
 1 194 Not_Light_Metering 1 11 1 1 1 1 1 1 1 1 0.95 0.95 0.95
 2 6 Not_Eff_Disp_Light 1 11 "NO" "NO" "NO" "NO" "NO" "NO" "NO" "NO" "YES" "YES" "YES"
 1 195 Not_Disp_Light_Lum_Eff 1 11 15 15 15 15 15 15 15 15 22 22 22
 1 196 tNOT_GROUP 30 11 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 197 tIF 6 11 0.15 0.15 0 0.371 0.15 0.15 0 0.15 0 0 0 0.2 0.15 0 0.408 0.2 0.15 0 0.15 0 0 0 0.2 0.15 0
0.408 0.2 0.15 0 0.15 0 0 0 0.15 0.15 0 0.371 0.15 0.15 0 0.15 0 0 0 0.15 0.15 0 0.371 0.15 0.15 0 0.15 0 0 0
0.2 0.15 0 0.408 0.2 0.15 0 0.15 0 0 0

 1 198 tRB 6 11 0.1 0.1 0 0.15 0.1 0.1 0 0.1 0 0 0 0.1 0.1 0 0.15 0.1 0.1 0 0.1 0 0 0 0.1 0.1 0 0.15 0.1 0.1 0 0.1
0 0 0 0.1 0.1 0 0.15 0.1 0.1 0 0.1 0 0 0 0.1 0.1 0 0.15 0.1 0.1 0 0.1 0 0 0 0.1 0.1 0 0.15 0.1 0.1 0 0.1 0 0 0
 1 199 NOT_LIGHT_FACTORS 18 11 5.2 0 0 5.2 5.2 0 0 0 0 0 0 3.75 0 0 3.75 3.75 0 0 0 0 0 0 0 5.2 0 0 0
5.2 0 5.2 0 0 0 0 3.75 0 0 0 3.75 0 3.75 0 0 0 0 0 6.2 0 0 0 6.2 0 0 0 0 0 0 4.5 0 0 0 4.5 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
 1 200 NOT_PDC_0 3 11 0.069 0.069 0.069 0.069 0.069 0.069 0.069 0.069 0.069 0.069 0.069 0.008 0.008
0.008 0.008 0.008 0.008 0.008 0.008 0.008 0.008 0.008 2.11 2.11 2.11 2.11 2.11 2.11 2.11 2.11 2.11 2.11 2.11
 1 201 NOT_PDC_1 3 11 0.069 0.069 0.069 0.069 0.069 0.069 0.069 0.069 0.069 0.069 0.069 0.008 0.008
0.008 0.008 0.008 0.008 0.008 0.008 0.008 0.008 0.008 2.11 2.11 2.11 2.11 2.11 2.11 2.11 2.11 2.11 2.11 2.11
 1 202 NOT_PDC_2 3 11 0.069 0.069 0.069 0.069 0.069 0.069 0.069 0.069 0.069 0.069 0.069 0.008 0.008
0.008 0.008 0.008 0.008 0.008 0.008 0.008 0.008 0.008 2.11 2.11 2.11 2.11 2.11 2.11 2.11 2.11 2.11 2.11 2.11
 1 203 Not_Buildings_CASE_ 1 11 1 2 3 4 5 6 7 8 9 10 11
 2 7 Not_Buildings_END 1 11 "EW_NOT_2002" "EW_REF_2002" "EW_TYP_2002" "S6_NOT_2010"
"RoI_REF_2002" "RoI_NOT_2002" "RoI_TYP_2002" "JER_NOT_2002" "EW_NOT_2010_Not_Lit"
"EW_NOT_2010_Side_Lit" "EW_NOT_2010_Top_Lit"
 1 204 T003_BEL 1 12 4.73 4.06 6.24 6.84 10.92 12.53 14.6 14.7 12.76 9.93 6.68 5.74
 1 205 T003_BIR 1 12 4.93 5.5 6.32 7.47 11.61 14.23 17.11 16.28 13.15 9.89 6.61 5.19
 1 206 T003_CAR 1 12 6.2 4.91 6.6 8.48 12.19 13.97 16.06 16.12 13.86 11.17 8.56 6.75
 1 207 T003_DUB 1 12 5.78 4.82 6.81 8.51 10.55 13.52 15.65 14.61 13.21 10.51 7.19 6.24
 1 208 T003_EDI 1 12 4.04 5.08 6.31 7.38 10.09 12.77 14.81 14.31 11.75 8.98 5.76 4.1
 1 209 T003_GLA 1 12 2.98 2.94 5.88 6.78 10.43 13.12 14.48 14.18 12.26 9.12 6.05 5.37
 1 210 T003_JER 1 12 6.54 6.27 7.38 8.82 12 14.23 16.36 17.21 15.02 12.94 9.35 7.34
 1 211 T003_LEE 1 12 4.34 5.44 6.72 8.73 11.62 14.23 17.43 15.64 13.55 10.37 7.23 6.4
 1 212 T003_LON 1 12 6.22 6.29 7.13 9.63 13.48 16.01 18.51 17.89 15.06 11.84 8.67 6.2
 1 213 T003_MAN 1 12 5.4 5.83 6.58 7.99 11.1 13.99 15.74 16.41 13.53 10.57 6.57 5.98
 1 214 T003_NEW 1 12 4.62 5.25 6.95 7.24 10.18 13.36 15.39 14.84 13.11 11.45 7.01 5.41
 1 215 T003_NOR 1 12 4.87 4.61 6.54 8.72 11.24 13.84 16.47 16.61 14.3 10.29 7.67 5.81
 1 216 T003_NOT 1 12 3.98 4.74 6.2 7.01 11.74 14.29 16.93 15.52 12.95 10.11 6 5.62
 1 217 T003_PLY 1 12 7.45 6.87 7.03 9.41 12.1 13.98 16.58 15.62 14.27 11.78 9.53 8.27
 1 218 T003_SOU 1 12 5.07 6.7 7.09 9.25 12.54 15.76 16.86 17.19 14.83 11.68 7.98 6.25
 1 219 T003_SWI 1 12 4.55 5.05 5.49 7.27 11.87 14.33 16.57 16 13.37 10.34 6.43 5.85
 1 220 T025nc_BEL 1 12 4.45 3.64 5.37 5.18 8.72 10.96 12.71 13.07 11.67 9.4 6.68 5.44
 1 221 T025nc_BIR 1 12 4.21 4.89 4.63 5.8 9.24 12.08 14.57 13.8 11.48 8.89 6 4.89
 1 222 T025nc_CAR 1 12 5.98 4.08 5.46 6.75 10.77 12 14.19 14.48 12.11 10.36 7.84 6.27
 1 223 T025nc_DUB 1 12 5.42 4.18 5.71 6.83 8.37 11.55 13.66 12.57 11.78 9.24 6.48 5.65
 1 224 T025nc_EDI 1 12 3.54 4.15 5.1 5.66 8.15 10.62 12.39 12.38 10.2 8.01 4.91 3.73
 1 225 T025nc_GLA 1 12 2.91 2.52 4.96 4.61 8.02 10.94 12.24 12.44 10.68 8.08 5.15 5.32
 1 226 T025nc_JER 1 12 6.3 5.72 6.52 7.3 10 12.43 14.72 15.3 14.07 12.47 8.71 7.1
 1 227 T025nc_LEE 1 12 3.65 4.81 5.57 7.38 9.34 12.24 15.38 14.2 11.94 9.65 6.78 6.17
 1 228 T025nc_LON 1 12 5.73 5.38 5.53 7.75 11.58 13.38 16.37 15.71 13.47 10.64 7.93 5.56
 1 229 T025nc_MAN 1 12 5.05 5.14 5.11 6.42 9.35 12.11 13.49 14.64 12.02 9.35 6.08 5.67
 1 230 T025nc_NEW 1 12 4.2 4.51 6.2 6.14 8.48 11.92 13.73 13.61 12.22 10.82 6.59 5.21
 1 231 T025nc_NOR 1 12 4.24 3.79 5.2 6.94 9.26 11.18 13.89 14.01 12.11 9.13 6.77 5.6
 1 232 T025nc_NOT 1 12 3.35 4.24 4.77 5.43 9.31 11.96 14.47 13.45 11.37 8.77 5.54 5.15
 1 233 T025nc_PLY 1 12 7.16 6.34 6.36 8.17 10.84 12.3 14.64 14.09 13.31 11.17 8.98 7.54
 1 234 T025nc_SOU 1 12 4.64 5.95 6.05 7.57 10.71 13.65 15.04 15.51 13.58 10.78 7.51 5.44
 1 235 T025nc_SWI 1 12 4.02 4.21 4.22 5.53 9.73 12.66 13.42 13.3 11.41 9.03 5.71 5.09
 1 236 T010_BEL 12 9 140.726 90.764 35.783 25.0243 25.0243 25.3478 58.9724 123.235 63.2952 155.289
109.45 66.7671 48.1917 47.7209 53.502 95.2126 144.368 112.97 217.276 183.657 141.126 103.621 96.5419
119.317 170.87 210.025 228.887 234.431 228.852 204.699 166.981 150.947 197.34 243.618 253.532 337.504
306.49 308.783 290.649 243.493 233.797 317.657 375.891 355.169 516.462 275.562 287.128 285.197
252.764 246.824 312.776 350.208 319.054 494.993 267.195 272.739 266.774 240.528 230.418 277.686
```

310.547 297.485 459.272 248.842 241.477 223.932 193.549 178.917 221.172 265.141 272.132 382.997 240.076 198.848 161.316 129.671 121.4 163.956 230.648 262.613 282.801 184.714 134.81 89.9378 66.6029 65.538 75.5799 125.351 175.914 153.264 125.327 86.5833 44.4594 32.8345 32.8345 34.5076 63.3056 111.563 74.6532 89.21 58.4361 24.9379 20.1038 20.1038 20.1038 39.834 79.5024 43.9164
1 237 T010_BIR 12 9 144.168 94.6912 46.0149 34.2605 34.2605 35.6231 72.3876 130.625 80.5049 155.46 118.838 79.1464 55.8658 54.7474 62.4648 100.135 141.922 127.266 197.187 160.581 126.818 102.978 99.3128 126.37 176.309 207.181 223.255 226.919 209.48 186.559 158.503 149.767 198.479 248.153 256.611 331.771 273.803 296.379 290.268 245.767 221.354 276.33 314.364 299.893 467.699 278.313 283.134 275.921 243.582 239.048 311.75 355.657 327.728 511.521 282.229 304.885 301.766 259.66 233.548 296.96 339.238 320.579 507.015 277.08 265.65 240.405 204.331 194.903 253.278 305.085 308.174 429.553 236.88 206.608 168.788 135.548 126.669 159.183 211.93 243.985 289.57 191.121 147.473 103.889 78.2745 75.8797 90.1279 138.032 183.905 173.564 125.449 90.6066 52.5831 39.069 39.069 40.9481 68.7356 111.571 87.5078 112.372 75.4182 33.7803 25.011 25.011 25.011 49.8546 98.1507 57.33
1 238 T010_CAR 12 9 143.165 96.3225 48.3111 35.9096 35.9096 36.9476 71.598 128.217 84.5422 201.474 138.811 82.4582 56.4451 55.436 66.6653 126.468 190.829 141.715 226.911 189.86 146.709 110.345 102.442 132.345 189.84 228.856 249.21 273.688 249.296 215.2 170.882 154.196 224.102 299.063 314.676 387.699 301.78 303.428 286.673 237.361 219.616 317.567 388.832 367.696 527.067 265.447 269.446 266.233 239.134 247.389 333.793 375.292 329.02 499.085 305.392 303.838 294.11 258.426 251.29 344.677 405.967 375.777 555.958 299.071 299.59 273.492 222.443 201.704 274.976 336.048 335.525 471.851 272.041 228.552 178.722 136.576 125.455 175.725 250.6 291.053 324.096 204.95 149.287 102.911 79.9366 78.2741 92.4049 149.145 202.202 184.578 145.033 99.169 55.7752 43.515 43.515 45.7629 81.7655 133.677 99.4859 133.508 88.3575 39.3746 29.268 29.268 29.268 60.0496 117.596 69.318
1 239 T010_DUB 12 9 107.65 74.8333 39.287 32.4112 32.4112 32.6893 54.9251 96.6708 68.8241 147.369 111.68 71.32 56.3449 55.9624 59.0596 88.3107 132.994 125.314 217.868 187.655 146.322 115.222 110.154 124.912 165.304 204.809 248.749 251.303 215.653 191.216 172.778 167.319 206.145 261.754 282.042 366.91 308.557 303.75 282.437 242.526 223.282 272.18 323.066 331.554 509.391 302.017 289.713 273.66 247.75 239.076 293.51 341.971 340.559 532.228 295.964 285.621 272.213 247.187 239.155 294.549 341.648 336.455 513.398 285.188 258.973 227.681 194.477 184.922 230.32 287.426 307.623 429.099 220.064 182.392 153.297 134.562 130.921 147.577 193.205 225.815 282.536 199.528 147.835 97.9094 80.1106 79.6518 85.445 129.552 187.251 181.334 119.389 82.565 49.66 42.3032 42.3 43.8638 72.0176 112.623 87.9703 86.9227 63.2965 32.0154 26.2717 26.2717 26.3041 40.3513 75.0528 54.3396
1 240 T010_EDI 12 9 120.314 80.7172 32.6605 24.8317 24.8317 24.8639 49.1544 104.011 58.0032 132.617 94.6722 60.6938 46.9634 46.6776 51.2581 85.2367 125.086 102.831 191.788 162.123 130.468 103.496 97.9934 121.344 166.593 195.363 214.766 217.803 206.47 184.241 155.441 146.272 180.228 218.66 230.36 311.333 267.945 276.908 267.964 232.845 218.796 274.709 314.003 300.153 451.471 265.805 286.263 288.033 255.213 240.778 289.184 317.191 293.528 472.615 282.001 279.727 272.068 246.686 252.152 326.274 367.743 335.445 489.02 242.879 246.462 223.987 184.844 166.332 200.505 235.54 247.14 359.972 228.386 205.969 168.049 130.538 120.226 149.637 196.544 227.167 272.697 173.406 134.351 89.7521 65.4126 64.426 71.4264 109.743 156.609 144.955 145.868 96.7558 42.6905 31.2624 31.2624 31.2624 64.6742 127.845 75.888 99.6844 68.735 23.9382 18.6232 18.6232 18.6232 34.1306 83.1492 42.7356
1 241 T010_GLA 12 9 122.849 74.8715 33.5827 26.6983 26.6983 27.6057 62.8897 115.41 60.0801 180.783 117.292 68.4255 49.7184 49.4035 59.7991 118.497 178.024 119.304 199.37 160.702 125.74 100.025 95.9296 127.414 182.104 213.024 214.941 245.286 244.173 213.972 168.028 154.229 201.442 244.601 254.074 343.575 288.761 290.099 271.14 229.172 225.088 296.854 344.365 325.975 481.267 279.147 275.555 266.83 239.832 242.587 307.761 348.937 323.742 490.054 281.241 278.137 265.465 237.005 244.297 314.321 353.581 325.436 481.996 242.359 231.419 210.744 180.467 173.628 232.95 281.64 279.198 366.24 265.167 214.892 161.968 122.745 114.522 162.571 238.62 283.469 287.924 181.453 132.732 88.8874 65.7903 65.1992 76.8326 126.088 174.3 148.226 161.403 98.0592 44.6215 34.0175 34.0175 36.0663 84.3358 152.175 81.4027 112.456 68.6594 25.3108 19.6762 19.6762 19.6762 48.8176 101.903 44.5896
1 242 T010_JER 12 9 109.046 80.3501 50.389 45.2693 45.2693 45.79 65.2903 100.903 93.2855 198.706 137.029 84.3218 66.3266 65.92 72.61 126.005 189.695 163.809 292.835 227.3 164.22 126.287 122.17 146.099 220.049 286.443 320.411 338.499 282.512 224.305 175.706 163.904 222.606 314.18 362.714 475.273 349.23 325.344 288.801 238.717 218.089 292.441 373.411 391.275 604.629 332.374 307.328 282.099 246.293 233.087 307.915 382.217 387.294 624.942 335.101 311.75 287.557 246.812 232.066 325.547 410.103 406.322 621.788 350.824 318.971 272.888 215.958 195.72 262.419 348.016 378.757 546.405 303.613 246.346 187.578 148.334 142.057 174.673 247.773 305.136 370.548 201.242 151.882 107.969 92.234 91.7449 99.3112 143.12 194.515 206.995 138.704 99.2362 60.2333 52.196 52.196 52.9122 81.2297 128.213 114.218 103.658 73.7147 40.6646 35.9878 35.9878 35.9935 54.9894 93.9674 75.9985
1 243 T010_LEE 12 9 112.756 75.0268 36.3389 28.413 28.413 28.6079 55.0354 101.273 63.288 128.481 100.576 66.3387 49.1097 48.4204 53.0956 79.2526 114.853 107.855 217.828 169.73 131.246 103.017 96.2539 131.886 199.3 237.105 232.092 225.85 214.42 188.719 157.751 145.868 186.536 229.267 242.979

318.43 281.491 294.295 283.971 241.036 222.221 300.183 351.568 330.745 485.422 261.867 286.81 290.326 258.616 241.894 289.852 314.6 289.903 473.906 276.33 295.378 292.905 257.562 243.574 300.979 333.582 309.487 485.763 265.685 268.084 246.443 203.388 185.137 238.672 285.396 287.888 407.927 239.651 213.947 173.024 135.835 126.048 158.895 208.773 241.443 287.441 173.568 135.065 94.0359 68.5606 66.5258 81.8582 125.832 166.734 155.175 119.979 82.3965 45.961 35.6425 35.6425 36.1115 64.8404 108.627 78.4283 93.9942 65.0268 28.7473 23.2502 23.2502 23.2502 40.4104 81.5208 50.6124
1 244 T010_LON 12 9 124.634 85.0607 44.5308 33.3814 33.3814 33.9207 63.4609 111.293 76.3947 130.749 103.97 72.6378 56.4737 55.7694 60.5902 86.2694 119.131 120.414 212.519 178.586 142.58 111.851 104.874 129.287 180.506 214.785 240.939 239.868 237.775 211.909 170.434 151.883 192.892 236.42 249.981 346.863 268.466 274.701 263.779 231.282 220.408 279.031 317.96 303.576 463.842 288.062 321.391 325.643 283.469 256.618 322.257 358.312 328.804 542.135 288.054 297.728 289.262 251.147 235.748 307.159 354.907 334.551 519.746 282.06 277.676 256.469 210.542 186.448 253.057 317.264 321.139 443.917 259.637 235.656 187.612 138.504 125.993 159.084 212.968 250.934 307.562 200.345 153.445 108.312 80.5497 77.701 92.0807 144.173 192.63 181.016 128.516 94.5346 52.9515 36.9259 36.9259 38.2499 64.9425 110.169 85.7841 117.725 81.319 35.8858 24.9833 24.9833 24.9833 48.9569 99.8043 59.4828
1 245 T010_MAN 12 9 123.67 86.2742 39.6848 29.471 29.471 29.6077 53.7455 106.022 67.9824 158.459 116.994 72.3217 51.6593 50.7683 56.9429 95.0863 143.905 120.894 212.58 176.767 142.551 112.123 106.365 136.478 191.788 222.045 237.714 239.489 231.013 205.574 168.356 153.919 198.547 245.229 256.904 343.591 271.665 282.959 269.48 226.194 207.359 275.305 322.962 309.483 464.17 263.669 268.55 265.192 241.645 244.639 310.381 343.38 310.389 477.364 283.201 286.103 277.881 246.64 246.959 328.344 373.86 340.134 502.195 281.99 266.313 242.102 204.742 194.385 266.072 329.367 328.395 435.681 246.76 212.793 170.154 130.775 120.166 159.772 221.764 256.784 290.469 208.365 149.144 99.8155 73.0721 70.7782 85.9193 147.989 204.424 173.229 112.604 79.5149 44.6604 35.8373 35.8373 36.1469 60.0954 101.034 77.6249 108.02 68.8136 30.8986 23.8781 23.8781 23.8781 51.492 97.9371 53.5968
1 246 T010_NEW 12 9 121.506 78.3291 34.4364 27.0965 27.0965 27.4432 56.8822 109.726 62.5608 200.55 139.231 77.672 52.6253 52.0171 58.1082 111.48 181.559 130.86 205.96 168.254 131.516 104.326 99.8996 121.566 171.548 207.629 222.438 231.768 216.004 189.676 160.001 149.236 182.901 225.393 243.616 329.273 264.546 276.176 265.905 226.386 209.203 278.09 323.954 306.567 450.929 252.582 262.063 260.749 237.206 241.246 301.147 327.359 292.322 453.265 286.1 283.501 271.116 242.129 241.654 308.771 351.299 330.255 494.716 270.634 262.799 235.272 195.438 181.584 227.245 274.04 285.819 405.16 211.997 179.004 146.334 117.246 111.381 141.798 194.333 222.333 250.936 198.545 146.826 102.56 75.2146 73.0703 88.2015 145.127 194.037 169.756 125.343 86.9809 45.4873 32.8925 32.8925 33.002 61.5996 109.658 74.6208 85.0477 59.3264 23.2563 19.5012 19.5012 19.5012 32.4814 72.3727 42.894
1 247 T010_NOR 12 9 129.004 94.5426 46.368 30.6202 30.6202 30.6202 54.3517 105.833 72.6876 195.919 151.164 90.9682 56.0934 54.6361 59.9764 103.293 164.71 141.019 232.034 202.844 159.326 116.293 105.252 129.08 181.789 221.825 251.706 289.54 281.925 243.252 184.992 155.727 206.546 270.462 298.852 398.682 308.196 324.929 309.955 253.863 221.608 303.341 366.012 354.729 524.707 252.508 288.146 295.908 260.111 228.57 266.56 289.318 272.378 463.712 304.885 328.091 322.4 271.808 235.463 300.831 353.271 342.726 541.516 308.889 302.797 271.731 216.352 187.528 256.682 327.874 341.867 467.669 260.625 221.126 177.019 135.158 118.771 154.607 224.8 269.25 298.072 225.79 173.832 117.691 81.0342 77.458 91.1614 149.11 208.137 189.724 147.941 105.572 56.3864 38.5621 38.5621 38.7837 71.3118 126.459 92.4912 123.034 93.4705 40.5695 24.395 24.395 24.395 41.5208 94.8158 58.8024
1 248 T010_NOT 12 9 123.309 85.4704 40.8896 29.8444 29.8444 30.3038 56.337 106.857 68.7811 218.333 158.211 88.3622 53.1851 52.0459 60.3236 116.216 190.464 142.415 239.284 195.447 150.459 111.942 104.01 137.61 203.939 245.412 253.737 227.945 219.822 194.472 158.106 144.373 197.413 246.353 253.886 331.301 278.337 295.558 288.117 240.402 213.426 295.652 353.16 332.293 484.793 278.641 297.716 294.967 256.123 240.872 303.406 340.783 315.226 510.59 323.413 338.169 322.897 265.592 237.659 326.456 392.086 375.153 576.94 293.852 292.736 265.367 215.645 193.404 256.006 313.872 320.971 448.427 243.359 224.822 183.264 136.115 124.163 162.238 213.953 242.1 292.05 232.807 172.827 110.51 72.1311 68.3006 86.7602 153.299 218.711 182.321 121.987 82.9169 45.0187 34.4264 34.4264 35.279 65.2094 110.618 76.8871 136.176 82.0579 31.3778 25.033 25.033 25.033 61.8749 125.187 60.2496
1 249 T010_PLY 12 9 171.452 116.344 55.2042 36.0122 36.0122 37.272 78.8193 148.481 92.5697 166.072 129.305 82.356 55.4561 54.0209 59.9542 96.3301 144.569 132.665 235.72 188.894 144.794 109.453 103.286 129.436 193.141 237.284 250.754 252.509 242.719 211.633 162.102 142.476 190.477 247.785 265.47 363.653 291.679 290.663 273.167 226.729 207.063 299.061 369.66 354.792 512.529 253.168 268.766 269.983 232.775 234.662 333.791 380.448 323.971 498.141 284.552 293.581 290.436 255.162 239.809 322.907 376.652 347.764 524.932 295.491 302.354 279.29 221.838 192.36 276.146 345.357 341.479 471.964 261.669 223.854 181.785 139.98 127.018 179.008 253.015 285.561 322.285 219.451 158.688 106.078 78.2295 76.8006 92.8454 155.612 214.124 187.043 132.78 96.899 54.9766 42.214 42.214 44.0302 70.7749 117.425 95.9762 128.077 89.2161 41.4656 30.5244 30.5244 30.5244 56.0154 109.793 72.072

```
 1 250 T010_SOU 12 9 145.365 100.713 51.4844 36.7106 36.7106 37.9547 71.3861 127.614 87.6557
165.243 125.618 80.7996 57.4348 56.5099 62.6892 100.538 148.278 133.475 205.196 172.469 136.76
104.724 98.2071 119.328 168.596 202.889 228.219 273.523 255.954 221.442 174.166 152.167 214.695
284.448 304.529 388.928 266.487 276.527 266.412 226.064 213.265 302.264 356.034 327.07 474.349 281.67
290.132 286.253 254.334 237.913 303.384 348.751 329.468 526.712 266.12 274.321 270.682 241.377
232.697 304.115 346.239 318.437 482.951 276.267 264.664 243.069 204.617 190.748 263.889 327.492
324.784 437.731 266.833 223.461 181.918 141.067 127.801 180.821 260.648 295.048 324.547 218.3 169.349
116.074 82.6706 80.1986 94.4536 148.257 203.08 196.19 149.552 100.973 55.6036 41.6088 41.6088 44.787
84.0875 138.078 97.9977 142.316 93.3493 41.2227 29.2644 29.2644 29.2644 63.6449 125.059 71.064
 1 251 T010_SWI 12 9 127.517 91.1465 47.3548 34.457 34.457 34.6083 60.4578 109.526 79.0884 187.568
147.989 90.6438 58.0997 56.6251 59.8152 96.8906 155.107 143.107 245.151 205.514 158.786 116.536
106.003 129.113 189.887 236.92 265.402 235.99 220.438 194.349 157.864 139.943 188.585 245.349 261.843
340.452 274.11 286.912 281.375 241.894 212.985 276.884 328.317 318.309 479.221 248.625 262.836
263.873 237.044 221.213 274.162 307.303 287.136 461.329 314.099 319.187 311.51 270.235 250.07 341.237
406.722 382.836 574.081 325.189 317.166 286.379 228.888 195.942 268.015 346.116 362.519 502.578
267.986 242.636 198.388 146.926 127.185 166.523 231.547 269.934 323.381 205.188 165.483 116.184
82.2219 78.9786 89.3267 134.693 184.554 187.596 124.257 94.5356 55.145 39.4481 39.4481 39.6834
62.3813 104.534 88.5528 123.346 82.9235 39.5174 29.5265 29.5265 29.5265 57.822 108.81 67.8348
 1 252 TSOL_BEL 1 3 221.618 304.924 233.775
 1 253 TSOL_BIR 1 3 226.435 312.701 239.738
 1 254 TSOL_CAR 1 3 234.282 340.879 261.341
 1 255 TSOL_DUB 1 3 216.658 324.586 248.849
 1 256 TSOL_EDI 1 3 217.252 290.514 222.727
 1 257 TSOL_GLA 1 3 215.051 302.092 231.604
 1 258 TSOL_JER 1 3 238.737 399.772 306.492
 1 259 TSOL_LEE 1 3 228.242 303.058 232.345
 1 260 TSOL_LON 1 3 237.414 323.416 247.952
 1 261 TSOL_MAN 1 3 221.28 309.785 237.502
 1 262 TSOL_NEW 1 3 211.792 293.862 225.294
 1 263 TSOL_NOR 1 3 250.655 332.08 254.594
 1 264 TSOL_NOT 1 3 239.643 325.886 249.846
 1 265 TSOL_PLY 1 3 233.024 331.974 254.514
 1 266 TSOL_SOU 1 3 227.374 324.791 249.006
 1 267 TSOL_SWI 1 3 237.6 330.438 253.336
 1 268 tF_shade_BEL 12 9 0.805895 0.546104 0 0 0 0 0.336516 0.718531 0 0.589806 0.429453 0.0329663 0
0 0.0206056 0.36787 0.551589 0.15602 0.516551 0.426255 0.235649 0 0 0.106177 0.371687 0.526005
0.409815 0.407202 0.353107 0.272703 0.0133381 0 0.200229 0.40807 0.47532 0.567705 0.458515 0.456412
0.337576 0.117326 0.0824365 0.361961 0.528562 0.540249 0.743272 0.355136 0.327221 0.275332 0.143034
0.0791816 0.313466 0.46402 0.407877 0.731574 0.294261 0.234106 0.164171 0.064569 0.013905 0.174778
0.314784 0.329829 0.693087 0.315861 0.25599 0.173527 0.0245986 0 0.130714 0.34319 0.342731 0.604636
0.451101 0.302234 0.130434 0 0 0.173503 0.443146 0.547216 0.460577 0.54345 0.411874 0.0822713 0 0
0.0370334 0.372651 0.514052 0.222016 0.608681 0.377551 0 0 0 0 0.282262 0.502044 0 0.547124 0.281227
0 0 0 0 0.172108 0.434649 0
 1 269 tF_shade_BIR 12 9 0.614833 0.370869 0 0 0 0 0.356391 0.567719 0 0.48758 0.443638 0.0875432 0 0
0.0378237 0.349813 0.443537 0.194757 0.430292 0.312141 0.137784 0 0 0.128396 0.388094 0.487017
0.335224 0.430338 0.346023 0.209929 0 0.0154815 0.194542 0.461693 0.480211 0.608994 0.311326
0.397562 0.349321 0.092893 0.0402917 0.228182 0.375427 0.348964 0.722319 0.372927 0.342855 0.279204
0.0843196 0.0674699 0.339807 0.501028 0.471053 0.747932 0.379196 0.413876 0.333421 0.1326 0.0392381
0.302297 0.477997 0.467116 0.750226 0.410547 0.37032 0.214796 0.033661 0.0231865 0.21511 0.440904
0.478844 0.704825 0.422044 0.341784 0.180755 0 0 0.139569 0.340626 0.487077 0.484808 0.490103
0.408571 0.104327 0 0 0.043024 0.337223 0.477362 0.252592 0.541104 0.388095 0 0 0 0 0.245429 0.469732
0.0248892 0.622193 0.35755 0 0 0 0 0.171042 0.497912 0
 1 270 tF_shade_CAR 12 9 0.661037 0.370896 0 0 0 0 0.262145 0.596028 0 0.66025 0.519006 0.0567112 0
0 0.0530718 0.476264 0.632905 0.245013 0.563429 0.438674 0.269282 0 0 0.146357 0.421529 0.551486
0.470091 0.557962 0.473042 0.313285 0.0197237 0 0.32595 0.569221 0.623522 0.704866 0.557762 0.515327
0.397338 0.146756 0.0637705 0.44249 0.654759 0.648391 0.762467 0.371202 0.341708 0.231153 0.105263
0.0897885 0.42336 0.549324 0.494212 0.750087 0.441455 0.393722 0.262075 0.0915074 0.0624578
0.382605 0.586257 0.585198 0.807542 0.45918 0.438454 0.318708 0.0524826 0 0.287676 0.491823 0.527395
0.751234 0.630524 0.497722 0.253696 0 0 0.232041 0.518205 0.647283 0.638967 0.578716 0.404525
0.0725553 0 0 0.0262148 0.409872 0.573097 0.393745 0.545899 0.318625 0 0 0 0 0.335512 0.483063
0.045124 0.635792 0.383169 0 0 0 0 0.291377 0.583186 0
```

1 271 tF_shade_DUB 12 9 0.588693 0.317291 0 0 0 0 0.209028 0.552692 0 0.553554 0.412152 0.0620932 0 0 0 0.296446 0.510594 0.237981 0.508277 0.415308 0.187558 0 0 0.0392104 0.321508 0.476698 0.436097 0.397606 0.235486 0.0852188 0 0 0.171001 0.418468 0.499663 0.670432 0.419392 0.365868 0.254456 0.0483071 0 0.197686 0.419286 0.4528 0.755717 0.381492 0.301894 0.213617 0.0387179 0.00467851 0.226874 0.412648 0.453893 0.731529 0.328503 0.265314 0.188793 0.044977 0 0.219293 0.395745 0.421283 0.741026 0.45676 0.357847 0.217486 0 0 0.206423 0.423599 0.489114 0.702745 0.442731 0.270992 0.107346 0 0 0.01498 0.332898 0.466511 0.521329 0.622476 0.445457 0.0113398 0 0 0 0.33234 0.574725 0.372519 0.487619 0.299732 0 0 0 0 0.235996 0.512927 0 0.559345 0.302509 0 0 0 0 0 0.474816 0
1 272 tF_shade_EDI 12 9 0.710576 0.456423 0 0 0 0 0.16094 0.558884 0 0.442344 0.242586 0 0 0 0 0.296123 0.423967 0.0557343 0.351476 0.284612 0.130459 0 0 0.117666 0.300296 0.380085 0.30335 0.384157 0.305873 0.152356 0 0.00739741 0.145142 0.330713 0.37554 0.515174 0.313617 0.329109 0.268532 0.096325 0.0332664 0.220739 0.410281 0.370339 0.685615 0.244779 0.284485 0.277225 0.139661 0.0501113 0.225462 0.347656 0.293135 0.671547 0.267131 0.215839 0.140811 0.0559593 0.0690062 0.312766 0.462386 0.392826 0.721818 0.374047 0.367394 0.257415 0.0255661 0 0.117107 0.267351 0.348483 0.564374 0.407301 0.371538 0.225958 0 0 0.0751324 0.266732 0.394554 0.377166 0.468426 0.385411 0.106583 0 0 0 0.247461 0.385567 0.139177 0.693335 0.516351 0 0 0 0 0.357652 0.615001 0 0.664173 0.432 0 0 0 0 0.0325343 0.546629 0
1 273 tF_shade_GLA 12 9 0.630052 0.339348 0 0 0 0 0.368892 0.562788 0 0.64961 0.446761 0.0479453 0 0 0.0682812 0.516797 0.640629 0.138624 0.463839 0.306387 0.110554 0 0 0.182568 0.42637 0.500377 0.340051 0.437729 0.431113 0.311835 0 0.00713855 0.221791 0.381754 0.427959 0.588218 0.419722 0.412939 0.314278 0.0675158 0.0698371 0.300989 0.490256 0.457444 0.710789 0.364781 0.304518 0.203723 0.0647042 0.055905 0.262146 0.46015 0.427667 0.726533 0.350635 0.272611 0.160881 0.0263633 0.0627177 0.293701 0.454616 0.431636 0.710424 0.360824 0.289304 0.16974 0 0 0.281132 0.445708 0.454232 0.610046 0.589964 0.468199 0.201531 0 0 0.196348 0.545577 0.654453 0.540768 0.503103 0.381294 0.0661703 0 0 0.0418748 0.380969 0.533927 0.174771 0.718375 0.459029 0 0 0 0 0.486635 0.668202 0 0.70084 0.405331 0 0 0 0 0.366687 0.62665 0
1 274 tF_shade_JER 12 9 0.463649 0.267961 0 0 0 0 0.156598 0.427262 0.0505158 0.67101 0.479874 0.0129236 0 0 0 0.445877 0.63944 0.414043 0.591468 0.480678 0.231089 0 0 0.0919606 0.449833 0.606452 0.669617 0.68302 0.515949 0.333108 0 0 0.279992 0.593911 0.677179 0.79737 0.622394 0.531921 0.324184 0.0720956 0 0.33791 0.579241 0.677974 0.834391 0.578573 0.493663 0.273669 0.0794019 0 0.357053 0.592094 0.655512 0.829661 0.547416 0.471637 0.293371 0.0692867 0 0.418378 0.608861 0.675027 0.820877 0.621835 0.526484 0.386828 0 0 0.305274 0.581276 0.621889 0.830522 0.621079 0.498258 0.261121 0 0 0.0928977 0.512322 0.642272 0.760304 0.531578 0.402252 0.0226843 0 0 0 0.335944 0.532124 0.444151 0.586843 0.318304 0 0 0 0 0.171563 0.55324 0.131149 0.492034 0.198185 0 0 0 0 0 0.519339 0
1 275 tF_shade_LEE 12 9 0.598814 0.375032 0 0 0 0 0.218447 0.539848 0 0.449155 0.418179 0.108345 0 0 0.0204195 0.234203 0.374148 0.151637 0.526282 0.405449 0.164177 0 0 0.185739 0.501012 0.594196 0.42291 0.370265 0.341843 0.20332 0.00696909 0 0.177697 0.36811 0.422272 0.578829 0.429456 0.427438 0.332541 0.12047 0.049374 0.369304 0.545796 0.503054 0.721584 0.215012 0.277233 0.286226 0.137614 0.041143 0.221001 0.320895 0.283738 0.707334 0.271176 0.328903 0.265499 0.122237 0.0480381 0.233827 0.389661 0.32361 0.721959 0.406564 0.398278 0.285555 0.056253 0.017904 0.208629 0.375302 0.405175 0.676613 0.439495 0.367004 0.16602 0 0 0.116791 0.28587 0.452965 0.434331 0.547657 0.495249 0.23539 0 0 0.0859543 0.357596 0.54829 0.284079 0.500446 0.341069 0 0 0 0 0.227758 0.43324 0 0.573981 0.313773 0 0 0 0 0.12215 0.465368 0
1 276 tF_shade_LON 12 9 0.602343 0.397005 0 0 0 0 0.272847 0.520534 0 0.38287 0.354352 0.11415 0 0 0 0.180664 0.344729 0.175076 0.474025 0.369301 0.206981 0 0 0.106754 0.33483 0.495731 0.396624 0.439722 0.41933 0.304285 0.0136344 0 0.163322 0.349543 0.429019 0.601624 0.373704 0.349125 0.280249 0.099623 0.0243871 0.251417 0.44745 0.410573 0.705638 0.325956 0.378497 0.36137 0.179686 0.0393743 0.322831 0.458142 0.367176 0.772214 0.382824 0.391209 0.306369 0.111133 0.0617404 0.326562 0.500869 0.468328 0.758558 0.465147 0.432215 0.329174 0.0735238 0.00618494 0.273165 0.505561 0.531497 0.736063 0.572641 0.511559 0.318322 0 0 0.132668 0.395488 0.552972 0.608447 0.513829 0.436761 0.141658 0 0 0.0275335 0.344481 0.524328 0.281312 0.601251 0.523611 0 0 0 0 0.273064 0.534022 0.0397417 0.725518 0.602675 0 0 0 0 0.303964 0.557361 0
1 277 tF_shade_MAN 12 9 0.627431 0.44975 0 0 0 0 0.117568 0.500248 0 0.551771 0.44598 0.0497637 0 0 0.043758 0.340037 0.517061 0.190967 0.374878 0.304679 0.151941 0 0 0.145814 0.350001 0.461089 0.2908 0.428208 0.386432 0.273273 0.00688375 0.0079717 0.204705 0.420915 0.458464 0.624945 0.419236 0.424124 0.343482 0.123519 0.0538863 0.33668 0.501215 0.486419 0.693414 0.267352 0.253906 0.17648 0.0914158 0.0696739 0.312892 0.438223 0.387446 0.722354 0.348882 0.298345 0.206406 0.0722685 0.0736142 0.349067 0.502762 0.443795 0.726272 0.418779 0.34838 0.230129 0.0218776 0.00557495 0.284338 0.497274 0.518145 0.701943 0.500578 0.443886 0.235508 0 0 0.178507 0.407707 0.558221 0.496519 0.617636 0.450423 0.160512 0 0 0.039519 0.457395 0.653899 0.303476 0.43847 0.262856 0 0 0 0

0.193908 0.382131 0 0.584209 0.358763 0 0 0 0 0.230039 0.476829 0
1 278 tF_shade_NEW 12 9 0.61011 0.333893 0 0 0 0 0.230233 0.552862 0 0.679979 0.519265 0.0295665 0 0 0 0.420538 0.587857 0.11868 0.368815 0.322787 0.130963 0 0 0.0710257 0.327024 0.457493 0.296448 0.427776 0.313085 0.200855 0.0072013 0 0.120955 0.361069 0.453488 0.559286 0.415027 0.392604 0.326559 0.103529 0.0409204 0.324132 0.503013 0.491409 0.696826 0.217535 0.210813 0.160663 0.0710291 0.0729137 0.293979 0.379192 0.315443 0.675602 0.312352 0.261396 0.166237 0.0375099 0.0486531 0.273071 0.441299 0.42181 0.724431 0.454325 0.40147 0.240231 0.0233047 0.011991 0.164316 0.371198 0.419398 0.660175 0.439031 0.317591 0.156098 0 0 0.0991046 0.353224 0.466407 0.383146 0.456457 0.369194 0.160396 0 0 0.0412734 0.373505 0.481882 0.199388 0.50087 0.346669 0 0 0 0 0.280843 0.417528 0 0.636028 0.392405 0 0 0 0 0.0691411 0.514266 0
1 279 tF_shade_NOR 12 9 0.667748 0.504236 0 0 0 0 0.223831 0.515049 0 0.659641 0.616248 0.217202 0 0 0.0194047 0.340023 0.532454 0.29539 0.516368 0.487393 0.276533 0 0 0.112518 0.348889 0.474979 0.421705 0.594375 0.556305 0.431503 0.0745643 0 0.216088 0.468877 0.594725 0.706786 0.543531 0.545687 0.448253 0.198395 0.0147027 0.354239 0.571066 0.584446 0.766706 0.285468 0.373859 0.380984 0.202005 0.0299068 0.217802 0.315655 0.275077 0.681847 0.418045 0.459459 0.414442 0.203293 0.00929975 0.258131 0.455304 0.456739 0.766442 0.556891 0.521776 0.400866 0.095848 0 0.30493 0.527573 0.574263 0.775811 0.572612 0.480162 0.279437 0.00804814 0 0.155116 0.457672 0.634826 0.542792 0.615653 0.477764 0.230285 0 0 0.0321795 0.428088 0.593997 0.342327 0.613889 0.467325 0.0603138 0 0 0 0.262918 0.572404 0.023665 0.75802 0.577636 0 0 0 0 0.147036 0.503701 0
1 280 tF_shade_NOT 12 9 0.656527 0.456631 0 0 0 0 0.240642 0.518099 0 0.736972 0.66634 0.175977 0 0 0.0426863 0.505754 0.652351 0.242518 0.524391 0.435034 0.228607 0 0 0.161833 0.432685 0.53628 0.424659 0.478888 0.436991 0.313988 0.028682 0.00756895 0.28804 0.486348 0.563988 0.60613 0.482496 0.485832 0.403227 0.190488 0.0483134 0.388804 0.596818 0.57194 0.725787 0.391836 0.407883 0.363944 0.141075 0.055462 0.303199 0.475653 0.440474 0.751354 0.524418 0.543483 0.438878 0.191485 0.0500159 0.403232 0.619927 0.621315 0.801156 0.484255 0.472656 0.326356 0.0498113 0 0.258939 0.492713 0.52247 0.729654 0.475615 0.451277 0.276841 0 0 0.161129 0.356878 0.440052 0.450798 0.73077 0.654698 0.327743 0 0 0.0711126 0.466692 0.689572 0.413093 0.62369 0.426287 0 0 0 0 0.287348 0.536462 0 0.699667 0.489164 0 0 0 0 0.376098 0.604455 0
1 281 tF_shade_PLY 12 9 0.751377 0.549584 0 0 0 0 0.294275 0.6599 0.0235282 0.629212 0.536054 0.161535 0 0 0.0204572 0.377543 0.56032 0.298333 0.578047 0.444552 0.265176 0 0 0.165339 0.478826 0.632948 0.467268 0.569561 0.522778 0.413399 0.0419518 0 0.259792 0.49159 0.54387 0.688069 0.537316 0.534962 0.414307 0.149115 0.0549499 0.440015 0.64297 0.642261 0.763284 0.42866 0.447041 0.407107 0.239766 0.156333 0.534998 0.621597 0.580989 0.726618 0.39123 0.364218 0.276389 0.120109 0.0445862 0.355447 0.556882 0.544863 0.775019 0.561661 0.505627 0.407508 0.0901835 0 0.356647 0.58857 0.642339 0.766805 0.569822 0.470738 0.283331 0 0 0.228994 0.504523 0.624055 0.595117 0.64978 0.489265 0.166571 0 0 0.0517091 0.458645 0.626819 0.380724 0.566518 0.407399 0 0 0 0 0.220786 0.486848 0.0715303 0.657376 0.454956 0 0 0 0 0.245027 0.545094 0
1 282 tF_shade_SOU 12 9 0.645353 0.495 0 0 0 0 0.298273 0.523307 0.0125673 0.546423 0.486471 0.0850455 0 0 0.019985 0.331466 0.472271 0.225803 0.526151 0.432737 0.220807 0 0 0.0615368 0.404281 0.513025 0.401251 0.616765 0.533291 0.36883 0.0255869 0 0.273096 0.563264 0.620753 0.69965 0.463756 0.467385 0.384563 0.144192 0.0505428 0.419668 0.598852 0.552917 0.698599 0.419999 0.382026 0.286174 0.132363 0.0394289 0.293933 0.493116 0.489568 0.755839 0.335684 0.324496 0.235622 0.0940195 0.0459182 0.324242 0.499394 0.438067 0.715154 0.458044 0.388401 0.272263 0.0630791 0.018348 0.305785 0.54555 0.55544 0.722062 0.604703 0.437085 0.272014 0 0 0.230366 0.569346 0.645279 0.604957 0.605079 0.49331 0.188544 0 0 0.0504879 0.35965 0.573889 0.39068 0.61552 0.466317 0 0 0 0 0.330037 0.592309 0.0708262 0.745347 0.505255 0 0 0 0 0.338779 0.631701 0
1 283 tF_shade_SWI 12 9 0.637169 0.442418 0 0 0 0 0.179121 0.547626 0 0.667742 0.571466 0.23558 0 0 0 0.296826 0.551827 0.350624 0.575047 0.508604 0.322894 0 0 0.0853383 0.423201 0.570969 0.496861 0.516035 0.466045 0.334541 0.0755263 0 0.266356 0.479419 0.58032 0.640224 0.437883 0.422686 0.363045 0.196424 0.0105197 0.307681 0.518149 0.500525 0.737096 0.310549 0.312687 0.269577 0.12936 0.0300424 0.295225 0.443542 0.396684 0.708577 0.442389 0.409472 0.319875 0.155395 0 0.369958 0.590072 0.541035 0.779679 0.578311 0.540221 0.406283 0.115905 0 0.301776 0.544606 0.607888 0.789212 0.531835 0.485104 0.322484 0 0 0.18991 0.440062 0.597358 0.597564 0.530253 0.481586 0.212621 0 0 0 0.32706 0.506683 0.320303 0.556922 0.450729 0.0823557 0 0 0 0.226459 0.464899 0.0643548 0.610483 0.338667 0 0 0 0 0.23179 0.559658 0
2 8 T033_tables 16 12 "T033_BEL_JAN" "T033_BEL_FEB" "T033_BEL_MAR" "T033_BEL_APR" "T033_BEL_MAY" "T033_BEL_JUN" "T033_BEL_JUL" "T033_BEL_AUG" "T033_BEL_SEP" "T033_BEL_OCT" "T033_BEL_NOV" "T033_BEL_DEC" "T033_BIR_JAN" "T033_BIR_FEB" "T033_BIR_MAR" "T033_BIR_APR" "T033_BIR_MAY" "T033_BIR_JUN" "T033_BIR_JUL" "T033_BIR_AUG" "T033_BIR_SEP" "T033_BIR_OCT" "T033_BIR_NOV" "T033_BIR_DEC" "T033_CAR_JAN" "T033_CAR_FEB" "T033_CAR_MAR" "T033_CAR_APR" "T033_CAR_MAY"

"T033_CAR_JUN" "T033_CAR_JUL" "T033_CAR_AUG" "T033_CAR_SEP" "T033_CAR_OCT" "T033_CAR_NOV" "T033_CAR_DEC" "T033_DUB_JAN" "T033_DUB_FEB" "T033_DUB_MAR" "T033_DUB_APR" "T033_DUB_MAY" "T033_DUB_JUN" "T033_DUB_JUL" "T033_DUB_AUG" "T033_DUB_SEP" "T033_DUB_OCT" "T033_DUB_NOV" "T033_DUB_DEC" "T033_EDI_JAN" "T033_EDI_FEB" "T033_EDI_MAR" "T033_EDI_APR" "T033_EDI_MAY" "T033_EDI_JUN" "T033_EDI_JUL" "T033_EDI_AUG" "T033_EDI_SEP" "T033_EDI_OCT" "T033_EDI_NOV" "T033_EDI_DEC" "T033_GLA_JAN" "T033_GLA_FEB" "T033_GLA_MAR" "T033_GLA_APR" "T033_GLA_MAY" "T033_GLA_JUN" "T033_GLA_JUL" "T033_GLA_AUG" "T033_GLA_SEP" "T033_GLA_OCT" "T033_GLA_NOV" "T033_GLA_DEC" "T033_JER_JAN" "T033_JER_FEB" "T033_JER_MAR" "T033_JER_APR" "T033_JER_MAY" "T033_JER_JUN" "T033_JER_JUL" "T033_JER_AUG" "T033_JER_SEP" "T033_JER_OCT" "T033_JER_NOV" "T033_JER_DEC" "T033_LEE_JAN" "T033_LEE_FEB" "T033_LEE_MAR" "T033_LEE_APR" "T033_LEE_MAY" "T033_LEE_JUN" "T033_LEE_JUL" "T033_LEE_AUG" "T033_LEE_SEP" "T033_LEE_OCT" "T033_LEE_NOV" "T033_LEE_DEC" "T033_LON_JAN" "T033_LON_FEB" "T033_LON_MAR" "T033_LON_APR" "T033_LON_MAY" "T033_LON_JUN" "T033_LON_JUL" "T033_LON_AUG" "T033_LON_SEP" "T033_LON_OCT" "T033_LON_NOV" "T033_LON_DEC" "T033_MAN_JAN" "T033_MAN_FEB" "T033_MAN_MAR" "T033_MAN_APR" "T033_MAN_MAY" "T033_MAN_JUN" "T033_MAN_JUL" "T033_MAN_AUG" "T033_MAN_SEP" "T033_MAN_OCT" "T033_MAN_NOV" "T033_MAN_DEC" "T033_NEW_JAN" "T033_NEW_FEB" "T033_NEW_MAR" "T033_NEW_APR" "T033_NEW_MAY" "T033_NEW_JUN" "T033_NEW_JUL" "T033_NEW_AUG" "T033_NEW_SEP" "T033_NEW_OCT" "T033_NEW_NOV" "T033_NEW_DEC" "T033_NOR_JAN" "T033_NOR_FEB" "T033_NOR_MAR" "T033_NOR_APR" "T033_NOR_MAY" "T033_NOR_JUN" "T033_NOR_JUL" "T033_NOR_AUG" "T033_NOR_SEP" "T033_NOR_OCT" "T033_NOR_NOV" "T033_NOR_DEC" "T033_NOT_JAN" "T033_NOT_FEB" "T033_NOT_MAR" "T033_NOT_APR" "T033_NOT_MAY" "T033_NOT_JUN" "T033_NOT_JUL" "T033_NOT_AUG" "T033_NOT_SEP" "T033_NOT_OCT" "T033_NOT_NOV" "T033_NOT_DEC" "T033_PLY_JAN" "T033_PLY_FEB" "T033_PLY_MAR" "T033_PLY_APR" "T033_PLY_MAY" "T033_PLY_JUN" "T033_PLY_JUL" "T033_PLY_AUG" "T033_PLY_SEP" "T033_PLY_OCT" "T033_PLY_NOV" "T033_PLY_DEC" "T033_SOU_JAN" "T033_SOU_FEB" "T033_SOU_MAR" "T033_SOU_APR" "T033_SOU_MAY" "T033_SOU_JUN" "T033_SOU_JUL" "T033_SOU_AUG" "T033_SOU_SEP" "T033_SOU_OCT" "T033_SOU_NOV" "T033_SOU_DEC" "T033_SWI_JAN" "T033_SWI_FEB" "T033_SWI_MAR" "T033_SWI_APR" "T033_SWI_MAY" "T033_SWI_JUN" "T033_SWI_JUL" "T033_SWI_AUG" "T033_SWI_SEP" "T033_SWI_OCT" "T033_SWI_NOV" "T033_SWI_DEC"

1 284 T033_BEL_JAN 8 7 63.2952 37.6099 35.733 33.7679 31.207 28.2247 25.0243 63.2952 42.178 36.0412 33.8088 31.207 28.2247 25.0243 63.2952 57.365 53.7822 50.5132 46.0732 41.1381 35.783 63.2952 78.9221 90.876 98.3422 100.812 98.2559 90.764 63.2952 91.937 116.019 133.9 144.361 146.689 140.726 63.2952 87.41 107.273 121.532 129.213 129.794 123.235 63.2952 67.993 70.1464 71.0023 68.8179 65.1996 58.9724 63.2952 45.8409 37.6071 34.5331 31.8219 28.7105 25.3478

1 285 T033_BEL_FEB 8 7 112.97 83.5732 69.4768 65.4845 60.2816 54.2228 47.7209 112.97 89.6382 75.7133 68.0713 61.6983 54.8573 48.1917 112.97 105.782 100.094 93.8626 86.5104 76.8419 66.7671 112.97 125.325 133.287 135.493 131.716 122.92 109.45 112.97 138.588 158.012 169.921 173.502 168.511 155.289 112.97 135.639 152.317 161.866 163.772 157.822 144.368 112.97 118.15 120.219 119.223 114.604 106.492 95.2126 112.97 96.5696 83.6187 75.5523 68.3506 61.1214 53.502

1 286 T033_BEL_MAR 8 7 228.887 193.129 153.08 132.575 121.936 109.68 96.5419 228.887 199.472 173.516 153.734 136.894 120.465 103.621 228.887 220.301 211.042 199.015 182.989 163.174 141.126 228.887 242.188 247.766 244.85 231.835 211.564 183.657 228.887 255.626 271.524 275.496 267.274 247.416 217.276 228.887 251.917 264.481 266.503 257.343 238.521 210.025 228.887 232.162 230.049 223.159 211.096 193.071 170.87 228.887 207.849 185.394 168.343 152.867 136.586 119.317

1 287 T033_BEL_APR 8 7 337.504 309.388 270.169 223.101 194.865 173.277 150.947 337.504 309.728 278.705 249.329 221.757 194.438 166.981 337.504 325.476 311.421 292.731 267.836 237.973 204.699 337.504 341.679 338.335 324.955 300.717 268.405 228.852 337.504 352.602 354.459 342.448 317.082 280.093 234.431 337.504 353.396 356.622 347.349 325.736 294.135 253.532 337.504 341.303 336.328 323.99 304.139 277.074 243.618 337.504 322.839 298.783 274.185 250.481 225.009 197.34

1 288 T033_BEL_MAY 8 7 516.462 486.835 437.602 372.118 304.181 266.713 233.797 516.462 476.778 422.536 368.619 323.421 282.869 243.493 516.462 490.269 460.552 426.005 385.489 339.782 290.649 516.462 510.413 495.141 465.885 422.708 369.242 308.783 516.462 526.279 516.05 487.723 440.661 378.194 306.49 516.462 534.603 532.991 512.219 474.094 420.549 355.169 516.462 527.888 522.67 504.724 473.59 429.904 375.891 516.462 507.946 478.99 440.395 401.614 361.42 317.657

1 289 T033_BEL_JUN 8 7 494.993 477.826 442.372 391.047 328.576 285.305 246.824 494.993 468.107 424.494 376.84 334.635 293.9 252.764 494.993 472.767 445.829 414.067 376.186 332.742 285.197 494.993 483.977 465.224 434.473 392.522 342.469 287.128 494.993 494.025 477.082 444.431 397.811 339.491

275.562 494.993 502.421 493.177 468.55 429.882 378.885 319.054 494.993 502.09 494.543 475.302 444.093 401.624 350.208 494.993 491.902 469.571 435.21 398.407 357.799 312.776
1 290 T033_BEL_JUL 8 7 459.272 441.249 407.443 360.16 305.839 267.068 230.418 459.272 436.039 398.882 357.533 318.382 279.678 240.528 459.272 442.3 418.725 389.291 353.186 311.72 266.774 459.272 452.524 436.165 408.645 370.461 324.225 272.739 459.272 460.975 447.109 418.705 377.131 324.537 267.195 459.272 465.59 456.976 434.224 398.316 351.898 297.485 459.272 462.267 452.078 431.465 399.733 358.552 310.547 459.272 452.048 428.543 395.719 359.281 320.045 277.686
1 291 T033_BEL_AUG 8 7 382.997 357.747 319.913 272.096 233.716 206.326 178.917 382.997 356.098 321.358 288.081 256.785 225.281 193.549 382.997 368.212 348.507 324.33 295.18 261.282 223.932 382.997 383.236 373.738 353.444 322.792 285.195 241.477 382.997 393.943 390.286 372.706 341.586 298.985 248.842 382.997 396.184 395.574 381.719 354.895 317.785 272.132 382.997 386.583 379.53 363.002 337.805 304.614 265.141 382.997 370.589 346.114 317.101 287.295 255.206 221.172
1 292 T033_BEL_SEP 8 7 282.801 246.585 201.859 169.25 154.461 138.366 121.4 282.801 246.052 213.687 189.911 169.864 149.922 129.671 282.801 264.944 249.094 231.506 210.933 186.854 161.316 282.801 288.571 287.325 277.667 258.628 232.121 198.848 282.801 308.017 320.515 319.444 304.877 277.806 240.076 282.801 311.709 327.69 330.818 320.119 297.017 262.613 282.801 295.282 299.164 295.112 281.859 260.143 230.648 282.801 268.292 248.086 228.454 208.803 187.444 163.956
1 293 T033_BEL_OCT 8 7 153.264 119.756 95.8599 90.2022 82.9782 74.5657 65.538 153.264 125.92 107.782 96.036 86.2526 76.471 66.6029 153.264 144.129 136.945 128.463 117.488 104.762 89.9378 153.264 165.97 172.69 172.848 166.759 153.786 134.81 153.264 181.12 201.099 211.84 212.611 203.359 184.714 153.264 178.567 196.167 204.916 204.17 194.078 175.914 153.264 159.458 161.142 158.912 151.974 140.782 125.351 153.264 135.224 118.461 107.301 97.4454 86.9775 75.5799
1 294 T033_BEL_NOV 8 7 74.6532 52.0167 48.3397 45.4945 41.7865 37.4684 32.8345 74.6532 57.134 49.5184 45.6168 41.7865 37.4684 32.8345 74.6532 70.1259 66.832 63.002 57.8407 51.7962 44.4594 74.6532 86.7065 95.2297 99.8433 99.8903 95.3676 86.5833 74.6532 97.1671 115.297 127.807 133.845 132.999 125.327 74.6532 93.6047 108.415 118.074 121.925 119.704 111.563 74.6532 78.106 79.293 78.6408 76.0083 70.5543 63.3056 74.6532 60.282 52.2884 48.1276 43.9837 39.3113 34.5076
1 295 T033_BEL_DEC 8 7 43.9164 31.0387 29.9077 28.1087 25.7641 23.0338 20.1038 43.9164 32.2847 29.9077 28.1087 25.7641 23.0338 20.1038 43.9164 40.4761 38.1307 35.5132 32.3572 28.9489 24.9379 43.9164 53.0261 59.8923 64.0469 65.2069 63.2932 58.4361 43.9164 60.991 75.2792 85.8073 91.8578 93.0184 89.21 43.9164 58.4785 70.4254 78.943 83.4508 83.6416 79.5024 43.9164 46.9604 48.2809 48.6217 47.0924 44.0795 39.834 43.9164 33.8222 29.9953 28.1087 25.7641 23.0338 20.1038
1 296 T033_BIR_JAN 8 7 80.5049 53.6934 49.9867 47.1009 43.34 38.9604 34.2605 80.5049 59.5307 50.8658 47.157 43.34 38.9604 34.2605 80.5049 74.2079 70.2256 65.8642 59.8744 53.5745 46.0149 80.5049 94.2171 103.843 109.061 109.272 104.37 94.6912 80.5049 107.375 129.263 144.677 152.565 152.392 144.168 80.5049 103.87 122.492 135.1 140.837 139.311 130.625 80.5049 85.7549 88.4106 88.4477 86.0317 80.0875 72.3876 80.5049 64.2303 54.3387 49.5202 44.9757 40.5072 35.6231
1 297 T033_BIR_FEB 8 7 127.266 99.2401 80.1079 75.4542 69.3893 62.3266 54.7474 127.266 106.121 91.1867 81.0535 72.4918 64.0834 55.8658 127.266 122.146 117.484 111.081 102.569 91.5909 79.1464 127.266 139.75 147.397 148.743 144.504 134.384 118.838 127.266 150.861 167.906 177.24 178.226 170.797 155.46 127.266 147.068 160.578 166.876 165.533 157.044 141.922 127.266 130.227 129.96 127.437 121.648 112.131 100.135 127.266 110.351 96.3998 87.486 79.6619 71.1379 62.4648
1 298 T033_BIR_MAR 8 7 223.255 193.7 159.028 138.477 126.876 113.578 99.3128 223.255 195.215 170.609 151.343 134.988 118.894 102.978 223.255 210.649 198.593 184.465 167.656 147.908 126.818 223.255 229.331 229.561 221.979 207.564 186.671 160.581 223.255 244.363 255.586 256.158 246.041 225.924 197.187 223.255 246.001 258.754 261.012 252.545 234.171 207.181 223.255 231.865 233.272 228.426 216.954 199.351 176.309 223.255 210.201 193.132 177.402 161.635 144.734 126.37
1 299 T033_BIR_APR 8 7 331.771 305.441 268.183 222.734 193.392 171.645 149.767 331.771 301.303 267.366 236.755 210.015 184.261 158.503 331.771 313.934 294.761 272.871 247.259 217.898 186.559 331.771 330.861 322.786 305.95 280.343 246.895 209.48 331.771 345.534 346.106 332.98 307.175 271.161 226.919 331.771 350.802 356.465 348.949 329.052 297.391 256.611 331.771 341.443 340.646 330.454 311.076 283.213 248.153 331.771 322.562 302.922 279.374 254.673 227.745 198.479
1 300 T033_BIR_MAY 8 7 467.699 445.045 405.78 352.577 293.694 255.535 221.354 467.699 445.066 409.749 369.687 329.94 288.345 245.767 467.699 457.315 440.512 414.732 380.351 338.483 290.268 467.699 468.369 458.611 435.363 398.959 351.397 296.379 467.699 472.443 461.459 433.159 390.483 336.585 273.803 467.699 472.645 461.925 437.535 400.809 353.692 299.893 467.699 465.121 450.305 427.468 397.597 359.52 314.364 467.699 453.431 422.868 386.423 352.371 316.183 276.33
1 301 T033_BIR_JUN 8 7 511.521 491.028 451.272 394.961 325.935 276.994 239.048 511.521 479.412 429.691 374.934 327.543 284.819 243.582 511.521 484.089 450.311 412.759 370.458 324.382 275.921 511.521 498.039 474.606 439.847 394.434 340.867 283.134 511.521 512.464 494.751 459.675 409.126 347.578 278.313 511.521 522.97 514.892 488.995 447.163 391.969 327.728 511.521 521.828 514.051

491.613 456.607 410.517 355.657 511.521 508.597 485.213 446.624 404.151 359.702 311.75
1 302 T033_BIR_JUL 8 7 507.015 484.243 442.679 385.154 317.238 271.437 233.548 507.015 481.076 438.544 391.39 346.706 303.346 259.66 507.015 490.152 466.681 436.769 398.316 352.503 301.766 507.015 501.999 486.112 457.51 416.119 363.902 304.885 507.015 509.755 493.513 460.008 410.846 350.5 282.229 507.015 512.615 500.855 473.86 433.346 380.999 320.579 507.015 506.672 492.076 467.518 433.819 390.438 339.238 507.015 494.792 463.402 422.24 382.301 340.863 296.96
1 303 T033_BIR_AUG 8 7 429.553 399.881 355.424 299.212 252.181 223.219 194.903 429.553 395.06 352.396 309.938 272.89 238.141 204.331 429.553 409.612 385.92 356.382 321.132 281.58 240.405 429.553 428.28 417.649 395.157 360.63 315.894 265.65 429.553 442.524 439.612 419.587 383.571 335.231 277.08 429.553 448.067 449.563 434.58 404.796 361.582 308.174 429.553 437.997 432.245 415.338 387.715 350.31 305.085 429.553 417.948 391.263 358.308 324.597 289.934 253.278
1 304 T033_BIR_SEP 8 7 289.57 255.481 212.617 177.178 161.353 144.597 126.669 289.57 258.378 228.216 202.231 179.431 157.293 135.548 289.57 277.073 263.323 245.764 223.757 197.754 168.788 289.57 297.546 298.001 288.02 269.103 240.665 206.608 289.57 312.53 322.797 319.671 303.365 275.13 236.88 289.57 312.071 322.367 319.767 305.349 278.998 243.985 289.57 295.311 292.177 282.105 265.174 241.548 211.93 289.57 271.733 247.204 224.33 203.53 181.881 159.183
1 305 T033_BIR_OCT 8 7 173.564 140.93 113.485 104.982 96.4579 86.5317 75.8797 173.564 147.701 128.196 113.939 102.073 90.2809 78.2745 173.564 165.628 158.483 149.062 136.254 121.239 103.889 173.564 185.724 191.986 191.461 183.376 167.965 147.473 173.564 200.062 218.097 226.441 224.524 212.477 191.121 173.564 197.575 213.292 219.804 217.063 204.699 183.905 173.564 179.103 179.983 176.378 167.979 155.333 138.032 173.564 155.555 138.144 125.724 114.644 102.932 90.1279
1 306 T033_BIR_NOV 8 7 87.5078 64.5474 57.7424 54.3158 49.8501 44.6497 39.069 87.5078 70.2215 60.5202 55.0118 50.1713 44.6854 39.069 87.5078 83.2256 79.4192 74.5051 68.6595 61.137 52.5831 87.5078 98.7099 106.287 109.308 107.543 101.112 90.6066 87.5078 108.214 124.208 134.401 138.096 135.043 125.449 87.5078 104.622 117.269 124.587 126.077 121.637 111.571 87.5078 90.0379 89.9626 88.2299 83.7127 77.5307 68.7356 87.5078 73.3414 63.3884 57.8901 52.8166 47.1139 40.9481
1 307 T033_BIR_DEC 8 7 57.33 38.1047 36.7413 34.5888 31.7835 28.5167 25.011 57.33 41.8973 36.8247 34.5888 31.7835 28.5167 25.011 57.33 53.6195 51.024 47.7299 44.1559 39.4469 33.7803 57.33 69.2751 78.2037 83.5073 84.8244 82.0654 75.4182 57.33 78.8394 96.6805 109.637 116.827 117.76 112.372 57.33 75.1587 89.5699 99.5816 104.511 104.023 98.1507 57.33 60.3891 61.7196 61.2994 59.6298 55.6547 49.8546 57.33 43.865 37.6031 34.7424 31.7892 28.5167 25.011
1 308 T033_CAR_JAN 8 7 84.5422 57.2999 52.3651 49.3455 45.4102 40.8275 35.9096 84.5422 63.237 53.767 49.5006 45.4102 40.8275 35.9096 84.5422 78.3464 73.6639 68.8989 62.6454 55.9193 48.3111 84.5422 98.2525 107.714 112.296 111.982 106.559 96.3225 84.5422 110.645 131.655 146.139 153.112 152.098 143.165 84.5422 106.776 124.18 135.569 140.167 137.659 128.217 84.5422 88.9122 90.66 89.8413 86.4885 79.8619 71.598 84.5422 68.0706 56.9857 51.5323 46.8757 42.1217 36.9476
1 309 T033_CAR_FEB 8 7 141.715 101.848 78.8876 74.5842 68.9758 62.4447 55.436 141.715 109.776 90.2231 79.4811 71.5544 63.8946 56.4451 141.715 131.529 124.444 116.421 107.142 95.0731 82.4582 141.715 157.732 168.421 171.425 167.037 156.363 138.811 141.715 176.572 203.175 219.709 225.048 218.829 201.474 141.715 173.556 197.348 211.468 214.955 208.08 190.829 141.715 150.174 154.253 154.286 150.051 139.901 126.468 141.715 120.345 102.742 92.0873 83.9185 75.3152 66.6653
1 310 T033_CAR_MAR 8 7 249.21 211.994 167.839 140.048 128.863 116.085 102.442 249.21 216.572 187.548 164.407 145.49 127.835 110.345 249.21 237.54 226.212 211.629 193.011 170.337 146.709 249.21 260.653 264.288 258.993 243.845 220.726 189.86 249.21 276.424 291.781 294.235 283.619 260.656 226.911 249.21 274.881 288.828 291.201 281.004 260.048 228.856 249.21 255.429 254.449 247.567 233.964 214.622 189.84 249.21 229.369 205.427 185.31 168.197 150.981 132.345
1 311 T033_CAR_APR 8 7 387.699 348.668 296.086 233.716 196.558 175.072 154.196 387.699 344.083 298.174 259.066 227.381 198.651 170.882 387.699 363.664 339.979 314.441 284.694 250.571 215.2 387.699 388.975 381.434 363.851 334.268 294.862 249.296 387.699 410.621 416.305 403.829 374.044 329.559 273.688 387.699 416.811 428.786 423.389 401.505 364.54 314.676 387.699 401.467 402.18 392.201 371.42 339.95 299.063 387.699 373.074 345.608 315.307 285.985 255.957 224.102
1 312 T033_CAR_MAY 8 7 527.067 494.336 440.905 370.414 292.839 251.891 219.616 527.067 482.378 423.682 365.58 317.971 276.221 237.361 527.067 495.675 462.184 424.488 382.249 335.884 286.673 527.067 518.055 499.135 467.142 421.876 365.569 303.428 527.067 536.929 526.896 495.105 443.97 378.78 301.78 527.067 549.394 550.373 530.833 492.58 436.654 367.696 527.067 542.297 539.392 521.92 489.689 444.585 388.832 527.067 519.261 489.805 448.338 406.001 363.109 317.567
1 313 T033_CAR_JUN 8 7 499.085 484.378 450.609 400.078 336.229 286.383 247.389 499.085 465.788 416.367 363.732 318.887 278.262 239.134 499.085 466.804 432.043 396.037 355.712 311.899 266.233 499.085 479.553 453.949 419.098 375.035 323.996 269.446 499.085 496.037 475.827 440.58 390.532 331.385 265.447 499.085 512.316 506.969 484.019 444.758 391.756 329.02 499.085 517.279 517.981 502.511 472.888 429.632 375.292 499.085 505.706 491.812 461.833 424.473 381.799 333.793

```
 1 314 T033_CAR_JUL 8 7 555.958 530.428 483.584 418.616 340.636 290.964 251.29 555.958 514.058
454.525 393.718 343.224 299.643 258.426 555.958 520.02 480.345 439.1 393.398 344.556 294.11 555.958
538.566 512.113 474.113 424.529 366.048 303.838 555.958 559.304 540.96 504.709 449.585 381.802
305.392 555.958 574.874 572.422 549.041 506.424 447.375 375.777 555.958 573.599 571.716 552.702
517.716 467.594 405.967 555.958 555.001 531.363 491.786 446.311 396.766 344.677
 1 315 T033_CAR_AUG 8 7 471.851 437.017 385.292 320.198 262.392 230.605 201.704 471.851 433.78
386.516 340.436 299.091 260.023 222.443 471.851 451.552 428.217 398.744 362.413 319.812 273.492
471.851 472.638 462.966 440.249 403.525 355.3 299.59 471.851 487.432 484.439 462.019 421.004 365.106
299.071 471.851 491.486 492.082 474.871 441.497 394.082 335.525 471.851 478.437 470.51 451.95 422.775
383.722 336.048 471.851 455.847 423.317 385.642 350.012 313.795 274.976
 1 316 T033_CAR_SEP 8 7 324.096 279.373 224.13 172.913 156.88 141.613 125.455 324.096 280.003
237.935 205.159 179.75 157.648 136.576 324.096 304.102 284.821 263.373 238.26 209.431 178.722 324.096
333.394 333.333 322.088 300.443 268.082 228.552 324.096 355.251 370.717 369.438 351.502 318.132
272.041 324.096 357.294 374.986 376.465 362.272 332.817 291.053 324.096 336.028 336.716 328.491
311.108 284.87 250.6 324.096 303.64 274.645 247.188 223.129 199.646 175.725
 1 317 T033_CAR_OCT 8 7 184.578 148.478 116.811 107.513 98.9492 88.9763 78.2741 184.578 153.48
130.838 115.785 103.685 91.8457 79.9366 184.578 172.604 162.205 150.406 136.422 120.837 102.911
184.578 196.162 200.887 198.685 189.082 171.929 149.287 184.578 213.742 233.674 243.016 241.131
228.147 204.95 184.578 212.601 231.469 240.064 237.835 224.731 202.202 184.578 192.942 195.253
192.051 182.807 168.812 149.145 184.578 166.328 146.876 131.715 118.967 106.116 92.4049
 1 318 T033_CAR_NOV 8 7 99.4859 72.4199 63.9686 60.2153 55.3239 49.6277 43.515 99.4859 77.8267
66.4 60.7057 55.5367 49.7027 43.515 99.4859 92.5202 86.8077 80.8875 73.6325 65.4091 55.7752 99.4859
111.401 119.004 121.779 119.22 111.502 99.169 99.4859 123.854 142.746 154.877 159.418 156.06 145.033
99.4859 120.914 137.068 146.847 149.583 145.091 133.677 99.4859 104.306 105.63 104.408 99.9608
92.5195 81.7655 99.4859 83.9914 72.3497 65.2535 59.1255 52.5182 45.7629
 1 319 T033_CAR_DEC 8 7 69.318 44.5583 42.6086 40.1606 36.9702 33.255 29.268 69.318 49.7723
42.9113 40.1606 36.9702 33.255 29.268 69.318 63.8787 60.0111 55.9646 51.2415 45.8739 39.3746 69.318
83.2468 93.4971 99.3703 100.466 96.7098 88.3575 69.318 94.9327 116.072 131.297 139.568 140.322
133.508 69.318 90.8144 108.117 120.045 125.788 124.952 117.596 69.318 73.3044 75.1922 74.824 72.4927
67.2364 60.0496 69.318 53.3853 44.3285 40.4482 36.9702 33.255 29.268
 1 320 T033_DUB_JAN 8 7 68.8241 50.9367 48.5543 45.5919 41.7314 37.2357 32.4112 68.8241 54.5021
48.7519 45.5996 41.7314 37.2357 32.4112 68.8241 64.8299 61.1297 56.9931 51.6259 45.5842 39.287
68.8241 78.491 85.0177 87.9593 87.1167 82.6682 74.8333 68.8241 87.0566 101.565 111.361 115.776
114.511 107.65 68.8241 84.215 96.0755 103.597 106.268 103.906 96.6708 68.8241 71.653 72.0969 70.8393
67.0438 61.8754 54.9251 68.8241 57.1063 49.9589 46.2278 42.2635 37.6551 32.6893
 1 321 T033_DUB_FEB 8 7 125.314 99.5522 82.7212 77.8109 71.4116 63.9595 55.9624 125.314 104.851
89.4134 80.5267 72.7293 64.5736 56.3449 125.314 119.633 112.62 104.242 94.6525 83.0057 71.32 125.314
136.608 142.881 143.28 137.729 126.975 111.68 125.314 146.609 161.726 169.636 169.799 162.204 147.369
125.314 142.789 154.347 159.201 157.128 148.204 132.994 125.314 127.393 125.539 120.315 112.301
101.35 88.3107 125.314 109.477 94.2785 84.2711 76.0705 67.7072 59.0596
 1 322 T033_DUB_MAR 8 7 248.749 216.163 177.421 153.077 140.376 125.787 110.154 248.749 222.126
194.4 171.349 152.385 133.995 115.222 248.749 240.998 229.205 213.476 193.68 170.806 146.322 248.749
260.95 264.072 257.989 241.885 218.358 187.655 248.749 271.905 284.038 284.321 272.735 250.069
217.868 248.749 267.214 275.03 272.14 258.456 235.907 204.809 248.749 248.985 242.004 229.9 212.678
190.406 165.304 248.749 227.83 202.929 181.763 163 144.14 124.912
 1 323 T033_DUB_APR 8 7 366.91 338.317 297.961 249.076 216.724 192.326 167.319 366.91 332.064
290.902 255.634 227.148 199.964 172.778 366.91 343.397 315.754 286.661 255.828 223.883 191.216 366.91
363.177 349.35 326.436 294.433 257.111 215.653 366.91 381.791 382.176 367.87 339.786 299.839 251.303
366.91 388.334 394.841 386.267 363.19 327.84 282.042 366.91 378.334 376.401 362.545 337.592 303.204
261.754 366.91 357.593 335.789 306.503 274.241 240.422 206.145
 1 324 T033_DUB_MAY 8 7 509.391 477.285 427.297 362.834 295.922 256.988 223.282 509.391 476.331
426.917 373.844 327.095 283.918 242.526 509.391 490.078 459.41 422.574 379.896 332.652 282.437
509.391 508.658 492.126 461.105 416.967 363.336 303.75 509.391 521.53 513.086 485.357 439.338 378.728
308.557 509.391 522.414 515.455 490.038 448.95 394.926 331.554 509.391 509.91 492.892 464.092 424.92
377.063 323.066 509.391 491.185 454.453 408.036 362.415 317.633 272.18
 1 325 T033_DUB_JUN 8 7 532.228 505.511 459.761 398.097 324.901 277.921 239.076 532.228 497.78
444.98 384.785 333.214 288.652 247.75 532.228 504.852 465.331 421.656 373.875 323.849 273.66 532.228
521.641 495.782 457.028 407.296 350.412 289.713 532.228 538.299 523.55 489.744 438.531 374.234
302.017 532.228 545.822 538.388 511.625 467.785 409.151 340.559 532.228 539.481 526.995 498.521
455.793 402.215 341.971 532.228 522.784 493.13 447.59 396.279 344.414 293.51
 1 326 T033_DUB_JUL 8 7 513.398 488.929 446.596 389.285 322.709 277.748 239.155 513.398 480.537
```

```
430.942 377.018 331.063 288.482 247.187 513.398 486.662 449.841 410.22 367.112 320.529 272.213
513.398 502.249 477.804 441.09 394.611 342.174 285.621 513.398 518.544 504.198 472.272 424.381 363.69
295.964 513.398 526.742 520.505 495.76 454.487 400.066 336.455 513.398 521.624 511.226 485.733 447.5
398.404 341.648 513.398 505.943 479.478 438.273 392.438 344.433 294.549
 1 327 T033_DUB_AUG 8 7 429.099 394.69 345.711 285.499 238.798 211.544 184.922 429.099 391.62
343.845 298.109 260.435 226.443 194.477 429.099 406.979 378.29 345.005 307.851 268.439 227.681
429.099 429.241 416.718 391.726 354.669 310.01 258.973 429.099 446.599 446.211 428.348 393.658 344.47
285.188 429.099 449.926 452.763 438.105 406.915 362.642 307.623 429.099 436.113 427.332 406.019
374.262 334.082 287.426 429.099 413.225 381.994 343.444 305.472 267.683 230.32
 1 328 T033_DUB_SEP 8 7 282.536 254.297 217.654 184.892 168.342 150.252 130.921 282.536 254.389
223.931 199.011 177.46 156.12 134.562 282.536 268.205 249.71 228.538 205.025 179.33 153.297 282.536
286.697 281.45 267.307 244.904 216.087 182.392 282.536 300.441 306.795 301.163 283.929 256.268
220.064 282.536 300.987 307.861 303.036 286.673 260.473 225.815 282.536 287.702 283.086 270.39
250.076 224.082 193.205 282.536 268.359 246.041 221.244 196.732 172.147 147.577
 1 329 T033_DUB_OCT 8 7 181.334 146.959 117.806 110.327 101.343 90.8798 79.6518 181.334 151.923
128.947 114.444 103.051 91.7405 80.1106 181.334 170.972 159.415 146.468 131.71 115.519 97.9094
181.334 194.144 199.517 197.082 187.597 170.798 147.835 181.334 208.895 227.649 236.317 234.308
221.76 199.528 181.334 205.599 221.282 227.324 223.304 209.542 187.251 181.334 186.065 184.289
177.186 164.939 149.034 129.552 181.334 161.809 140.341 124.547 111.558 98.9247 85.445
 1 330 T033_DUB_NOV 8 7 87.9703 68.5092 63.6874 59.7627 54.648 48.6918 42.3 87.9703 72.3277
64.4958 59.9211 54.7161 48.7279 42.3032 87.9703 82.5295 77.9022 72.503 65.5813 58.1611 49.66 87.9703
96.4307 101.339 102.538 99.6328 92.8203 82.565 87.9703 106.366 120.396 129.104 131.897 128.583
119.389 87.9703 104.614 117.011 124.317 126.034 122.044 112.623 87.9703 92.2396 93.5704 92.0665
88.0878 81.0558 72.0176 87.9703 76.6719 67.9631 62.485 56.7882 50.42 43.8638
 1 331 T033_DUB_DEC 8 7 54.3396 41.1453 39.607 37.1599 33.9709 30.2571 26.2717 54.3396 43.3578
39.6248 37.1599 33.9709 30.2571 26.2717 54.3396 51.9968 49.4299 46.1804 41.9043 37.336 32.0154
54.3396 62.9576 69.0754 72.2827 72.358 69.2926 63.2965 54.3396 69.0809 80.9047 89.0054 92.831 92.1206
86.9227 54.3396 66.0087 74.9698 80.6122 82.5513 80.6551 75.0528 54.3396 55.5707 55.1061 53.4984
50.2268 45.6219 40.3513 54.3396 44.4197 39.8668 37.2268 34.0299 30.3043 26.3041
 1 332 T033_EDI_JAN 8 7 58.0032 37.6841 36.2902 34.1875 31.4473 28.2562 24.8317 58.0032 40.615
36.3174 34.1893 31.4473 28.2562 24.8317 58.0032 53.4552 50.0899 46.7062 42.385 37.9463 32.6605
58.0032 71.3288 81.4857 87.7866 89.8031 87.3918 80.7172 58.0032 81.5855 101.3 115.804 124.108 125.646
120.314 58.0032 77.366 93.1486 104.276 109.989 109.898 104.011 58.0032 61.1418 61.881 61.6949 59.1045
54.7829 49.1544 58.0032 43.189 37.0231 34.2791 31.4877 28.2938 24.8639
 1 333 T033_EDI_FEB 8 7 102.831 79.2473 69.2902 65.1407 59.733 53.4356 46.6776 102.831 83.8432
73.0319 66.5684 60.3852 53.8318 46.9634 102.831 96.493 91.433 85.6672 78.6581 70.0201 60.6938 102.831
112.287 117.737 118.35 114.664 106.714 94.6722 102.831 123.16 138.277 147.152 149.18 144.222 132.617
102.831 121.137 134.369 141.625 142.426 136.828 125.086 102.831 107.3 108.854 107.494 103.241 95.4112
85.2367 102.831 90.029 79.9094 72.79 66.0612 58.7389 51.2581
 1 334 T033_EDI_MAR 8 7 214.766 186.513 153.541 137.171 125.571 112.238 97.9934 214.766 189.586
168.446 151.758 136.059 119.985 103.496 214.766 205.179 195.983 184.517 169.56 150.891 130.468
214.766 222.642 224.434 218.935 206.078 187.054 162.123 214.766 235.063 246.019 246.887 237.608
218.815 191.788 214.766 234.564 245.1 246.338 237.63 220.462 195.363 214.766 220.246 220.234 215.276
204.344 187.851 166.593 214.766 200.334 183.707 169.587 155.247 139.23 121.344
 1 335 T033_EDI_APR 8 7 311.333 287.052 252.89 213.052 188.894 167.916 146.272 311.333 286.839
257.568 230.863 205.941 180.914 155.441 311.333 299.639 284.377 265.728 242.243 214.446 184.241
311.333 314.231 309.169 295.252 272.273 241.928 206.47 311.333 324.144 325.315 314.008 290.983
258.084 217.803 311.333 324.956 326.864 317.324 297.341 267.651 230.36 311.333 314.59 309.281 296.791
277.001 250.61 218.66 311.333 298.773 277.732 255.382 232.516 207.265 180.228
 1 336 T033_EDI_MAY 8 7 451.471 430.083 392.82 342.22 289.052 253.115 218.796 451.471 425.405
386.097 345.463 308.366 271.071 232.845 451.471 434.44 412.553 385.27 351.351 311.652 267.964 451.471
447.197 433.674 409.042 373.211 327.735 276.908 451.471 456.388 445.166 418.756 378.087 327.096
267.945 451.471 460.236 453.719 433.255 399.512 354.038 300.153 451.471 454.896 445.972 427.447
398.849 360.466 314.003 451.471 442.235 416.947 384.952 351.702 314.991 274.709
 1 337 T033_EDI_JUN 8 7 472.615 457.141 424.729 377.588 320.777 279.223 240.778 472.615 453.702
418.346 377.15 337.731 297.406 255.213 472.615 459.487 439.421 412.565 377.356 335.259 288.033
472.615 467.303 452.901 426.196 387.862 339.969 286.263 472.615 471.434 454.803 423.532 379.505
325.688 265.805 472.615 473.536 459.765 433.862 396.042 348.154 293.528 472.615 471.166 457.577
435.381 404.506 364.549 317.191 472.615 464.375 438.703 403.716 369.192 331.021 289.184
 1 338 T033_EDI_JUL 8 7 489.02 472.058 437.777 388.514 332.176 291.081 252.152 489.02 456.811
410.163 363.997 324.608 286.147 246.686 489.02 460.282 429.442 396.443 358.81 316.846 272.068 489.02
```

473.839 452.491 421.092 379.979 331.967 279.727 489.02 489.236 473.607 443.473 399.095 343.689 282.001 489.02 502.759 499.095 479.037 443.65 394.659 335.445 489.02 504.584 504.36 490.332 462.142 420.448 367.743 489.02 491.83 476.123 448.167 413.782 372.918 326.274
1 339 T033_EDI_AUG 8 7 359.972 333.376 295.025 247.908 215.977 191.28 166.332 359.972 337.581 307.427 276.98 246.605 215.817 184.844 359.972 352.902 340.015 320.952 294.429 261.161 223.987 359.972 366.959 363.825 349.528 323.808 288.785 246.462 359.972 373.011 372.348 357.606 329.576 290.134 242.879 359.972 369.349 366.09 350.92 325.153 289.83 247.14 359.972 356.312 344.235 325.793 301.001 270.588 235.54 359.972 341.293 312.402 283.402 257.112 229.833 200.505
1 340 T033_EDI_SEP 8 7 272.697 239.503 198.231 168.293 153.673 137.496 120.226 272.697 244.503 217.29 193.79 172.686 151.54 130.538 272.697 263.785 253.631 239.035 219.512 195.33 168.049 272.697 283.441 286.338 279.457 263.433 238.14 205.969 272.697 295.491 306.33 304.477 290.058 264.055 228.386 272.697 292.506 300.994 298.024 284.086 259.52 227.167 272.697 275.036 270.416 260.641 245.059 223.66 196.544 272.697 252.925 229.071 209.233 190.698 170.813 149.637
1 341 T033_EDI_OCT 8 7 144.955 114.439 95.1435 89.4819 82.1432 73.5971 64.426 144.955 121.871 105.814 95.0401 85.3485 75.5935 65.4126 144.955 139.375 133.99 126.75 116.596 104.443 89.7521 144.955 158.64 166.51 168.408 163.406 151.659 134.351 144.955 170.417 188.656 198.429 199.07 190.535 173.406 144.955 165.814 179.764 185.985 183.986 173.839 156.609 144.955 147.45 146.476 142.741 135.028 124.413 109.743 144.955 126.315 110.847 101.093 91.9377 82.2109 71.4264
1 342 T033_EDI_NOV 8 7 75.888 48.4134 45.1922 42.6361 39.3048 35.4255 31.2624 75.888 53.7441 46.1143 42.8129 39.4084 35.456 31.2624 75.888 69.7933 65.502 61.0241 56.2848 50.0498 42.6905 75.888 91.3184 102.656 109.128 110.294 106.074 96.7558 75.888 104.029 127.212 143.856 152.826 153.513 145.868 75.888 99.365 118.201 131.112 137.219 136.104 127.845 75.888 80.0589 81.79 81.2693 78.5934 73.0394 64.6742 75.888 58.1225 48.4936 44.04 40.0785 35.6759 31.2624
1 343 T033_EDI_DEC 8 7 42.7356 28.3559 27.3493 25.748 23.6612 21.231 18.6232 42.7356 29.1444 27.3493 25.748 23.6612 21.231 18.6232 42.7356 39.276 36.9356 34.1404 31.5366 28.0589 23.9382 42.7356 54.8839 64.5611 71.1077 74.0775 73.2683 68.735 42.7356 62.8942 80.0358 92.9922 100.88 103.163 99.6844 42.7356 58.6146 71.7682 81.3 86.5606 87.1913 83.1492 42.7356 44.552 44.6526 44.1666 42.0448 38.3728 34.1306 42.7356 30.1398 27.3493 25.748 23.6612 21.231 18.6232
1 344 T033_GLA_JAN 8 7 60.0801 40.9319 39.4054 37.0736 34.0348 30.4959 26.6983 60.0801 43.6724 39.4834 37.0822 34.0348 30.4959 26.6983 60.0801 54.4133 51.0789 47.8099 43.4284 38.8266 33.5827 60.0801 71.1759 79.2407 83.772 84.6874 81.6519 74.8715 60.0801 83.8283 103.683 118.292 126.658 128.213 122.849 60.0801 81.903 99.9639 113.032 120.216 121.028 115.41 60.0801 66.5279 70.4928 72.6924 71.8319 68.6155 62.8897 60.0801 47.3728 41.2762 38.3212 35.2417 31.5904 27.6057
1 345 T033_GLA_FEB 8 7 119.304 84.408 71.5073 67.4512 62.1652 56.0094 49.4035 119.304 90.7233 76.5348 69.369 63.0426 56.4272 49.7184 119.304 108.777 102.418 95.7203 88.1794 78.4839 68.4255 119.304 132.206 140.927 143.411 139.92 131.4 117.292 119.304 150.926 175.629 191.73 198.132 194.398 180.783 119.304 150.13 174.092 189.556 195.522 191.619 178.024 119.304 129.911 136.629 139.333 137.369 130.134 118.497 119.304 102.4 89.7985 82.0544 75.0621 67.7214 59.7991
1 346 T033_GLA_MAR 8 7 214.941 184.113 149.847 133.619 122.437 109.626 95.9296 214.941 185.931 163.62 146.548 131.167 115.745 100.025 214.941 202.105 191.931 179.963 164.617 145.796 125.74 214.941 221.525 222.813 217.328 204.145 185.441 160.702 214.941 237.658 250.716 253.227 245.017 226.648 199.37 214.941 240.132 255.508 260.562 254.484 238.531 213.024 214.941 225.496 229.367 227.213 218.508 203.003 182.104 214.941 202.302 186.548 173.596 160.399 145 127.414
1 347 T033_GLA_APR 8 7 343.575 313.343 271.845 223.491 197.796 176.366 154.229 343.575 315.815 284.235 254.479 225.472 196.736 168.028 343.575 333.871 321.868 304.605 280.426 249.239 213.972 343.575 351.08 350.461 339.503 316.799 284.374 244.173 343.575 360.507 364.174 353.26 328.359 291.307 245.286 343.575 359.181 361.477 350.931 328.457 295.634 254.074 343.575 344.979 337.535 323.755 303.627 277.115 244.601 343.575 325.852 298.975 274.418 252.716 228.692 201.442
1 348 T033_GLA_MAY 8 7 481.267 455.064 411.199 352.663 293.046 257.922 225.088 481.267 446.516 398.167 349.45 307.466 267.629 229.172 481.267 458.226 431.501 399.418 361.173 317.783 271.14 481.267 475.909 461.633 434.987 395.686 345.94 290.099 481.267 489.578 480.062 453.632 410.358 354.175 288.761 481.267 496.598 493.277 472.93 436.56 386.277 325.975 481.267 490.54 483.853 464.87 434.578 393.873 344.365 481.267 473.292 446.715 410.755 375.263 337.877 296.854
1 349 T033_GLA_JUN 8 7 490.054 471.042 434.228 382.121 320.721 279.892 242.587 490.054 458.787 410.989 360.146 317.626 278.727 239.832 490.054 463.002 430.957 395.373 355.407 312.131 266.83 490.054 476.937 455.756 423.31 380.33 329.691 275.555 490.054 491.256 476.241 445.305 399.865 343.007 279.147 490.054 502.222 496.442 474.05 436.015 384.665 323.742 490.054 501.755 496.992 478.076 446.18 402.285 348.937 490.054 488.84 468.612 434.788 396.564 353.914 307.761
1 350 T033_GLA_JUL 8 7 481.996 463.201 427.224 376.514 319.467 280.967 244.297 481.996 449.89 403.572 356.57 315.135 275.972 237.005 481.996 455.295 426.003 392.866 354.148 311.036 265.465 481.996 469.441 450.552 420.658 379.848 331.607 278.137 481.996 483.979 470.237 441.347 398.037

342.782 281.241 481.996 495.655 491.555 470.509 434.077 384.579 325.436 481.996 495.601 492.734 476.054 446.317 404.728 353.581 481.996 482.128 463.878 433.053 397.975 358.542 314.321
1 351 T033_GLA_AUG 8 7 366.24 341.579 304.855 258.837 226.043 200.069 173.628 366.24 334.861 299.5 268.114 239.191 209.908 180.467 366.24 346.23 326.758 304.27 276.878 245.239 210.744 366.24 362.53 352.935 334.738 306.919 272.037 231.419 366.24 377.158 374.729 358.917 329.864 289.57 242.359 366.24 384.921 389.501 380.125 357.512 323.254 279.198 366.24 377.802 378.118 368.438 348.534 319.274 281.64 366.24 359.764 341.629 319.267 294.526 265.508 232.95
1 352 T033_GLA_SEP 8 7 287.924 243.024 189.439 156.934 143.981 129.686 114.522 287.924 244.433 206.861 180.374 159.979 141.097 122.745 287.924 269.055 252.619 234.503 213.405 188.593 161.968 287.924 298.319 301.503 294.237 277.009 249.829 214.892 287.924 321.004 340.009 343.644 331.662 304.879 265.167 287.924 323.612 345.088 351.82 343.226 319.714 283.469 287.924 301.873 306.344 302.404 289.413 268.41 238.62 287.924 268.488 243.295 222.218 203.7 184.02 162.571
1 353 T033_GLA_OCT 8 7 148.226 115.984 95.9986 90.327 82.9672 74.3966 65.1992 148.226 122.36 105.805 95.3967 85.7102 75.9127 65.7903 148.226 139.899 133.246 125.49 115.08 103.204 88.8874 148.226 160.539 167.424 168.407 162.812 150.564 132.732 148.226 175.486 195.23 206.112 207.392 198.981 181.453 148.226 173.481 191.357 200.646 200.707 191.534 174.3 148.226 155.032 157.646 156.945 151.355 141.2 126.088 148.226 131.275 116.953 107.103 97.961 88.0989 76.8326
1 354 T033_GLA_NOV 8 7 81.4027 52.7565 49.3712 46.5537 42.8819 38.6061 34.0175 81.4027 57.6421 50.1648 46.5781 42.8819 38.6061 34.0175 81.4027 73.1332 68.3967 63.6442 58.5417 52.2012 44.6215 81.4027 95.8331 106.527 112.361 112.855 107.977 98.0592 81.4027 112.758 138.747 157.599 168.029 169.327 161.403 81.4027 110.37 134.133 151.074 160.037 160.413 152.175 81.4027 90.0669 95.5906 98.1427 97.5223 92.7704 84.3358 81.4027 64.3626 54.6993 49.9034 45.6862 40.8895 36.0663
1 355 T033_GLA_DEC 8 7 44.5896 30.0678 28.9931 27.2834 25.0553 22.4606 19.6762 44.5896 30.936 28.9931 27.2834 25.0553 22.4606 19.6762 44.5896 40.3633 38.2423 35.4974 32.8726 29.4238 25.3108 44.5896 56.4185 65.7435 71.929 74.5536 73.4384 68.6594 44.5896 67.7538 87.6417 102.898 112.482 115.742 112.456 44.5896 65.0226 82.3653 95.4359 103.344 105.549 101.903 44.5896 49.8247 53.1354 55.3418 55.1913 52.8125 48.8176 44.5896 32.2832 29.0072 27.2834 25.0553 22.4606 19.6762
1 356 T033_JER_JAN 8 7 93.2855 75.4859 68.3159 64.0867 58.5752 52.157 45.2693 93.2855 78.417 69.3705 64.1822 58.5752 52.157 45.2693 93.2855 88.2312 82.0651 75.2828 67.3186 59.1343 50.389 93.2855 100.596 104.139 103.688 99.3925 91.4447 80.3501 93.2855 108.156 118.741 124.319 124.51 119.3 109.046 93.2855 106.048 114.669 118.561 117.458 111.435 100.903 93.2855 95.5062 94.4652 90.3805 84.1194 75.2482 65.2903 93.2855 82.757 72.3881 65.6817 59.4576 52.7975 45.79
1 357 T033_JER_FEB 8 7 163.809 126.322 94.9417 89.362 82.496 74.5003 65.92 163.809 130.933 106.377 93.2265 84.0652 75.1477 66.3266 163.809 151.274 138.682 125.915 113.027 98.3781 84.3218 163.809 177.261 183.82 182.345 173.653 158.495 137.029 163.809 194.841 217.087 229.032 229.86 219.517 198.706 163.809 192.404 212.378 222.372 221.704 210.567 189.695 163.809 171.267 172.288 167.945 158.48 143.82 126.005 163.809 143.905 122.467 106.249 94.1712 83.1613 72.61
1 358 T033_JER_MAR 8 7 320.411 269.485 208.73 163.791 151.277 137.228 122.17 320.411 273.24 226.175 190.636 165.478 145.097 126.287 320.411 300.839 277.504 252.127 224.342 193.906 164.22 320.411 335.027 336.805 325.604 302.507 269.057 227.3 320.411 357.827 379.184 383.026 369.091 338.328 292.835 320.411 355.107 373.933 375.898 360.766 330.329 286.443 320.411 328.124 322.88 307.798 284.607 255.159 220.049 320.411 292.657 255.549 221.44 193.7 169.216 146.099
1 359 T033_JER_APR 8 7 475.273 419.41 346.005 260.061 202.802 182.548 163.904 475.273 416.618 347.054 284.686 238.241 203.705 175.706 475.273 443.459 402.994 360.401 315.318 268.567 224.305 475.273 480.998 468.886 441.367 399.041 343.652 282.512 475.273 509.79 520.614 507 469.984 412.693 338.499 475.273 513 527.044 516.772 484.274 431.584 362.714 475.273 487.868 479.65 454.753 416.84 369.04 314.18 475.273 449.094 403.901 349.695 300.965 259.97 222.606
1 360 T033_JER_MAY 8 7 604.629 557.217 486.096 396.113 295.864 248.243 218.089 604.629 551.757 477.107 396.744 330.691 280.405 238.717 604.629 570.461 521.246 464.996 406.096 346.333 288.801 604.629 600.627 575.023 531.482 472.429 401.577 325.344 604.629 625.152 617.939 582.965 523.547 443.82 349.23 604.629 630.646 628.233 599.307 547.101 475.674 391.275 604.629 613.076 595.52 557.759 505.968 443.162 373.411 604.629 582.643 535.292 470.075 404.46 345.682 292.441
1 361 T033_JER_JUN 8 7 624.942 585.54 521.414 436.934 337.858 268.847 233.087 624.942 574.744 500.745 413.915 342.008 288.701 246.293 624.942 585.803 527.719 464.256 400.141 338.518 282.099 624.942 611.61 575.542 522.026 455.925 382.206 307.328 624.942 636.935 621.109 578.506 512.555 428.236 332.374 624.942 647.731 641.777 608.656 551.714 475.756 387.294 624.942 637.15 621.897 583.063 526.734 457.9 382.217 624.942 611.389 571.351 508.796 437.156 369.697 307.915
1 362 T033_JER_JUL 8 7 621.788 581.431 516.445 431.258 331.689 268.032 232.066 621.788 566.815 488.986 404.912 339.432 289.01 246.812 621.788 577.295 518.308 458.555 399.944 342.768 287.557 621.788 605.435 568.637 516.952 453.664 383.167 311.75 621.788 634.766 620.046 578.614 513.151 430.549 335.101 621.788 649.383 647.987 618.92 565.668 492.781 406.322 621.788 639.823 630.254 597.4

547.647 483.834 410.103 621.788 611.67 574.875 516.029 450.791 387.42 325.547
1 363 T033_JER_AUG 8 7 546.405 494.196 421.415 333.021 251.851 220.305 195.72 546.405 492.293 421.173 352.226 297.623 253.445 215.958 546.405 515.99 473.275 426.863 377.626 325.658 272.888 546.405 549.532 531.949 497.878 448.336 386.25 318.971 546.405 574.489 576.746 552.83 504.345 435.679 350.824 546.405 576.476 580.743 559.645 516.236 454.376 378.757 546.405 553.785 538.133 505.956 461.848 408.316 348.016 546.405 519.704 471.047 411.05 356.504 308.24 262.419
1 364 T033_JER_SEP 8 7 370.548 320.942 259.133 196.991 176.353 159.702 142.057 370.548 322.838 271.002 229.317 197.36 171.368 148.334 370.548 348.7 321.363 291.618 259.371 224.202 187.578 370.548 381.672 378.816 361.776 333.43 292.783 246.346 370.548 404.57 420.689 417.807 396.121 357.179 303.613 370.548 403.339 418.372 414.751 393.446 355.629 305.136 370.548 378.352 371.131 352.147 324.149 288.864 247.773 370.548 344.206 305.338 264.843 231.727 202.493 174.673
1 365 T033_JER_OCT 8 7 206.995 174.757 140.482 127.343 116.917 104.775 91.7449 206.995 177.882 151.617 133.601 119.349 105.847 92.234 206.995 195.093 180.672 164.649 146.592 128.02 107.969 206.995 216.773 218.387 212.277 198.387 177.229 151.882 206.995 231.407 246.302 250.665 244.197 227.34 201.242 206.995 229.54 242.695 245.581 238.058 220.563 194.515 206.995 212.174 209.535 200.155 185.446 166.263 143.12 206.995 189.482 167.119 147.244 130.794 114.948 99.3112
1 366 T033_JER_NOV 8 7 114.218 89.9979 77.6149 72.9504 66.8716 59.7927 52.196 114.218 93.8431 80.4286 73.5269 67.0462 59.8283 52.196 114.218 107.141 98.946 90.6134 81.1276 71.0808 60.2333 114.218 124.094 129.186 129.091 123.756 113.544 99.2362 114.218 134.495 149.163 157.223 158.125 151.809 138.704 114.218 131.779 143.917 149.805 149.04 141.676 128.213 114.218 117.539 116.707 112.306 104.342 94.1824 81.2297 114.218 100.187 85.962 76.4738 68.8185 61.1071 52.9122
1 367 T033_JER_DEC 8 7 75.9985 57.8493 53.9906 50.687 46.3817 41.368 35.9878 75.9985 61.196 54.2552 50.687 46.3817 41.368 35.9878 75.9985 71.2676 65.9535 60.604 54.5169 47.9941 40.6646 75.9985 84.3981 89.4994 90.9548 88.6644 82.7841 73.7147 75.9985 92.1496 104.473 112.13 114.598 111.708 103.658 75.9985 89.6414 99.6279 105.277 106.205 102.347 93.9674 75.9985 78.3427 77.9689 75.0446 70.3381 63.491 54.9894 75.9985 64.9835 55.8448 51.0849 46.4462 41.3742 35.9935
1 368 T033_LEE_JAN 8 7 63.288 44.2031 42.0576 39.5538 36.2907 32.4908 28.413 63.288 47.7679 42.5024 39.5825 36.2907 32.4908 28.413 63.288 58.6588 55.5542 51.8003 47.3501 42.3092 36.3389 63.288 74.1137 81.8251 85.9477 86.158 82.4331 75.0268 63.288 83.9291 100.787 112.713 118.893 118.908 112.756 63.288 80.9571 95.0455 104.593 108.949 107.816 101.273 63.288 66.9386 68.4797 68.1938 65.9649 61.177 55.0354 63.288 50.5603 43.7011 40.1041 36.5216 32.7107 28.6079
1 369 T033_LEE_FEB 8 7 107.855 85.6007 71.6687 67.4025 61.8428 55.3683 48.4204 107.855 91.3384 78.8873 70.78 63.7127 56.4534 49.1097 107.855 104.311 100.279 94.2611 86.4652 76.992 66.3387 107.855 118.325 124.555 125.693 122.076 113.545 100.576 107.855 126.551 139.923 147.059 147.472 141.136 128.481 107.855 122.845 132.764 136.934 135.073 127.553 114.853 107.855 109.172 107.854 104.432 98.5142 89.6482 79.2526 107.855 93.7683 82.117 74.7552 67.9596 60.7004 53.0956
1 370 T033_LEE_MAR 8 7 232.092 195.998 154.369 131.884 121.312 109.208 96.2539 232.092 196.998 169.613 149.957 133.845 118.481 103.017 232.092 215.845 202.64 188.883 172.364 152.575 131.246 232.092 239.15 240.573 233.462 219.162 197.532 169.73 232.092 258.929 274.68 278.272 269.459 248.843 217.828 232.092 262.168 280.938 288.102 282.775 265.293 237.105 232.092 245.144 250.937 249.562 239.976 223.163 199.3 232.092 217.741 199.162 182.249 167.025 150.436 131.886
1 371 T033_LEE_APR 8 7 318.43 291.735 254.873 212.934 188.416 167.421 145.868 318.43 291.915 262.548 235.037 208.963 183.487 157.751 318.43 306.285 292.32 273.715 249.725 220.788 188.719 318.43 322.176 318.374 305.103 282.442 251.181 214.42 318.43 333.256 335.44 324.478 301.166 267.603 225.85 318.43 334.495 338.246 330.117 310.914 281.002 242.979 318.43 323.085 318.906 307.337 287.947 261.54 229.267 318.43 305.221 283.27 259.98 237.102 212.957 186.536
1 372 T033_LEE_MAY 8 7 485.422 460.199 416.928 358.558 295.931 256.949 222.221 485.422 452.627 406.294 360.18 319.474 280.238 241.036 485.422 463.145 438.499 409.132 373.33 330.933 283.971 485.422 478.915 463.9 437.667 399.216 350.129 294.295 485.422 491.027 479.657 450.107 404.863 348.138 281.491 485.422 498.773 494.831 475.087 439.595 390.192 330.745 485.422 493.242 487.427 470.216 441.747 401.589 351.568 485.422 477.063 450.391 415.607 380.38 342.261 300.183
1 373 T033_LEE_JUN 8 7 473.906 459.854 428.991 383.423 326.57 281.665 241.894 473.906 456.925 423.956 384.217 344.03 302.282 258.616 473.906 462.293 442.992 416.653 381.286 338.482 290.326 473.906 468.687 453.461 426.594 388.481 340.809 286.81 473.906 471.688 453.778 421.169 375.876 321.748 261.867 473.906 473.121 457.254 429.465 390.846 343.324 289.903 473.906 471.306 455.873 432.518 401.492 361.524 314.6 473.906 465.81 440.499 405.266 370.474 332.139 289.852
1 374 T033_LEE_JUL 8 7 485.763 468.047 433.062 383.192 324.502 282.372 243.574 485.763 463.086 425.271 383.365 342.414 300.505 257.562 485.763 470.11 449.277 421.288 384.875 341.424 292.905 485.763 479.767 465.024 438.37 399.554 350.481 295.378 485.763 486.245 470.101 438.848 393.561 338.497 276.33 485.763 489.964 478.215 452.715 414.814 365.968 309.487 485.763 487.133 475.22 453.991 423.021 382.144 333.582 485.763 477.975 452.477 418.091 382.832 343.954 300.979

```
 1 375 T033_LEE_AUG 8 7 407.927 379.176 336.577 283.034 240.558 212.452 185.137 407.927 378.811
342.904 307.177 272.768 238.259 203.388 407.927 394.537 377.794 354.875 324.411 287.511 246.443
407.927 410.673 404.317 386.463 356.438 315.717 268.084 407.927 420.951 418.541 399.801 366.028
320.466 265.685 407.927 421.918 420.619 404.838 376.531 336.459 287.888 407.927 410.223 401.059
383.956 358.883 325.719 285.396 407.927 392.411 363.706 332.627 303.489 272.44 238.672
 1 376 T033_LEE_SEP 8 7 287.441 252.521 209.024 176.211 160.667 143.959 126.048 287.441 257.11
227.549 201.672 179.166 157.265 135.835 287.441 277.04 265.345 249.276 227.842 202.145 173.024
287.441 297.8 300.264 292.283 274.905 247.556 213.947 287.441 311.326 322.55 320.346 304.865 277.358
239.651 287.441 309.257 319.057 316.215 302.073 276.014 241.443 287.441 291.359 287.544 277.396
260.773 237.699 208.773 287.441 267.695 242.973 221.379 201.82 181.217 158.895
 1 377 T033_LEE_OCT 8 7 155.175 124.558 98.0893 91.6192 84.2695 75.7106 66.5258 155.175 130.774
112.464 100.306 89.7941 79.3386 68.5606 155.175 147.945 141.65 133.28 122.055 109.168 94.0359 155.175
166.764 173.136 173.458 166.81 153.327 135.065 155.175 179.859 196.82 204.901 203.552 192.865 173.568
155.175 177.564 192.385 198.762 196.574 185.523 166.734 155.175 160.394 161.466 159.14 152.475
141.462 125.832 155.175 138.442 123.092 112.48 103.146 93.0197 81.8582
 1 378 T033_LEE_NOV 8 7 78.4283 57.3915 52.9248 49.7535 45.6205 40.8075 35.6425 78.4283 62.2427
54.5219 50.2086 45.838 40.8123 35.6425 78.4283 73.8501 70.2645 65.5112 60.2399 53.4252 45.961 78.4283
88.9501 95.9458 98.8318 97.4117 91.7821 82.3965 78.4283 98.7982 114.864 125.531 130.073 128.179
119.979 78.4283 95.8602 109.188 117.504 120.242 117.214 108.627 78.4283 81.8567 82.9431 82.0345
78.6009 72.9934 64.8404 78.4283 65.4018 56.5443 51.7836 46.9674 41.5889 36.1115
 1 379 T033_LEE_DEC 8 7 50.6124 35.9307 34.6192 32.533 29.8141 26.648 23.2502 50.6124 38.0196
34.6202 32.533 29.8141 26.648 23.2502 50.6124 47.4599 44.9106 41.6969 38.2843 33.8542 28.7473 50.6124
60.4926 67.8348 72.1387 73.1109 70.6851 65.0268 50.6124 67.99 82.3186 92.6218 98.1974 98.6655 93.9942
50.6124 64.7616 76.0819 83.8017 87.3951 86.6172 81.5208 50.6124 52.6987 53.0254 51.9082 49.7063
45.704 40.4104 50.6124 39.414 34.8497 32.5858 29.8141 26.648 23.2502
 1 380 T033_LON_JAN 8 7 76.3947 53.2327 49.0584 46.1816 42.4325 38.0666 33.3814 76.3947 58.3396
50.3371 46.3389 42.4325 38.0666 33.3814 76.3947 71.0597 67.1488 63.0936 57.6898 51.7208 44.5308
76.3947 88.1651 96.209 100.106 99.4785 94.3469 85.0607 76.3947 98.5471 116.259 128.322 133.915
132.658 124.634 76.3947 95.094 109.588 118.888 122.361 119.77 111.293 76.3947 79.8286 80.9841 80.0553
77.1271 71.2529 63.4609 76.3947 62.1919 52.8545 47.8418 43.2206 38.673 33.9207
 1 381 T033_LON_FEB 8 7 120.414 99.1017 83.2116 78.1718 71.6103 63.9693 55.7694 120.414 104.296
91.3272 81.9925 73.7414 65.2049 56.4737 120.414 116.452 111.629 104.692 95.6323 84.7791 72.6378
120.414 129.658 134.357 133.809 128.574 118.406 103.97 120.414 137.617 149.243 154.498 153.026
144.925 130.749 120.414 134.428 143.082 145.785 142.355 133.15 119.131 120.414 121.743 119.864
115.416 108.143 98.1635 86.2694 120.414 107.098 94.7237 86.1289 78.2386 69.6358 60.5902
 1 382 T033_LON_MAR 8 7 240.939 208.44 170.073 145.487 133.413 119.645 104.874 240.939 212.28
186.161 165.132 147.204 129.414 111.851 240.939 230.162 218.646 204.359 186.85 165.762 142.58 240.939
249.955 252.101 245.083 230.274 207.343 178.586 240.939 264.161 276.526 277.189 266.106 244.033
212.519 240.939 262.942 274.293 274.813 264.493 243.765 214.785 240.939 246.227 244.895 237.828
224.44 205.236 180.506 240.939 223.617 202.834 184.408 166.665 148.483 129.287
 1 383 T033_LON_APR 8 7 346.863 317.365 276.426 227.059 195.644 173.734 151.883 346.863 320.811
289.558 258.684 228.855 199.636 170.434 346.863 337.442 324.004 305.082 279.715 248.057 211.909
346.863 353.76 351.068 337.156 312.832 278.38 237.775 346.863 362.91 364.411 351.265 324.72 286.945
239.868 346.863 360.741 361.345 349.158 325.692 291.35 249.981 346.863 346.675 337.402 321.206
298.883 270.411 236.42 346.863 328.523 300.65 272.266 247.315 221.255 192.892
 1 384 T033_LON_MAY 8 7 463.842 442.387 404.361 352.356 292.984 254.387 220.408 463.842 436.479
396.405 352.261 310.413 270.43 231.282 463.842 445.529 421.729 390.799 353.378 310.281 263.779
463.842 458.254 443.008 415.635 376.965 328.851 274.701 463.842 467.643 455.966 426.94 384.395
330.579 268.466 463.842 473.537 466.66 444.555 408.212 359.944 303.576 463.842 468.871 459.645
438.361 406.817 366.075 317.96 463.842 455.946 430.855 395.34 358.35 319.823 279.031
 1 385 T033_LON_JUN 8 7 542.135 523.167 483.834 426.815 355.997 300.044 256.618 542.135 519.572
477.795 428.759 380.611 332.52 283.469 542.135 525.622 502.065 470.962 430.241 380.874 325.643
542.135 534.237 515.822 484.384 439.855 384.263 321.391 542.135 539.444 517.973 478.269 424.316
359.196 288.062 542.135 543.039 525.562 493.572 448.161 391.83 328.804 542.135 539.761 521.727
494.262 457.367 411.934 358.312 542.135 531.53 499.989 455.932 413.08 369.478 322.257
 1 386 T033_LON_JUL 8 7 519.746 495.643 452.05 391.94 320.366 273.052 235.748 519.746 487.594
438.676 385.269 336.99 293.445 251.147 519.746 495.574 465.531 429.576 386.873 339.379 289.262
519.746 510.249 490.411 457.843 413.027 358.312 297.728 519.746 522.719 506.861 472.506 422.038
359.36 288.054 519.746 530.767 522.268 496.315 454.454 399.121 334.551 519.746 526.231 515.758
491.866 456.458 410.061 354.907 519.746 511.622 483.162 441.994 398.749 353.902 307.159
 1 387 T033_LON_AUG 8 7 443.917 410.083 360.378 298.191 243.378 213.168 186.448 443.917 406.46
```

360.021 316.907 280.215 245.014 210.542 443.917 422.749 397.618 369.777 336.902 298.643 256.469 443.917 443.419 431.958 408.775 373.959 328.788 277.676 443.917 459.619 457.092 435.55 397.045 345.215 282.06 443.917 463.498 465.425 450.774 420.891 376.42 321.139 443.917 450.769 444.408 428.099 400.979 363.416 317.264 443.917 428.457 397.623 362.531 327.916 291.867 253.057
1 388 T033_LON_SEP 8 7 307.562 266.756 216.309 174.449 158.48 142.738 125.993 307.562 274.154 239.763 209.841 184.479 160.785 138.504 307.562 298.561 286.457 269.595 247.142 219.517 187.612 307.562 322.643 328.111 321.33 303.025 273.034 235.656 307.562 335.936 349.943 348.63 332.24 301.96 259.637 307.562 330.445 339.897 335.244 318.64 289.118 250.934 307.562 308.041 300.353 287.18 268.355 243.281 212.968 307.562 281.535 250.167 224.493 203.082 181.634 159.084
1 389 T033_LON_OCT 8 7 181.016 146.232 115.81 107.04 98.4467 88.4397 77.701 181.016 152.612 131.38 117.085 105.134 93.0283 80.5497 181.016 171.791 163.623 153.745 140.823 125.602 108.312 181.016 193.801 200.067 199.479 190.847 174.692 153.445 181.016 209.238 228.497 237.478 235.571 222.906 200.345 181.016 206.504 223.215 230.016 227.364 214.55 192.63 181.016 186.82 187.873 184.715 176.339 162.625 144.173 181.016 161.75 143.596 129.92 117.921 105.28 92.0807
1 390 T033_LON_NOV 8 7 85.7841 61.3129 54.0445 50.9032 46.8094 42.042 36.9259 85.7841 67.7425 57.2895 51.8657 47.1692 42.046 36.9259 85.7841 82.2472 78.7264 74.1494 68.6563 61.2104 52.9515 85.7841 98.6871 107.669 111.831 110.889 104.906 94.5346 85.7841 107.824 125.033 136.238 140.674 138.04 128.516 85.7841 103.076 115.86 123.264 124.785 120.318 110.169 85.7841 87.2228 86.4722 84.3876 79.7246 73.5268 64.9425 85.7841 70.0036 59.6364 54.0145 49.0818 43.8688 38.2499
1 391 T033_LON_DEC 8 7 59.4828 37.7973 36.3167 34.237 31.5266 28.3704 24.9833 59.4828 43.1275 36.5684 34.237 31.5266 28.3704 24.9833 59.4828 56.4512 54.1923 50.9042 47.1984 41.9896 35.8858 59.4828 72.888 83.0286 89.2135 91.0212 88.3286 81.319 59.4828 82.3106 101.232 114.957 122.55 123.494 117.725 59.4828 77.6724 92.2713 102.285 107.03 106.184 99.8043 59.4828 61.6903 62.2662 61.3319 59.2631 55.119 48.9569 59.4828 44.5622 37.5642 34.6365 31.5769 28.3704 24.9833
1 392 T033_MAN_JAN 8 7 67.9824 45.3286 43.2188 40.696 37.4083 33.5797 29.471 67.9824 50.247 43.6915 40.7087 37.4083 33.5797 29.471 67.9824 63.8095 60.2837 56.5781 51.5962 46.2006 39.6848 67.9824 81.3365 91.1561 96.7948 97.8675 94.2792 86.2742 67.9824 91.0507 109.922 123.312 130.306 130.428 123.67 67.9824 86.4831 101.099 110.833 115.022 113.382 106.022 67.9824 70.3093 70.5388 69.1308 65.8916 60.2002 53.7455 67.9824 52.5702 44.5881 41.0438 37.5789 33.7387 29.6077
1 393 T033_MAN_FEB 8 7 120.894 90.9852 73.7946 69.5692 64.0625 57.6499 50.7683 120.894 97.7093 82.1152 73.447 66.2643 58.9904 51.6593 120.894 114.826 109.224 102.363 93.9915 83.6205 72.3217 120.894 134.218 142.638 144.808 141.299 131.912 116.994 120.894 146.377 165.344 176.503 179.094 172.939 158.459 120.894 142.38 157.622 165.582 165.718 158.401 143.905 120.894 124.414 124.758 122.148 116.52 106.758 95.0863 120.894 103.237 88.7857 79.969 72.6375 64.7942 56.9429
1 394 T033_MAN_MAR 8 7 237.714 205.748 168.807 148.645 136.08 121.689 106.365 237.714 208.203 183.964 165.185 147.664 129.886 112.123 237.714 225.405 214.425 201.631 185.392 165.159 142.551 237.714 245.332 246.858 240.226 225.807 204.339 176.767 237.714 260.729 273.224 274.348 264.024 242.956 212.58 237.714 261.688 275.08 277.629 268.928 250.043 222.045 237.714 246.001 247.599 243.442 232.563 215.193 191.788 237.714 222.828 205.012 190.127 174.4 156.456 136.478
1 395 T033_MAN_APR 8 7 343.591 314.8 274.682 226.791 198.396 176.125 153.919 343.591 315.034 281.734 251.053 223.403 196.057 168.356 343.591 330.685 315.12 295.103 269.666 239.152 205.574 343.591 347.795 343.592 329.605 304.971 270.877 231.013 343.591 359.272 361.293 348.82 322.714 285.607 239.489 343.591 359.864 362.871 353.011 331.186 298.231 256.904 343.591 347.414 342.021 329.188 308.612 280.176 245.229 343.591 328.544 303.921 278.574 253.645 227.128 198.547
1 396 T033_MAN_MAY 8 7 464.17 438.026 394.664 337.038 275.409 239.249 207.359 464.17 433.347 389.3 343.874 302.756 263.941 226.194 464.17 445.158 422.063 393.126 357.346 315.364 269.48 464.17 460.805 447.847 422.962 385.726 337.782 282.959 464.17 471.358 461.737 434.293 391.115 336.293 271.665 464.17 476.259 471.129 450.405 415.014 366.792 309.483 464.17 468.735 459.29 439.43 409.904 370.438 322.962 464.17 452.688 423.765 387.001 351.398 314.609 275.305
1 397 T033_MAN_JUN 8 7 477.364 463.364 432.169 385.904 327.874 283.673 244.639 477.364 450.497 408.168 362.128 320.727 281.116 241.645 477.364 452.47 423.445 390.663 352.769 310.454 265.192 477.364 462.934 441.255 409.895 368.96 320.499 268.55 477.364 475.045 456.97 424.551 378.8 324.466 263.669 477.364 486.082 477.986 454.752 417.471 368.217 310.389 477.364 488.526 484.122 466.541 436.516 394.473 343.38 477.364 479.116 462.599 432.563 396.79 355.961 310.381
1 398 T033_MAN_JUL 8 7 502.195 482.891 445.054 391.264 327.561 285.216 246.959 502.195 468.073 419.019 369.764 326.348 286.391 246.64 502.195 472.934 442.495 408.602 368.87 324.605 277.881 502.195 487.453 466.59 435.31 393.072 341.767 286.103 502.195 503.207 486.541 454.636 407.168 349.057 283.201 502.195 516.473 512.483 491.14 453.688 402.149 340.134 502.195 517.178 515.793 500.125 470.61 427.624 373.86 502.195 503.204 484.531 452.9 416.206 374.378 328.344
1 399 T033_MAN_AUG 8 7 435.681 404.373 357.928 299.513 251.646 222.26 194.385 435.681 395.867 348.716 306.867 271.711 238 204.742 435.681 409.766 382.385 352.982 319.893 282.223 242.102 435.681

431.116 417.58 393.482 358.427 314.86 266.313 435.681 449.779 447.456 427.608 391.116 341.49 281.99
435.681 459.079 464.585 452.484 424.83 382.478 328.395 435.681 449.703 448.93 436.687 412.15 375.715
329.367 435.681 426.984 402.776 372.667 340.249 305.027 266.072
1 400 T033_MAN_SEP 8 7 290.469 251.886 204.311 166.016 151.712 136.493 120.166 290.469 254.462
221.019 194.583 172.451 151.382 130.775 290.469 275.872 261.715 244.673 223.62 198.397 170.154
290.469 299.968 301.742 292.953 275.164 247.188 212.793 290.469 317.416 330.891 329.975 314.731
286.294 246.76 290.469 317.265 330.864 330.725 318.129 292.305 256.784 290.469 298.099 297.16 289.51
274.534 251.77 221.764 290.469 270.929 245.652 223.307 202.832 181.825 159.772
1 401 T033_MAN_OCT 8 7 173.229 134.448 103.031 96.3267 88.8437 80.1296 70.7782 173.229 140.209
118.304 105.144 94.3674 83.8863 73.0721 173.229 161.088 151.848 142.107 129.855 116.062 99.8155
173.229 186.501 193.104 193.284 185.509 169.914 149.144 173.229 205.348 228.297 240.51 241.157
230.193 208.365 173.229 203.735 225.179 236.332 236.612 225.59 204.424 173.229 182.259 186.183
185.161 178.322 166.189 147.989 173.229 153.565 134.824 121.561 110.111 98.4773 85.9193
1 402 T033_MAN_NOV 8 7 77.6249 57.7374 53.4279 50.2 45.9932 41.0944 35.8373 77.6249 62.0947
54.5077 50.4409 46.118 41.1153 35.8373 77.6249 73.216 69.2315 64.4006 58.9888 52.2216 44.6604 77.6249
87.4255 93.7786 96.1896 94.4878 88.789 79.5149 77.6249 96.0697 110.41 119.668 123.213 120.804 112.604
77.6249 93.0751 104.625 111.486 113.193 109.628 101.034 77.6249 80.1959 80.3273 78.4561 74.4164
68.3642 60.0954 77.6249 65.3105 56.8498 51.8748 47.029 41.7072 36.1469
1 403 T033_MAN_DEC 8 7 53.5968 36.5884 35.2739 33.1827 30.4574 27.2838 23.8781 53.5968 39.2954
35.3038 33.1827 30.4574 27.2838 23.8781 53.5968 49.4116 46.935 43.803 40.5059 36.1137 30.8986 53.5968
64.2143 72.083 76.6666 77.6528 74.9742 68.8136 53.5968 74.3617 91.6862 104.39 111.606 112.845 108.02
53.5968 71.7521 86.6448 97.2601 102.874 103.105 97.9371 53.5968 57.9141 60.4429 61.1439 60.2514
56.8825 51.492 53.5968 41.6134 36.0438 33.2939 30.4574 27.2838 23.8781
1 404 T033_NEW_JAN 8 7 62.5608 41.5841 39.7269 37.4092 34.3886 30.8712 27.0965 62.5608 45.1393
40.0369 37.4215 34.3886 30.8712 27.0965 62.5608 56.9803 53.2956 49.4936 44.9464 40.1347 34.4364
62.5608 74.5226 83.2524 88.2579 89.0966 85.7101 78.3291 62.5608 85.7873 105.014 118.931 126.59
127.468 121.506 62.5608 82.7384 99.124 110.601 116.388 116.089 109.726 62.5608 67.1618 69.4401
69.7675 67.7339 62.9374 56.8822 62.5608 48.6653 41.2404 38.0207 34.7897 31.2583 27.4432
1 405 T033_NEW_FEB 8 7 130.86 90.997 74.3984 70.2914 64.939 58.706 52.0171 130.86 100.084 82.5214
73.4854 66.5146 59.6521 52.6253 130.86 122.548 116.386 109.472 100.488 89.9353 77.672 130.86 149.323
161.479 166.548 164.875 155.582 139.231 130.86 166.617 194.563 212.796 220.072 215.895 200.55 130.86
161.544 184.764 198.938 203.099 197.193 181.559 130.86 136.952 139.148 137.986 133.465 123.792 111.48
130.86 107.596 90.2472 81.2436 73.6915 65.8584 58.1082
1 406 T033_NEW_MAR 8 7 222.438 190.816 154.727 139.234 127.667 114.26 99.8996 222.438 193.992
170.219 152.44 136.531 120.435 104.326 222.438 211.199 200.341 187.505 171.675 152.461 131.516
222.438 230.85 233.244 227.554 214.581 194.43 168.254 222.438 245.713 259.051 261.543 253.019 234.061
205.96 222.438 244.999 257.685 260.181 251.944 233.987 207.629 222.438 228.712 228.895 223.619
211.742 194.305 171.548 222.438 206.274 188.083 172.056 156.637 139.918 121.566
1 407 T033_NEW_APR 8 7 329.273 301.667 263.45 218.25 192.033 170.839 149.236 329.273 302.01
269.374 239.08 212.144 186.128 160.001 329.273 316.424 299.388 278.081 252.105 221.995 189.676
329.273 333.128 327.91 312.77 287.939 254.397 216.004 329.273 344.501 346.556 334.941 310.468 275.446
231.768 329.273 344.669 347.195 337.225 315.988 283.823 243.616 329.273 332.312 325.869 311.388
289.137 260.034 225.393 329.273 314.46 290.048 263.42 237.363 210.69 182.901
1 408 T033_NEW_MAY 8 7 450.929 427.619 387.927 334.558 278.734 242.492 209.203 450.929 421.314
378.753 337.42 300.404 263.388 226.386 450.929 430.481 407.754 381.315 348.853 309.849 265.905
450.929 444.907 431.018 406.763 371.703 327.15 276.176 450.929 456.31 445.637 418.645 377.043 325.418
264.546 450.929 462.703 458.43 439.798 406.923 361.358 306.567 450.929 457.123 450.737 434.314
407.332 369.927 323.954 450.929 442.438 417.371 385.594 353.291 317.867 278.09
1 409 T033_NEW_JUN 8 7 453.265 441.96 414.684 373.295 321.442 279.987 241.246 453.265 430.629
393.248 352.562 314.992 276.783 237.206 453.265 432.166 407.244 378.712 344.026 304.124 260.749
453.265 440.63 421.297 393.146 355.916 311.128 262.063 453.265 449.964 432.688 402.113 359.154 308.85
252.582 453.265 459.159 449.574 426.571 391.202 345.384 292.322 453.265 462.134 456.693 440.018
412.498 374.342 327.359 453.265 455.01 439.894 412.669 381.453 344.022 301.147
1 410 T033_NEW_JUL 8 7 494.716 473.698 435.083 381.503 320.532 279.432 241.654 494.716 462.392
414.523 365.364 322.796 282.088 242.129 494.716 467.986 436.641 401.931 362.148 317.984 271.116
494.716 482.72 461.936 430.315 388.262 337.93 283.501 494.716 497.8 483.174 452.914 407.227 350.814
286.1 494.716 508.01 503.101 480.961 442.814 391.294 330.255 494.716 505.972 500.643 480.712 448.296
404.278 351.299 494.716 491.742 470.054 435.556 397.042 354.853 308.771
1 411 T033_NEW_AUG 8 7 405.16 374.219 329.599 274.406 234.363 207.77 181.584 405.16 374.326
334.571 295.737 261.022 228.035 195.438 405.16 390.541 370.385 344.42 312.541 275.266 235.272 405.16
408.989 402.083 382.984 351.784 310.409 262.799 405.16 420.369 419.902 403.011 370.567 325.512

```
270.634 405.16 420.678 420.333 404.964 376.579 335.762 285.819 405.16 407.362 397.454 378.814 351.377
315.899 274.04 405.16 388.064 357.85 324.917 293.265 261.113 227.245
 1 412 T033_NEW_SEP 8 7 250.936 220.129 181.945 155.398 142.198 127.336 111.381 250.936 221.072
194.029 173.261 154.702 135.882 117.246 250.936 237.472 224.358 209.206 191.357 170.203 146.334
250.936 257.015 256.69 247.77 231.733 207.572 179.004 250.936 272.212 282.505 281.115 268.136 244.55
211.997 250.936 273.368 284.821 284.73 274.003 252.2 222.333 250.936 258.617 258.638 252.721 240.163
220.486 194.333 250.936 236.543 217.615 200.243 182.189 162.372 141.798
 1 413 T033_NEW_OCT 8 7 169.756 134.458 107.045 100.73 92.6285 83.1944 73.0703 169.756 140.556
121.561 109.131 97.9534 86.8339 75.2146 169.756 159.564 152.061 143.624 132.238 118.651 102.56
169.756 182.165 188.551 188.68 181.216 166.382 146.826 169.756 198.936 219.539 230.161 230.077
219.293 198.545 169.756 197.215 216.214 225.667 225.159 214.287 194.037 169.756 177.549 181.006
180.04 173.772 162.294 145.127 169.756 151.575 135.215 123.659 112.749 101.064 88.2015
 1 414 T033_NEW_NOV 8 7 74.6208 52.104 48.4535 45.598 41.8766 37.543 32.8925 74.6208 57.3015
49.7615 45.8781 42.0085 37.5459 32.8925 74.6208 70.4009 67.3976 63.4315 58.8668 52.6123 45.4873
74.6208 87.1697 96.0197 100.568 100.504 95.8322 86.9809 74.6208 97.1268 115.255 127.771 133.821
132.992 125.343 74.6208 93.0673 107.413 116.68 120.237 117.842 109.658 74.6208 77.3692 78.0209
77.1455 74.143 69.1052 61.5996 74.6208 59.8795 51.6329 47.2429 42.9172 38.0652 33.002
 1 415 T033_NEW_DEC 8 7 42.894 30.0514 28.9603 27.2245 24.9624 22.3281 19.5012 42.894 31.1553
28.9603 27.2245 24.9624 22.3281 19.5012 42.894 39.7093 37.1658 34.2384 31.2836 27.5258 23.2563 42.894
52.4044 59.6726 64.2031 65.6873 64.0239 59.3264 42.894 59.0616 72.5332 82.3908 87.9625 88.8687
85.0477 42.894 55.7811 66.1957 73.4282 76.9856 76.6256 72.3727 42.894 44.4845 44.3725 43.1925 40.9074
37.1634 32.4814 42.894 32.3454 28.9603 27.2245 24.9624 22.3281 19.5012
 1 416 T033_NOR_JAN 8 7 72.6876 48.0845 44.5481 41.9923 38.6615 34.7827 30.6202 72.6876 54.6176
45.8926 42.1063 38.6615 34.7827 30.6202 72.6876 69.7029 67.2207 64.1079 59.2291 53.5616 46.368
72.6876 87.7985 99.0128 105.566 107.012 103.252 94.5426 72.6876 96.7177 116.243 129.934 136.857
136.539 129.004 72.6876 90.7208 104.658 113.55 116.79 114.158 105.833 72.6876 73.3205 72.5881 70.4421
67.0393 61.1546 54.3517 72.6876 55.7304 46.6449 42.5574 38.6796 34.7827 30.6202
 1 417 T033_NOR_FEB 8 7 141.019 101.812 77.5271 73.3266 67.8523 61.4773 54.6361 141.019 112.645
92.1823 80.407 71.7718 63.8446 56.0934 141.019 136.465 132.045 125.435 116.375 104.601 90.9682
141.019 161.754 175.491 181.06 179.222 169.093 151.164 141.019 174.642 200.087 215.62 220.182 213.462
195.919 141.019 166.188 183.754 192.522 191.893 182.357 164.71 141.019 141.379 139.136 134.966
127.873 116.711 103.293 141.019 115.172 95.9648 84.9284 76.6176 68.2332 59.9764
 1 418 T033_NOR_MAR 8 7 251.706 213.902 170.303 144.677 132.936 119.581 105.252 251.706 221.841
194.42 172.289 153.024 134.481 116.293 251.706 244.242 235.401 223.036 206.133 184.295 159.326
251.706 266.536 273.235 269.637 256.376 233.513 202.844 251.706 279.528 295.471 298.449 288.258
265.594 232.034 251.706 274.029 285.279 285.478 274.186 252.32 221.825 251.706 252.873 248.501 239.84
225.58 206.295 181.789 251.706 227.836 202.411 183.126 165.792 147.966 129.08
 1 419 T033_NOR_APR 8 7 398.682 355.681 298.876 233.334 197.834 176.343 155.727 398.682 361.984
319.116 279.823 246.141 215.299 184.992 398.682 386.873 370.475 348.317 319.179 283.501 243.252
398.682 411.525 411.991 398.779 371.611 331.058 281.925 398.682 424.948 432.689 421.377 391.999
347.254 289.54 398.682 419.824 424.697 413.636 388.406 348.778 298.852 398.682 397.71 387.072 369.351
343.824 310.525 270.462 398.682 370.823 332.402 296.44 265.926 236.529 206.546
 1 420 T033_NOR_MAY 8 7 524.707 491.143 437.56 367.61 294.359 255.112 221.608 524.707 487.827
436.224 384.214 338.356 295.605 253.863 524.707 504.335 479.408 447.78 408.511 361.896 309.955
524.707 523.895 511.34 484.788 443.233 388.448 324.929 524.707 536.202 527.595 496.805 447.722 384.1
308.196 524.707 539.683 535.035 512.917 473.914 419.786 354.729 524.707 528.389 517.206 495.678
463.607 419.723 366.012 524.707 507.967 471.263 428.015 387.475 346.423 303.341
 1 421 T033_NOR_JUN 8 7 463.712 448.687 417.025 370.885 313.423 267.712 228.57 463.712 451.707
423.158 386.656 347.101 304.484 260.111 463.712 457.944 444.077 421.273 387.859 345.321 295.908
463.712 462.004 450.122 425.975 389.453 342.471 288.146 463.712 461.078 442.092 408.8 363.976 310.803
252.508 463.712 458.057 439.103 410.127 370.912 323.861 272.378 463.712 453.177 432.349 405.285
372.446 333.249 289.318 463.712 449.295 418.201 379.06 342.629 306.077 266.56
 1 422 T033_NOR_JUL 8 7 541.516 512.295 462.56 395.701 318.923 272.804 235.463 541.516 511.908
463.745 410.714 362.246 316.932 271.808 541.516 524.803 499.744 466.83 425.245 376.343 322.4 541.516
539.806 524.858 494.94 450.124 392.843 328.091 541.516 548.231 533.407 498.621 446.122 380.451
304.885 541.516 548.619 537.009 508.723 465.289 408.597 342.726 541.516 538.42 520.666 492.78 455.643
408.435 353.271 541.516 523.208 484.15 436.125 391.391 346.527 300.831
 1 423 T033_NOR_AUG 8 7 467.669 426.367 368.097 296.831 242.08 213.4 187.528 467.669 425.566
374.079 326.713 287.006 251.034 216.352 467.669 446.989 422.515 393.507 358.134 316.694 271.731
467.669 471.926 464.326 443.071 407.345 358.743 302.797 467.669 489.188 490.443 470.643 431.319
376.622 308.889 467.669 490.397 494.389 479.823 448.986 401.744 341.867 467.669 472.671 464.419
```

446.082 416.459 376.462 327.874 467.669 445.842 408.198 367.904 330.732 294.492 256.682
1 424 T033_NOR_SEP 8 7 298.072 255.234 203.155 162.855 148.617 134.161 118.771 298.072 259.439 223.426 196.484 175.609 155.366 135.158 298.072 283.435 268.384 250.729 229.643 204.936 177.019 298.072 310.192 313.401 305.047 286.604 256.86 221.126 298.072 328.663 344.921 345.74 331.062 302.09 260.625 298.072 326.654 342.28 343.567 331.904 305.912 269.25 298.072 304.802 304.195 296.645 280.952 256.688 224.8 298.072 274.847 246.816 222.007 199.08 176.788 154.607
1 425 T033_NOR_OCT 8 7 189.724 147.791 113.067 105.398 97.2146 87.6847 77.458 189.724 156.786 132.766 117.535 105.253 93.3159 81.0342 189.724 180.957 173.343 163.967 151.21 135.972 117.691 189.724 207.586 217.766 220.021 212.87 196.492 173.832 189.724 224.289 248.849 261.728 262.05 249.792 225.79 189.724 218.795 238.234 246.97 245.053 231.715 208.137 189.724 194.228 194.735 190.886 181.958 168.172 149.11 189.724 164.983 143.481 129.092 116.751 104.223 91.1614
1 426 T033_NOR_NOV 8 7 92.4912 63.4394 55.9379 52.7494 48.5941 43.755 38.5621 92.4912 70.6329 59.406 53.8953 49.0942 43.7738 38.5621 92.4912 87.5949 83.3575 78.2084 72.5303 65.0224 56.3864 92.4912 107.94 118.66 123.923 123.367 117.033 105.572 92.4912 118.963 139.956 154.038 160.252 158.172 147.941 92.4912 113.403 129.214 138.848 141.647 137.422 126.459 92.4912 94.5187 94.3643 92.4417 87.6908 80.9318 71.3118 92.4912 74.0782 62.0069 55.5011 50.2168 44.6565 38.7837
1 427 T033_NOR_DEC 8 7 58.8024 36.6018 35.3369 33.329 30.7123 27.6651 24.395 58.8024 41.9053 35.41 33.329 30.7123 27.6651 24.395 58.8024 57.768 56.604 54.4112 51.472 46.7441 40.5695 58.8024 75.5081 88.7304 97.5685 101.42 100.022 93.4705 58.8024 83.1596 103.512 118.473 127.022 128.578 123.034 58.8024 75.8562 89.403 98.5197 102.585 101.322 94.8158 58.8024 57.8762 56.5431 54.2546 51.6393 47.2577 41.5208 58.8024 41.6135 35.6062 33.3505 30.7149 27.6651 24.395
1 428 T033_NOT_JAN 8 7 68.7811 46.431 43.777 41.2203 37.8884 34.0083 29.8444 68.7811 51.54 44.5385 41.2719 37.8884 34.0083 29.8444 68.7811 64.7703 61.5383 57.9578 53.1769 47.6289 40.8896 68.7811 81.7326 91.1543 96.5198 97.3515 93.5828 85.4704 68.7811 91.6449 110.297 123.466 130.256 130.202 123.309 68.7811 87.3867 102.071 111.833 116.008 114.31 106.857 68.7811 71.4525 72.168 71.0482 68.3086 62.9478 56.337 68.7811 53.7889 45.6938 42.009 38.435 34.5218 30.3038
1 429 T033_NOT_FEB 8 7 142.415 97.1636 72.4444 68.7012 63.823 58.1422 52.0459 142.415 108.692 86.8279 75.4657 67.2058 59.9907 53.1851 142.415 135.454 129.769 122.616 113.605 102.086 88.3622 142.415 165.009 180.671 187.762 186.775 176.742 158.211 142.415 182.375 213.452 233.53 241.24 236.056 218.333 142.415 174.867 198.948 213.017 216.117 208.388 190.464 142.415 146.558 146.823 145.096 139.748 129.39 116.216 142.415 114.338 93.4133 82.9858 75.3286 67.7889 60.3236
1 430 T033_NOT_MAR 8 7 253.737 213.623 167.04 141.954 130.704 117.805 104.01 253.737 217.802 188.001 165.619 146.904 129.153 111.942 253.737 240.086 228.055 213.99 196.379 174.617 150.459 253.737 264.84 269.262 263.95 249.653 226.24 195.447 253.737 283.646 301.314 305.536 296.023 273.425 239.284 253.737 283.179 300.416 305.27 297.108 276.541 245.412 253.737 262.543 264.03 259.396 247.703 229.302 203.939 253.737 233.741 210.403 191.785 174.872 157.015 137.61
1 431 T033_NOT_APR 8 7 331.301 303.111 263.72 216.103 185.664 164.798 144.373 331.301 301.207 268.298 238.258 210.62 184.331 158.106 331.301 316.12 300.697 281.896 257.185 227.391 194.472 331.301 333.424 328.415 314.418 290.927 258.309 219.822 331.301 346.574 348.209 335.569 309.742 273.328 227.945 331.301 349.53 354.212 345.89 325.878 294.181 253.886 331.301 338.251 335.504 325.299 306.862 280.11 246.353 331.301 318.941 296.516 272.542 248.973 224.535 197.413
1 432 T033_NOT_MAY 8 7 484.793 457.871 412.538 351.885 285.838 246.574 213.426 484.793 451.207 404.188 357.937 318.141 279.419 240.402 484.793 462.841 438.708 409.951 375.727 334.676 288.117 484.793 479.26 465.125 438.889 400.431 351.416 295.558 484.793 491.677 480.839 450.661 404.694 346.667 278.337 484.793 498.461 494.351 474.991 440.27 391.502 332.293 484.793 491.712 485.338 469.407 442.193 402.902 353.16 484.793 474.695 446.017 410.972 375.921 337.785 295.652
1 433 T033_NOT_JUN 8 7 510.59 490.546 451.329 395.611 327.223 279.355 240.872 510.59 485.024 441.397 391.365 344.869 300.361 256.123 510.59 491.812 465.647 433.047 392.69 346.003 294.967 510.59 502.912 484.649 453.661 410.422 357.231 297.716 510.59 511.138 493.162 457.918 407.823 346.946 278.641 510.59 515.618 502.716 474.394 431.706 377.182 315.226 510.59 512.179 498.086 472.463 436.586 392.26 340.783 510.59 501.794 473.058 431.567 390.077 348.407 303.406
1 434 T033_NOT_JUL 8 7 576.94 542.155 484.561 408.083 321.361 274.128 237.659 576.94 535.993 475.063 411.918 357.687 310.101 265.592 576.94 549.667 516.795 478.093 431.613 379.386 322.897 576.94 570.591 552.127 518.822 469.919 407.916 338.169 576.94 586.545 572.193 535.702 478.546 406.937 323.413 576.94 592.707 585.726 558.247 512.207 449.553 375.153 576.94 582.371 568.813 541.864 502.704 451.93 392.086 576.94 561.318 521.904 470.31 421.893 374.941 326.456
1 435 T033_NOT_AUG 8 7 448.427 413.354 362.521 299.391 250.048 220.867 193.404 448.427 412.796 368.662 326.108 287.726 251.373 215.645 448.427 431.731 411.122 384.316 350.475 309.746 265.367 448.427 452.357 445.351 425.549 391.846 345.748 292.736 448.427 465.482 464.778 445.023 407.673 356.484 293.852 448.427 466.661 467.267 451.075 420.723 376.171 320.971 448.427 452.097 442.651 424.562 396.642 359.214 313.872 448.427 429.846 396.076 359.931 326.371 292.352 256.006

```
 1 436 T033_NOT_SEP 8 7 292.05 255.457 209.901 172.654 157.415 141.351 124.163 292.05 261.623
232.571 205.887 181.551 158.233 136.115 292.05 283.367 274.171 259.981 239.612 213.628 183.264 292.05
304.533 309.072 302.731 286.438 259.159 224.822 292.05 317.157 329.066 326.965 310.997 282.541
243.359 292.05 313.738 323.056 319.316 304.377 277.26 242.1 292.05 294.472 289.353 279.028 263.348
241.53 213.953 292.05 270.145 243.628 222.861 204.46 184.268 162.238
 1 437 T033_NOT_OCT 8 7 182.321 136.182 97.5417 90.8051 84.2137 76.5379 68.3006 182.321 145.842
120.032 104.66 93.1797 82.6679 72.1311 182.321 172.226 164.487 155.055 142.558 127.982 110.51 182.321
201.523 213.334 217.114 211.045 195.248 172.827 182.321 221.014 249.299 265.249 267.778 256.713
232.807 182.321 216.374 240.336 252.777 253.642 241.877 218.711 182.321 189.582 192.384 190.827
184.073 171.664 153.299 182.321 156.418 133.635 119.386 108.778 98.0394 86.7602
 1 438 T033_NOT_NOV 8 7 76.8871 54.8854 50.9225 47.8954 43.9504 39.3564 34.4264 76.8871 59.8522
52.1172 48.183 44.0633 39.3591 34.4264 76.8871 72.0066 68.3308 63.764 58.799 52.2447 45.0187 76.8871
87.7699 95.2196 98.5288 97.4697 92.1142 82.9169 76.8871 98.1027 114.979 126.366 131.487 129.993
121.987 76.8871 95.1602 109.294 118.326 121.641 119.012 110.618 76.8871 80.666 82.0306 81.5074
78.4039 73.1246 65.2094 76.8871 63.5602 55.0092 50.3317 45.7338 40.5988 35.279
 1 439 T033_NOT_DEC 8 7 60.2496 37.5736 36.2766 34.2134 31.5245 28.3932 25.033 60.2496 40.8432
36.2896 34.2134 31.5245 28.3932 25.033 60.2496 54.0607 50.3982 46.1779 42.2382 37.1711 31.3778
60.2496 73.8088 84.0439 90.2576 92.0263 89.2296 82.0579 60.2496 87.8154 111.103 128.525 138.894
141.503 136.176 60.2496 84.9715 105.609 120.755 129.377 130.889 125.187 60.2496 66.9429 71.0094
72.4798 71.9581 68.259 61.8749 60.2496 44.6743 37.3785 34.4408 31.5254 28.3932 25.033
 1 440 T033_PLY_JAN 8 7 92.5697 58.0221 51.1661 48.3853 44.7613 40.5411 36.0122 92.5697 66.4246
54.0802 48.9359 44.7613 40.5411 36.0122 92.5697 86.4162 81.8287 77.1864 70.855 63.5319 55.2042
92.5697 111.108 124.528 131.918 133.054 127.827 116.344 92.5697 125.697 152.712 171.775 181.585
181.475 171.452 92.5697 119.752 141.227 155.532 161.692 159.287 148.481 92.5697 96.7546 98.4153
97.7122 94.4406 87.546 78.8193 92.5697 70.9182 57.1074 51.1862 46.6602 42.1752 37.272
 1 441 T033_PLY_FEB 8 7 132.665 101.308 78.1254 73.4544 67.7624 61.134 54.0209 132.665 109.599
91.7195 80.5458 71.7255 63.3409 55.4561 132.665 128.337 123.302 115.858 106.754 94.8285 82.356
132.665 148.012 157.862 160.717 156.333 145.952 129.305 132.665 158.986 178.154 188.863 190.382
182.608 166.072 132.665 153.049 166.685 172.643 170.561 161.218 144.569 132.665 133.604 131.244
126.669 119.695 109.004 96.3301 132.665 112.217 94.9354 84.1933 75.994 68.0855 59.9542
 1 442 T033_PLY_MAR 8 7 250.754 211.452 166.011 141.147 129.908 117.036 103.286 250.754 215.859
185.436 162.763 144.505 126.955 109.453 250.754 237.295 224.219 208.359 189.484 167.702 144.794
250.754 262.046 265.204 259.566 243.654 220.08 188.894 250.754 280.005 297.213 301.206 291.711
269.374 235.72 250.754 278.814 294.951 298.728 289.277 268.823 237.284 250.754 258.303 258.248
251.925 238.79 218.694 193.141 250.754 230.38 206.267 185.548 166.729 148.352 129.436
 1 443 T033_PLY_APR 8 7 363.653 327.484 278.399 219.778 180.105 160.869 142.476 363.653 330.224
289.264 251.429 219.015 189.451 162.102 363.653 350.282 330.943 307.813 280.129 247.482 211.633
363.653 371.779 368.115 352.552 324.69 286.936 242.719 363.653 384.539 389.409 377.142 348.576
306.044 252.509 363.653 382.716 386.289 374.955 349.714 312.557 265.47 363.653 365.499 356.479
339.421 315.21 284.475 247.785 363.653 342.441 309.866 276.356 247.409 219.285 190.477
 1 444 T033_PLY_MAY 8 7 512.529 479.351 425.879 355.758 277.716 237.219 207.063 512.529 468.658
410.889 352.406 303.775 263.582 226.729 512.529 482.192 448.705 410.25 367.438 320.877 273.167
512.529 504.531 485.7 453.697 408.271 351.983 290.663 512.529 523.199 513.773 482.672 431.951 367.462
291.679 512.529 534.411 535.136 515.605 477.346 422.315 354.792 512.529 526.358 522.248 503.363
470.838 425.413 369.66 512.529 503.425 472.918 430.275 386.582 342.804 299.061
 1 445 T033_PLY_JUN 8 7 498.141 482.561 446.462 392.305 323.779 270.584 234.662 498.141 463.748
412.64 357.562 311.96 271.107 232.775 498.141 466.526 433.083 398.222 359.24 316.19 269.983 498.141
479.667 455.707 421.378 376.807 324.694 268.766 498.141 495.091 473.926 437.411 384.844 322.501
253.168 498.141 510.955 505.107 481.667 441.513 387.543 323.971 498.141 515.942 516.927 502.264
474.44 433.225 380.448 498.141 504.181 488.229 456.402 419.555 379.562 333.791
 1 446 T033_PLY_JUL 8 7 524.932 502.583 460.22 400.729 328.408 278.386 239.809 524.932 490.196
439.063 385.356 338.439 295.976 255.162 524.932 495.908 462.811 426.889 385.392 339.29 290.436
524.932 510.548 487.488 452.843 406.881 352.468 293.581 524.932 526.252 507.21 472.192 419.841 355.79
284.552 524.932 538.252 532.921 509.208 468.34 413.438 347.764 524.932 536.747 531.973 512.922
479.846 433.419 376.652 524.932 521.753 497.574 460.064 417.707 371.584 322.907
 1 447 T033_PLY_AUG 8 7 471.964 435.253 381.007 312.922 250.481 219.222 192.36 471.964 432.134
383.913 337.481 296.582 258.491 221.838 471.964 451.642 429.376 401.354 366.537 325.326 279.29
471.964 473.652 465.334 443.372 406.764 358.578 302.354 471.964 488.638 486.234 463.553 421.408
363.387 295.491 471.964 492.911 494.863 479.164 446.815 400.191 341.479 471.964 479.121 473.134
457.394 430.448 392.619 345.357 471.964 455.162 421.832 384.747 349.713 314.337 276.146
 1 448 T033_PLY_SEP 8 7 322.285 280.77 228.76 177.924 159.625 143.777 127.018 322.285 280.951
```

242.541 210.986 185.211 162.349 139.98 322.285 303.051 285.596 265.067 240.784 212.348 181.785 322.285 329.539 328.558 317.001 294.94 262.618 223.854 322.285 350.471 363.409 360.217 341.113 307.397 261.669 322.285 353.3 369.366 369.967 355.478 326.602 285.561 322.285 334.215 335.39 327.787 311.587 286.399 253.015 322.285 304.014 277.08 251.308 227.932 203.938 179.008
1 449 T033_PLY_OCT 8 7 187.043 146.448 112.248 104.651 96.4935 86.9944 76.8006 187.043 152.774 129.078 113.206 101.142 89.789 78.2295 187.043 174.567 164.919 153.907 140.212 124.58 106.078 187.043 200.844 207.53 206.995 198.597 181.898 158.688 187.043 220.207 243.598 255.622 255.46 243.123 219.451 187.043 218.458 240.22 250.99 250.013 237.231 214.124 187.043 196.124 199.068 196.726 188.114 175.028 155.612 187.043 166.29 145.561 131.155 118.896 106.551 92.8454
1 450 T033_PLY_NOV 8 7 95.9762 71.5346 62.1482 58.4902 53.723 48.1715 42.214 95.9762 77.357 65.4052 59.1862 54.0315 48.3015 42.214 95.9762 91.3555 86.6005 80.9209 73.3941 64.9604 54.9766 95.9762 107.828 115.399 118.312 116.04 108.736 96.899 95.9762 117.584 134.056 144.269 147.528 143.609 132.78 95.9762 113.61 126.379 133.412 134.23 128.777 117.425 95.9762 98.2331 97.3779 94.3467 88.6902 80.9805 70.7749 95.9762 80.6774 68.6913 62.1738 56.5048 50.3767 44.0302
1 451 T033_PLY_DEC 8 7 72.072 47.3613 44.4761 41.9159 38.5794 34.694 30.5244 72.072 53.1135 44.9998 41.9159 38.5794 34.694 30.5244 72.072 67.4672 63.5376 59.4877 54.1316 48.4079 41.4656 72.072 85.8468 95.8515 101.404 102.127 97.9695 89.2161 72.072 95.9047 115.282 128.883 135.781 135.506 128.077 72.072 91.1724 106.14 115.954 119.947 117.845 109.793 72.072 74.4221 74.6421 72.7852 69.383 63.3659 56.0154 72.072 56.0859 46.4646 42.3211 38.5799 34.694 30.5244
1 452 T033_SOU_JAN 8 7 87.6557 59.4644 53.3205 50.2725 46.3003 41.6746 36.7106 87.6557 66.1786 55.4825 50.5566 46.3003 41.6746 36.7106 87.6557 82.161 77.7924 73.2236 66.8512 59.8714 51.4844 87.6557 102.163 112.209 117.309 117.128 111.468 100.713 87.6557 114.042 135.158 149.565 156.281 154.849 145.365 87.6557 109.447 126.282 137.012 140.908 137.702 127.614 87.6557 91.0707 91.7985 90.1833 86.7457 79.9227 71.3861 87.6557 70.1273 58.6286 52.9368 48.0102 43.1259 37.9547
1 453 T033_SOU_FEB 8 7 133.475 102.989 82.3936 77.5894 71.4154 64.2255 56.5099 133.475 110.393 93.7198 82.5571 74.022 65.7413 57.4348 133.475 127.837 122.588 115.217 105.84 94.0374 80.7996 133.475 147.298 155.518 156.999 152.686 142.067 125.618 133.475 158.995 177.53 187.818 189.158 181.458 165.243 133.475 154.365 168.585 175.168 173.665 164.448 148.278 133.475 135.965 134.911 131.065 123.987 113.33 100.538 133.475 114.848 98.6719 88.6351 80.328 71.5269 62.6892
1 454 T033_SOU_MAR 8 7 228.219 196.098 158.054 135.594 124.531 111.828 98.2071 228.219 200.705 176.388 156.347 138.56 121.561 104.724 228.219 218.724 208.447 195.898 179.546 159.297 136.76 228.219 238.311 241.259 235.21 221.414 200.131 172.469 228.219 251.48 264.294 265.79 255.865 235.195 205.196 228.219 249.196 259.883 260.248 250.044 230.39 202.889 228.219 232.076 229.384 222.093 209.125 191.278 168.596 228.219 210.086 188.542 170.627 154.383 137.4 119.328
1 455 T033_SOU_APR 8 7 388.928 349.343 296.106 232.966 194.164 172.61 152.167 388.928 348.463 304.05 264.775 231.883 202.322 174.166 388.928 369.102 347.926 323.462 293.846 259.138 221.442 388.928 393.797 388.39 371.571 342.779 302.552 255.954 388.928 412.18 417.542 404.604 374.475 330.005 273.523 388.928 414.493 423.184 415.506 392.484 354.382 304.529 388.928 397.21 393.627 380.444 357.247 324.755 284.448 388.928 370.074 338.842 305.537 275.508 245.837 214.695
1 456 T033_SOU_MAY 8 7 474.349 450.682 408.842 351.682 286.376 246.359 213.265 474.349 438.803 390.903 343.459 302.22 263.435 226.064 474.349 447.779 420.238 389.437 353.62 312.042 266.412 474.349 463.964 446.143 418.177 379.327 330.98 276.527 474.349 478.594 466.704 436.024 390.603 333.034 266.487 474.349 490.599 488.558 470.109 435.454 386.504 327.07 474.349 487.781 485.984 471.282 444.266 405.247 356.034 474.349 471.118 448.863 415.937 381.047 343.782 302.264
1 457 T033_SOU_JUN 8 7 526.712 504.643 462.589 403.417 331.16 277.231 237.913 526.712 496.522 447.56 391.37 340.932 296.196 254.334 526.712 501.795 468.448 429.777 385.212 336.695 286.253 526.712 514.774 491.048 454.904 407.297 350.912 290.132 526.712 527.294 508.093 470.511 417.759 353.152 281.67 526.712 535.414 524.938 497.445 453.662 396.208 329.468 526.712 532.144 520.549 494.92 456.084 406.127 348.751 526.712 519.398 491.094 448.273 400.451 352.546 303.384
1 458 T033_SOU_JUL 8 7 482.951 464.875 429.179 378.297 316.356 270.876 232.697 482.951 453.58 409.683 363.393 321.532 281.305 241.377 482.951 457.362 428.92 396.707 358.946 316.531 270.682 482.951 469.227 448.189 417.364 376.491 327.524 274.321 482.951 482.176 464.793 431.984 385.299 329.226 266.12 482.951 493.484 487.074 464.519 427.168 377.239 318.437 482.951 493.177 488.217 469.762 439.235 397.269 346.239 482.951 481.236 461.035 428.268 390.953 349.707 304.115
1 459 T033_SOU_AUG 8 7 437.731 406.784 360.321 301.507 248.118 218.199 190.748 437.731 398.758 352.997 309.832 273.058 238.364 204.617 437.731 412.977 386.6 356.461 322.489 284.15 243.069 437.731 433.254 419.777 395.57 359.846 314.266 264.664 437.731 451.192 447.869 426.331 388.013 337.367 276.267 437.731 460.047 464.34 451.491 422.731 379.321 324.784 437.731 450.832 449.289 435.862 410.491 373.763 327.492 437.731 428.718 403.894 371.935 338.302 302.676 263.889
1 460 T033_SOU_SEP 8 7 324.547 281.821 228.561 178.122 159.842 144.306 127.801 324.547 280.897 239.714 208.767 184.989 162.623 141.067 324.547 303.141 283.253 261.949 238.401 211.606 181.918

```
324.547 331.377 329.66 316.698 294.552 261.475 223.461 324.547 353.789 367.554 364.904 346.021
312.715 266.833 324.547 357.658 375.784 377.626 364.697 335.893 295.048 324.547 338.385 341.956
336.387 321.125 295.565 260.648 324.547 306.772 280.988 256.035 231.899 206.755 180.821
 1 461 T033_SOU_OCT 8 7 196.19 156.629 119.233 109.159 100.677 90.7988 80.1986 196.19 164.567
139.657 121.73 107.802 95.2283 82.6706 196.19 187.211 178.27 167.174 152.665 135.785 116.074 196.19
212.108 220.316 220.404 211.296 193.255 169.349 196.19 227.981 249.701 259.87 257.794 243.615 218.3
196.19 223.402 240.855 247.446 243.269 227.979 203.08 196.19 200.582 199.078 193.098 182.78 167.765
148.257 196.19 172.963 149.78 133.787 120.87 107.823 94.4536
 1 462 T033_SOU_NOV 8 7 97.9977 69.5846 60.6672 57.17 52.6122 47.3046 41.6088 97.9977 75.5869
63.7069 58.0213 52.9734 47.3428 41.6088 97.9977 91.0509 85.7208 79.953 73.1619 64.7894 55.6036
97.9977 110.618 119.206 122.616 120.506 113.019 100.973 97.9977 124.014 144.415 157.809 163.285
160.469 149.552 97.9977 121.044 138.677 149.695 153.348 149.385 138.078 97.9977 103.448 105.646
105.306 101.288 94.4589 84.0875 97.9977 81.8602 69.8834 63.2147 57.4553 51.324 44.787
 1 463 T033_SOU_DEC 8 7 71.064 44.3606 42.2995 39.9075 36.7902 33.16 29.2644 71.064 50.4189
42.7664 39.9075 36.7902 33.16 29.2644 71.064 65.7373 62.2322 57.9573 53.6123 48.015 41.2227 71.064
86.2261 97.5064 104.136 105.663 101.984 93.3493 71.064 98.8997 121.99 138.761 148.07 149.283 142.316
71.064 94.4332 113.361 126.558 133.125 132.614 125.059 71.064 75.443 77.6904 77.8171 75.9983 71.1802
63.6449 71.064 53.9285 44.4279 40.5215 36.8135 33.16 29.2644
 1 464 T033_SWI_JAN 8 7 79.0884 54.9538 50.5992 47.6371 43.7767 39.2813 34.457 79.0884 60.7242
52.043 47.8001 43.7767 39.2813 34.457 79.0884 74.6107 70.7628 66.6378 60.9573 54.7947 47.3548 79.0884
92.2008 101.378 106.005 105.799 100.731 91.1465 79.0884 101.653 119.639 131.82 137.365 135.898
127.517 79.0884 96.9968 110.643 119.098 121.784 118.519 109.526 79.0884 80.9589 80.7551 78.6144
74.8569 68.3467 60.4578 79.0884 63.5511 53.6375 48.6065 44.006 39.4628 34.6083
 1 465 T033_SWI_FEB 8 7 143.107 106.376 80.9759 76.4267 70.627 63.873 56.6251 143.107 116.561
96.0011 83.5629 74.5827 66.2436 58.0997 143.107 139.266 134.161 126.72 117.142 104.661 90.6438
143.107 162.988 175.696 180.29 177.27 166.369 147.989 143.107 174.051 196.992 210.368 213.266 205.489
187.568 143.107 165.268 180.025 186.372 183.877 173.19 155.107 143.107 141.876 137.989 131.894
122.982 110.677 96.8906 143.107 118.068 97.8782 85.892 76.8805 68.1871 59.8152
 1 466 T033_SWI_MAR 8 7 265.402 223.957 175.175 143.546 132.49 119.712 106.003 265.402 230.247
198.25 173.683 153.439 134.784 116.536 265.402 253.998 241.716 226.822 207.854 184.392 158.786
265.402 279.814 284.745 279.05 263.494 238.531 205.514 265.402 295.985 313.621 317.108 306.208
281.665 245.151 265.402 291.129 304.38 305.478 293.77 270.355 236.92 265.402 268.206 264.108 254.312
238.357 216.845 189.887 265.402 240.352 211.724 187.816 168.018 148.844 129.113
 1 467 T033_SWI_APR 8 7 340.452 308.667 265.198 213.056 179.681 159.478 139.943 340.452 307.436
270.015 236.907 208.905 183.121 157.864 340.452 323.329 304.118 282.58 256.678 226.812 194.349
340.452 343.12 336.616 320.678 295.032 260.347 220.438 340.452 358.385 361.245 348.836 322.315
284.005 235.99 340.452 360.532 367.195 359.853 339.089 305.515 261.843 340.452 347.156 344.407
331.887 310.464 281.339 245.349 340.452 325.289 300.251 271.66 244.646 217.298 188.585
 1 468 T033_SWI_MAY 8 7 479.221 453.631 410.247 352.023 286.997 247.054 212.985 479.221 449.786
406.093 360.339 318.772 280.655 241.894 479.221 460.475 436.79 406.851 369.547 327.09 281.375 479.221
474.959 459.909 433.075 393.838 343.879 286.912 479.221 485.348 473.539 442.927 397.702 340.478
274.11 479.221 489.282 483.096 461.931 425.898 376.815 318.309 479.221 481.915 472.135 451.705
420.845 379.26 328.317 479.221 466.856 436.789 398.699 360.044 319.005 276.884
 1 469 T033_SWI_JUN 8 7 461.329 445.285 412.435 365.018 306.265 258.903 221.213 461.329 439.415
401.634 357.697 316.465 276.673 237.044 461.329 442.783 417.309 387.102 350.532 308.843 263.873
461.329 451.276 431.588 401.687 362.084 314.691 262.836 461.329 459.474 441.247 408.095 362.783
308.276 248.625 461.329 465.345 453.756 428.766 390.943 342.499 287.136 461.329 463.62 452.214
429.889 397.247 355.743 307.303 461.329 455.311 431.804 396.598 357.918 317.453 274.162
 1 470 T033_SWI_JUL 8 7 574.081 545.469 494.726 425.31 342.609 290.396 250.07 574.081 532.898
472.781 410.588 358.636 313.44 270.235 574.081 540.778 501.944 460.535 414.42 364.26 311.51 574.081
559.675 533.742 495.377 444.747 384.333 319.187 574.081 578.613 560.747 522.31 464.9 394.472 314.099
574.081 591.197 586.902 562.222 517.995 456.611 382.836 574.081 586.869 581.241 559.12 521.488
469.619 406.722 574.081 567.748 538.228 494.147 445.782 394.709 341.237
 1 471 T033_SWI_AUG 8 7 502.578 458.348 395.574 318.534 253.769 222.961 195.942 502.578 457.372
400.486 347.405 303.787 265.15 228.888 502.578 479.423 450.847 417.462 378.586 334.043 286.379
502.578 506.154 495.881 471.079 430.949 377.412 317.166 502.578 525.252 525.586 503.128 459.506
399.264 325.189 502.578 526.645 530.522 514.282 479.867 427.952 362.519 502.578 507.776 498.4 477.364
443.966 399.458 346.116 502.578 479.078 438.176 392.979 350.196 309.408 268.015
 1 472 T033_SWI_SEP 8 7 323.381 280.232 226.612 175.53 158.859 143.508 127.185 323.381 286.319
248.92 218.363 193.055 169.603 146.926 323.381 311.358 297.749 280.309 257.985 230.326 198.388
323.381 337.569 341.368 332.846 313.335 281.515 242.636 323.381 353.116 367.411 365.292 346.904
```

313.78 267.986 323.381 348.511 360.226 357.091 340.827 310.359 269.934 323.381 326.042 321.277 309.607 290.801 264.554 231.547 323.381 297.189 265.842 238.343 214.524 190.757 166.523
1 473 T033_SWI_OCT 8 7 187.596 151.293 117.714 108.295 99.7083 89.7091 78.9786 187.596 159.896 137.199 120.644 107.48 94.9635 82.2219 187.596 181.261 174.143 164.359 150.895 135.015 116.184 187.596 203.817 212.216 212.927 204.828 188.153 165.483 187.596 216.561 236.149 245.027 242.588 229 205.188 187.596 210.38 224.21 228.397 223.126 208.032 184.554 187.596 188.957 186.054 179.391 168.644 153.728 134.693 187.596 164.704 143.504 128.192 115.302 102.461 89.3267
1 474 T033_SWI_NOV 8 7 88.5528 65.7021 58.2733 54.8188 50.3168 45.0742 39.4481 88.5528 71.9337 61.4539 55.774 50.6779 45.11 39.4481 88.5528 85.5218 82.3155 77.9303 71.9455 64.1518 55.145 88.5528 101.076 109.414 112.99 111.553 105.203 94.5356 88.5528 108.83 124.379 134.139 137.447 134.076 124.257 88.5528 103.725 114.517 120.193 120.366 115.025 104.534 88.5528 88.7911 87.0043 83.664 78.1382 71.3265 62.3813 88.5528 73.3082 63.0179 57.2251 51.6643 45.7856 39.6834
1 475 T033_SWI_DEC 8 7 67.8348 45.3856 43.3478 40.8116 37.5062 33.6571 29.5265 67.8348 50.3972 43.7298 40.8116 37.5062 33.6571 29.5265 67.8348 63.169 59.8176 56.012 51.6601 46.13 39.5174 67.8348 80.3496 89.401 94.3719 94.9238 91.0189 82.9235 67.8348 90.8117 109.612 122.955 129.931 130.064 123.346 67.8348 87.0496 102.344 112.676 117.342 116.023 108.81 67.8348 71.2671 72.6675 72.1106 69.7845 64.8808 57.822 67.8348 53.4015 45.2444 41.3669 37.589 33.6571 29.5265
1 476 T3_Eext_BEL_630 1 12 0 0 0.476129 0.932 5.00516 6.892 5.19871 2.01677 0.028 0 0 0
1 477 T3_Eext_BEL_730 1 12 0 0.154286 3.98323 5.252 10.9161 12.492 11.16 7.08387 2.504 0.0541935 0.056 0
1 478 T3_Eext_BEL_830 1 12 0.216774 3.09857 9.36774 11.368 16.4206 18.18 16.9123 13.7032 7.976 2.4929 2.232 0.127742
1 479 T3_Eext_BEL_930 1 12 3.17806 6.95143 14.4581 17.148 20.9884 22.704 22.6839 19.7806 14.876 6.68903 5.932 2.65935
1 480 T3_Eext_BEL_1030 1 12 5.1871 10.8429 18.1897 22.116 26.4 28.244 27.9832 25.2735 19.136 11.2955 8.676 4.71484
1 481 T3_Eext_BEL_1130 1 12 6.79355 13.8557 20.1561 26.924 29.7716 33.088 32.4813 28.3703 23.416 13.8774 10.588 6.72774
1 482 T3_Eext_BEL_1230 1 12 7.74194 14.01 22.2658 28.6 32.7755 34.536 36.929 30.511 26.16 15.9135 10.296 7.36645
1 483 T3_Eext_BEL_1330 1 12 7.50581 13.83 21.4026 29.972 33.5574 37.372 37.9935 32.0516 27.344 16.2194 8.652 6.52645
1 484 T3_Eext_BEL_1430 1 12 5.48129 11.3614 18.5652 27.636 31.56 36.624 35.2142 32.7097 25.436 15.5923 6.18 4.16129
1 485 T3_Eext_BEL_1530 1 12 3.56903 7.98 14.4116 26.832 30.5768 33.82 34.0374 29.0787 22.636 12.7084 3.484 1.50581
1 486 T3_Eext_BEL_1630 1 12 0.530323 4.06286 9.87484 22.088 27.2129 29.492 30.2516 25.2542 17.864 8.57032 0.28 0
1 487 T3_Eext_BEL_1730 1 12 0 0.57 4.6529 16.412 21.5381 23.552 24.4103 20.8684 12.308 4.07613 0 0
1 488 T3_Eext_BEL_1830 1 12 0 0 0.588387 10.232 16.0335 17.512 18.0271 13.9587 6.056 0.487742 0 0
1 489 T3_Eext_BEL_1930 1 12 0 0 0 4.476 9.61161 11.404 11.369 7.28516 1.184 0 0 0
1 490 T3_Eext_BIR_630 1 12 0 0 1.28903 1.8 5.9729 7.76 6.32129 2.89161 0.188 0 0 0
1 491 T3_Eext_BIR_730 1 12 0 0.698571 5.67871 6.664 12.4955 14.652 12.6348 8.64 4.016 0.367742 0.528 0
1 492 T3_Eext_BIR_830 1 12 1.27355 4.78286 10.5561 11.592 18.6232 20.024 17.4735 15.1665 10.14 4.32387 4.276 1.05677
1 493 T3_Eext_BIR_930 1 12 4.75355 9.28286 15.631 16.632 24.6581 26.268 24.1084 21.4181 16.084 9.74323 7.732 4.33548
1 494 T3_Eext_BIR_1030 1 12 8.33419 12.9214 18.6581 22.924 29.6748 30.08 28.4748 27.1045 22.284 13.9277 10.86 6.73161
1 495 T3_Eext_BIR_1130 1 12 10.6258 15.6729 20.7716 26.744 33.631 33.392 33.7781 30.4684 25.764 17.2413 12.464 8.50452
1 496 T3_Eext_BIR_1230 1 12 10.8658 16.7186 23.4542 29.676 34.9587 33.272 35.9729 32.6671 27.48 18.7587 11.088 8.15226
1 497 T3_Eext_BIR_1330 1 12 9.93677 14.9271 21.809 30.968 36.2748 35.436 36.3406 34.3471 28.092 19.1923 10.252 6.87484
1 498 T3_Eext_BIR_1430 1 12 6.79742 13.0286 19.5135 28.292 34.7535 35.596 33.3639 33.4335 26.84 17.6671 6.856 4.40903
1 499 T3_Eext_BIR_1530 1 12 3.44903 8.05714 14.8258 25.004 31.6606 31.428 31.7806 31.6335 22.42 13.0374 3.18 1.39355
1 500 T3_Eext_BIR_1630 1 12 0.329032 3.74143 9.58065 21.36 26.6168 27.076 27.2439 26.2374 17.768

```
8.06323 0.132 0
 1 501 T3_Eext_BIR_1730 1 12 0 0.205714 3.60774 15.292 19.7961 21.616 21.9987 19.2194 10.392 3.35613
0 0
 1 502 T3_Eext_BIR_1830 1 12 0 0 0.150968 8.736 13.2348 16.096 15.4839 12.2323 4.324 0.166452 0 0
 1 503 T3_Eext_BIR_1930 1 12 0 0 0 2.912 6.92516 9.372 8.90323 4.93548 0.448 0 0 0
 1 504 T3_Eext_CAR_630 1 12 0 0 0.913548 1.228 5.04774 6.812 5.45032 2.22968 0.072 0 0 0
 1 505 T3_Eext_CAR_730 1 12 0 0.587143 5.44258 6.08 10.1342 12.804 12.0039 8.27226 3.524 0.247742
0.452 0
 1 506 T3_Eext_CAR_830 1 12 1.2271 4.47429 10.7381 11.852 17.0748 18.12 17.5858 15.5613 9.632
4.11871 4.184 1.10323
 1 507 T3_Eext_CAR_930 1 12 4.52516 8.46857 15.24 18.068 22.7768 23.996 23.849 21.9948 15.16
9.92129 7.956 4.63355
 1 508 T3_Eext_CAR_1030 1 12 7.95097 11.9529 19.3277 23.296 27.3406 29.54 29.1948 27.1819 19.612
13.9355 11.444 7.61806
 1 509 T3_Eext_CAR_1130 1 12 10.8387 14.8457 21.9406 27.488 28.9006 33.584 33.4026 31.4981 22.884
17.6439 13.648 9.42968
 1 510 T3_Eext_CAR_1230 1 12 11.551 15.4414 22.3858 29.552 29.52 37.276 36.3406 34.2968 25.984
19.1923 13.332 9.63097
 1 511 T3_Eext_CAR_1330 1 12 10.289 15.2486 22.5871 29.16 32.6323 35.944 36.5884 35.6168 27.552
19.2426 11.508 8.21419
 1 512 T3_Eext_CAR_1430 1 12 7.7729 12.7457 19.0645 27.444 30.3794 35.092 37.0955 33.9832 25.72
17.2026 7.832 5.19097
 1 513 T3_Eext_CAR_1530 1 12 4.2271 9.63857 14.8103 24.464 28.409 30.02 34.4284 31.2813 21.792
13.7071 3.92 2.21419
 1 514 T3_Eext_CAR_1630 1 12 0.661935 4.41 9.81677 20.12 24.569 26.748 28.4981 26.3535 16.884
8.95355 0.316 0
 1 515 T3_Eext_CAR_1730 1 12 0 0.42 4.19226 15.756 18.1277 21.54 22.831 19.5213 11.308 4.15742 0 0
 1 516 T3_Eext_CAR_1830 1 12 0 0 0.24 8.88 12.2013 15.772 16.44 11.7058 5.088 0.305806 0 0
 1 517 T3_Eext_CAR_1930 1 12 0 0 0 3.316 6.57677 10.032 9.82452 5.10581 0.552 0 0 0
 1 518 T3_Eext_DUB_630 1 12 0 0 0.623226 0.568 4.02194 5.952 4.15355 1.14194 0.032 0 0 0
 1 519 T3_Eext_DUB_730 1 12 0 0.21 4.93548 4.836 11.1794 13.16 10.9665 6.6 1.78 0.0812903 0.136 0
 1 520 T3_Eext_DUB_830 1 12 0.336774 3.10286 11.1484 12.132 17.5974 19.84 17.969 13.5794 8.172
2.26065 2.456 0.259355
 1 521 T3_Eext_DUB_930 1 12 3.56516 8.58 15.7045 19.24 24.6658 28.264 25.5523 20.191 15.832 8.27613
7.104 3.12774
 1 522 T3_Eext_DUB_1030 1 12 7.49032 12.8486 20.0168 26.368 29.0439 34.724 31.5252 26.4968 21.484
14.6516 11.004 7.1729
 1 523 T3_Eext_DUB_1130 1 12 9.73161 16.7186 24.1742 30.256 34.3703 37.572 37.2774 30.6039 25.668
18.1161 14.028 9.12
 1 524 T3_Eext_DUB_1230 1 12 11.2916 16.8943 24.9406 34.124 38.3729 41.916 39.2903 33.5574 29.712
20.2181 13.248 9.19742
 1 525 T3_Eext_DUB_1330 1 12 10.2465 16.2557 24.6232 35.032 39.5652 38.98 39.6735 33.8052 30.696
19.0684 12.568 8.34194
 1 526 T3_Eext_DUB_1430 1 12 7.8 13.6243 22.711 34.48 37.0219 38.464 38.8568 32.4813 28.46 18.5187
9.092 5.71355
 1 527 T3_Eext_DUB_1530 1 12 3.85548 9.97286 17.5935 30.14 34.6065 33.336 36.991 30.5419 25.568
15.8787 4.236 1.87355
 1 528 T3_Eext_DUB_1630 1 12 0.576774 4.58571 11.3729 25.06 29.1639 30.412 31.111 26.2955 20.156
10.3123 0.568 0
 1 529 T3_Eext_DUB_1730 1 12 0 0.612857 4.89677 18.336 23.4116 23.576 26.0245 20.2297 13.264
4.42839 0 0
 1 530 T3_Eext_DUB_1830 1 12 0 0 0.650323 11.684 16.7032 18.856 19.44 13.2465 6.4 0.483871 0 0
 1 531 T3_Eext_DUB_1930 1 12 0 0 0 4.116 9.53032 12.164 12.0697 6.66581 0.92 0 0 0
 1 532 T3_Eext_EDI_630 1 12 0 0 0.886452 1.752 5.85677 8.216 6.72774 2.89548 0.168 0 0 0
 1 533 T3_Eext_EDI_730 1 12 0 0.308571 4.99742 6.612 11.8065 14.316 12.9097 8.52 3.44 0.170323 0.128
0
 1 534 T3_Eext_EDI_830 1 12 0.348387 3.54429 10.4013 12.36 17.4581 21.024 19.0413 14.1368 9.328 3.12
2.484 0.236129
 1 535 T3_Eext_EDI_930 1 12 3.19355 7.73143 15.3677 17.888 22.5135 26.056 25.0219 18.809 14.868
7.37806 5.756 2.70194
 1 536 T3_Eext_EDI_1030 1 12 5.38839 11.7814 19.6258 22.28 27.9058 29.7 29.9419 24.1471 20.188
```

```
11.2955 8.18 4.68774
 1 537 T3_Eext_EDI_1130 1 12 7.64516 13.9286 21.9445 26.872 31.3548 33.236 34.2735 25.9626 23.644
15.0968 9.976 6.54968
 1 538 T3_Eext_EDI_1230 1 12 8.47742 14.3314 22.8387 28.528 33.9948 35.204 37.4903 29.431 25.856
16.4942 9.956 6.79355
 1 539 T3_Eext_EDI_1330 1 12 7.66839 13.3629 21.5071 28.816 35.2877 35.788 37.711 30.6 25.976
16.7574 8.336 5.37677
 1 540 T3_Eext_EDI_1430 1 12 5.52774 10.9114 19.6916 27.712 33.7394 34.888 36.9135 28.2387 25.216
15.0619 5.44 3.85161
 1 541 T3_Eext_EDI_1530 1 12 2.52 7.25571 14.6477 24.44 30.24 31.268 34.6684 26.8645 22.296 11.5626
2.284 0.661935
 1 542 T3_Eext_EDI_1630 1 12 0.15871 3.29143 8.75613 20.548 26.8452 27.308 29.2026 22.2077 17.492
7.26194 0.068 0
 1 543 T3_Eext_EDI_1730 1 12 0 0.205714 3.61935 15.432 21.2942 22.408 23.7639 17.8839 11.696
2.97677 0 0
 1 544 T3_Eext_EDI_1830 1 12 0 0 0.263226 9.168 14.9845 17.136 17.3303 11.5626 5.344 0.20129 0 0
 1 545 T3_Eext_EDI_1930 1 12 0 0 0 3.788 8.27613 11.296 10.6839 5.86452 0.728 0 0 0
 1 546 T3_Eext_GLA_630 1 12 0 0 0.456774 0.932 5.02065 6.76 5.32645 1.98581 0.028 0 0 0
 1 547 T3_Eext_GLA_730 1 12 0 0.154286 4.38194 5.556 10.4245 12.664 11.3381 6.56516 2.472 0.0541935
0.06 0
 1 548 T3_Eext_GLA_830 1 12 0.216774 2.94857 9.75484 11.308 16.5329 18.544 17.2335 12.8129 8.088
2.40387 2.276 0.127742
 1 549 T3_Eext_GLA_930 1 12 3.1471 7.08 14.2529 16.88 21.5652 23.4 24.2361 18.3484 13.656 6.67742
5.74 2.63613
 1 550 T3_Eext_GLA_1030 1 12 5.58581 11.4557 18.0426 21.56 25.8619 30.04 29.04 23.9806 18.3 11.52
9.056 4.80774
 1 551 T3_Eext_GLA_1130 1 12 8.12129 14.1686 21.5419 25.42 30.0503 33.644 32.8839 27.9445 21.532
14.6594 10.716 6.83226
 1 552 T3_Eext_GLA_1230 1 12 9.05419 14.9871 21.9832 28.968 32.8761 33.772 36.4374 30.8013 23.34
16.6335 10.216 6.94839
 1 553 T3_Eext_GLA_1330 1 12 8.32645 14.0571 20.8529 30.42 33.6852 34.836 36.7626 32.4697 24.32
16.1768 9.344 6.13935
 1 554 T3_Eext_GLA_1430 1 12 6.00774 11.4814 19.2542 28.812 32.8065 34.636 34.7032 28.7923 22.404
15.0039 6.42 3.72774
 1 555 T3_Eext_GLA_1530 1 12 3.48774 8.18571 14.5161 27.184 31.409 33.32 32.4155 28.0219 20.772
12.5381 3.372 1.50581
 1 556 T3_Eext_GLA_1630 1 12 0.538065 4.11 9.52645 22.904 26.5006 29.968 28.9858 23.3419 16.18
8.21419 0.296 0
 1 557 T3_Eext_GLA_1730 1 12 0 0.57 4.91226 16.64 21.3639 25.224 23.4426 18.9329 12.116 3.94839 0 0
 1 558 T3_Eext_GLA_1830 1 12 0 0 0.576774 10.672 15.569 18.184 18.2129 12.6271 5.98 0.487742 0 0
 1 559 T3_Eext_GLA_1930 1 12 0 0 0 4.808 9.27484 12.244 11.6981 6.60774 1.192 0 0 0
 1 560 T3_Eext_JER_630 1 12 0 0 0.809032 0.912 4.61419 6.084 4.76516 1.64516 0.152 0 0 0
 1 561 T3_Eext_JER_730 1 12 0.00774194 0.767143 6.89806 6.572 12.8903 14.612 12.449 8.71742 3.544
0.39871 0.692 0
 1 562 T3_Eext_JER_830 1 12 1.54839 6.03429 14.6052 14.384 19.3084 21.368 21.5497 17.4813 11.28
4.84258 5.036 1.61419
 1 563 T3_Eext_JER_930 1 12 6.96 12.1414 20.8955 20.396 25.1768 26.268 25.7342 22.7574 19.092
12.2361 10.496 6.21677
 1 564 T3_Eext_JER_1030 1 12 11.5006 15.9771 23.791 26.828 31.0684 32.352 32.0942 26.1019 23.656
18.2168 14.488 10.0297
 1 565 T3_Eext_JER_1130 1 12 13.8426 18.2486 24.9368 27.784 31.9548 35.14 35.431 29.9961 27.316
20.849 15.94 11.7406
 1 566 T3_Eext_JER_1230 1 12 15.0658 18.3514 26.2529 26.868 33.5071 38.82 36.0968 33.2477 29.216
21.5148 16.428 12.0735
 1 567 T3_Eext_JER_1330 1 12 13.0297 16.7786 24.2439 27.44 34.0258 35.892 35.1871 34.0684 31.504
22.7071 13.756 10.3006
 1 568 T3_Eext_JER_1430 1 12 9.75484 14.2671 20.7445 27.972 32.4735 33.572 33.2245 32.0826 26.748
20.7252 9.46 7.04516
 1 569 T3_Eext_JER_1530 1 12 4.79226 10.4443 16.1574 25.664 30.0658 31.212 27.9871 29.2026 24.128
16.169 3.98 2.02065
 1 570 T3_Eext_JER_1630 1 12 0.789677 4.55143 10.7806 21.22 24.6039 26.936 25.0103 24.0232 18.676
```

```
11.0013 0.332 0.00774194
 1 571 T3_Eext_JER_1730 1 12 0 0.381429 3.54968 16.456 19.7497 23.66 21.6968 19.1148 12.2 3.83226 0
0
 1 572 T3_Eext_JER_1830 1 12 0 0 0.154839 9.708 13.2426 17.872 16.529 12.7858 5.012 0.290323 0 0
 1 573 T3_Eext_JER_1930 1 12 0 0 0 2.164 6.25935 9.384 9.27484 4.48645 0.396 0 0 0
 1 574 T3_Eext_LEE_630 1 12 0 0 1.30839 1.996 6.06968 8.276 7.06065 3.22839 0.252 0 0 0
 1 575 T3_Eext_LEE_730 1 12 0 0.69 5.57419 6.88 12.1742 14.708 12.7123 8.77548 3.992 0.406452 0.404
0
 1 576 T3_Eext_LEE_830 1 12 0.99871 4.24286 10.8581 13.428 19.1187 20.42 18.9252 14.6013 10.428
3.97935 3.688 0.774194
 1 577 T3_Eext_LEE_930 1 12 4.67613 8.22857 15.8361 18.196 22.7381 27.28 26.8065 20.7329 16.04 7.56
6.812 4.16903
 1 578 T3_Eext_LEE_1030 1 12 7.22323 11.3957 20.0516 24.02 27.4258 31.084 31.5561 26.2529 21.832
11.5587 9.868 6.31355
 1 579 T3_Eext_LEE_1130 1 12 8.87613 13.35 20.6206 26.14 32.0826 34.136 35.6748 30.7819 25.916
15.0619 11.468 8.05935
 1 580 T3_Eext_LEE_1230 1 12 8.91871 15.18 20.9729 28.828 33.3213 36.864 36.089 33.8981 27.572
16.9974 11.016 7.85419
 1 581 T3_Eext_LEE_1330 1 12 8.00129 14.2629 19.4206 30.136 32.9342 38.576 36.5574 33.4877 28.26
16.8735 9.72 6.48387
 1 582 T3_Eext_LEE_1430 1 12 5.64387 11.3914 16.7884 26.604 32.6942 35.908 36.6039 31.9239 25.96
15.6426 6.12 4.54065
 1 583 T3_Eext_LEE_1530 1 12 2.96129 7.35429 14.1445 24.048 29.5161 34.24 31.8542 29.04 22.252
11.4581 2.62 0.92129
 1 584 T3_Eext_LEE_1630 1 12 0.193548 3.33857 8.28774 20.084 24.6426 28.784 27.6387 23.4658 17.396
7.08387 0.06 0
 1 585 T3_Eext_LEE_1730 1 12 0 0.167143 3.06581 14.28 19.0994 22.988 23.009 17.5858 10.348 2.91484
0 0
 1 586 T3_Eext_LEE_1830 1 12 0 0 0.127742 8.176 13.2077 16.38 16.68 11.0516 4.368 0.154839 0 0
 1 587 T3_Eext_LEE_1930 1 12 0 0 0 3.028 6.94452 10.144 9.73935 5.0671 0.44 0 0 0
 1 588 T3_Eext_LON_630 1 12 0 0 1.76903 2.18 5.87226 8.412 6.90194 3.18581 0.336 0 0 0
 1 589 T3_Eext_LON_730 1 12 0 1.21286 6.32129 7.6 12.5303 14.584 12.9368 8.82194 4.604 0.673548
0.912 0
 1 590 T3_Eext_LON_830 1 12 2.0671 5.48571 11.3381 14.024 19.0452 21.004 19.6297 15.24 10.032
4.95097 4.572 1.83097
 1 591 T3_Eext_LON_930 1 12 5.11355 10.2729 16.0645 18.988 24.1587 28.264 24.12 21.3677 15.488
9.89032 8.032 4.14194
 1 592 T3_Eext_LON_1030 1 12 8.04 13.9371 20.8103 22.8 29.8916 32.608 28.7265 26.3071 20.4 14.2568
10.364 6.46452
 1 593 T3_Eext_LON_1130 1 12 10.6142 16.5514 22.6026 27.396 34.3432 36.62 32.8026 29.5471 25.452
17.7948 10.22 7.95484
 1 594 T3_Eext_LON_1230 1 12 10.7226 17.3014 24.6232 30.056 36.1084 37.464 34.7884 32.5819 27.764
19.5561 10.588 8.08258
 1 595 T3_Eext_LON_1330 1 12 8.68258 15.7586 22.9006 31.588 35.7019 40.236 36.449 33.0077 28.112
18.9329 8.956 6.79742
 1 596 T3_Eext_LON_1430 1 12 6.39097 12.0986 19.8774 29.996 34.0026 37.936 34.6606 31.9239 26.644
17.5084 6.316 4.3471
 1 597 T3_Eext_LON_1530 1 12 3.47613 7.99286 15.1858 25.192 30.5497 34.388 31.9006 28.7265 21.296
13.7342 2.968 1.31613
 1 598 T3_Eext_LON_1630 1 12 0.251613 3.37714 9.01548 20.428 25.1574 30.1 27.3523 22.8774 16.064
7.92387 0.064 0
 1 599 T3_Eext_LON_1730 1 12 0 0.12 3.03484 14.34 19.8619 23.764 21.6116 17.0748 9.536 2.85677 0 0
 1 600 T3_Eext_LON_1830 1 12 0 0 0.0580645 8.152 12.9639 16.708 14.8103 10.4787 3.74 0.0851613 0 0
 1 601 T3_Eext_LON_1930 1 12 0 0 0 2.316 5.86839 9.184 7.8929 4.11484 0.244 0 0 0
 1 602 T3_Eext_MAN_630 1 12 0 0 1.19226 1.82 5.50065 8.272 6.61161 3.03097 0.168 0 0 0
 1 603 T3_Eext_MAN_730 1 12 0 0.552857 5.94968 6.612 11.2994 13.58 12.6542 8.94194 3.588 0.325161
0.384 0
 1 604 T3_Eext_MAN_830 1 12 0.948387 4.21286 11.4735 12.872 16.9394 19.444 19.0413 15.4142 9.544
3.91742 3.62 0.770323
 1 605 T3_Eext_MAN_930 1 12 4.51355 8.93571 16.8774 18.436 22.4439 24.696 24.6968 21.8361 16.172
8.93806 7.124 4.27355
```

```
1 606 T3_Eext_MAN_1030 1 12 6.94065 11.9871 21.8903 23.636 27.1665 30.6 29.4387 26.9497 20.184 13.0297 10.12 6.5071
1 607 T3_Eext_MAN_1130 1 12 8.45032 14.0314 24.6774 27.988 29.7135 34.42 33.5652 31.0568 24.284 15.3368 11.3 8.17548
1 608 T3_Eext_MAN_1230 1 12 10.0065 14.9314 24.6194 29.024 31.4748 36.952 35.4 33.5265 25.236 16.32 11.208 7.90839
1 609 T3_Eext_MAN_1330 1 12 8.45806 14.5414 23.2529 30.264 32.8103 37.512 37.1381 34.0297 25.36 17.88 9.372 6.91742
1 610 T3_Eext_MAN_1430 1 12 6.06194 11.6443 18.9135 29.264 30.1548 36.812 34.6219 32.671 24.388 15.6813 6.288 4.14968
1 611 T3_Eext_MAN_1530 1 12 3.10452 7.75286 15.5187 26.016 25.6374 34.048 30.6387 29.2219 21.384 12.2245 2.864 1.12258
1 612 T3_Eext_MAN_1630 1 12 0.274839 3.3 9.39871 21.244 23.8723 28.56 26.8529 24.5884 16.136 7.93935 0.108 0
1 613 T3_Eext_MAN_1730 1 12 0 0.201429 3.66194 14.664 18.96 23.108 22.7613 19.0181 10.284 3.17032 0 0
1 614 T3_Eext_MAN_1830 1 12 0 0 0.181935 8.656 12.4529 16.444 16.7652 11.9845 4.312 0.193548 0 0
1 615 T3_Eext_MAN_1930 1 12 0 0 0 3.228 6.74323 10.236 9.47226 5.20645 0.46 0 0 0
1 616 T3_Eext_NEW_630 1 12 0 0 1.30839 1.972 6.24387 8.512 7.08774 3.45677 0.248 0 0 0
1 617 T3_Eext_NEW_730 1 12 0 0.54 5.59355 6.78 11.8374 14.964 13.049 9.20129 3.496 0.344516 0.288 0
1 618 T3_Eext_NEW_830 1 12 0.700645 4.42714 11.5239 12.184 17.3768 20.408 20.0013 15.1858 9.328 4.06839 3.14 0.499355
1 619 T3_Eext_NEW_930 1 12 4.01032 8.79429 15.72 18.524 22.6026 26.72 25.4284 20.6477 14.556 8.58581 6.52 3.62323
1 620 T3_Eext_NEW_1030 1 12 6.49935 12.5443 19.8 24.1 26.1213 31.42 29.3265 25.4632 19.22 12.9058 9.392 5.26839
1 621 T3_Eext_NEW_1130 1 12 9.0271 14.5543 22.2271 27.696 29.2258 33.512 34.111 29.969 22 16.3316 10.624 6.67742
1 622 T3_Eext_NEW_1230 1 12 9.15871 15.5271 22.6142 28.868 32.9729 35.648 36.7974 32.6439 24.48 18.7626 10.276 6.43742
1 623 T3_Eext_NEW_1330 1 12 7.50968 14.1729 22.4903 30.568 33.2052 37.556 36.4645 32.8723 23.948 18.9948 8.724 5.59355
1 624 T3_Eext_NEW_1430 1 12 5.38065 11.6143 19.3355 29.096 31.8619 35.572 35.2568 30.8361 23.456 16.9432 5.392 3.99484
1 625 T3_Eext_NEW_1530 1 12 2.41161 7.42286 14.5858 24.952 27.96 31.376 33.9561 27.4374 20.18 13.1342 2.12 0.619355
1 626 T3_Eext_NEW_1630 1 12 0.12 2.97857 8.49677 20.456 23.5006 26.94 29.3342 23.0632 15.608 7.78452 0.036 0
1 627 T3_Eext_NEW_1730 1 12 0 0.115714 3.1471 14.332 18.5574 22.016 22.6839 17.88 10.08 2.64774 0 0
1 628 T3_Eext_NEW_1830 1 12 0 0 0.150968 8.352 12.7742 15.984 16.4594 11.1948 4.02 0.127742 0 0
1 629 T3_Eext_NEW_1930 1 12 0 0 0 3.16 6.83613 10.136 9.99097 5.11742 0.42 0 0 0
1 630 T3_Eext_NOR_630 1 12 0 0 2.03613 2.708 6.89419 8.608 7.39355 4.1071 0.548 0 0 0
1 631 T3_Eext_NOR_730 1 12 0.00387097 1.39714 6.52258 7.472 12.7084 15.232 13.92 10.0103 5.124 0.932903 0.98 0
1 632 T3_Eext_NOR_830 1 12 2.00516 5.43 12.3794 13.216 18.4994 21.644 19.4206 16.3781 10.696 5.44258 4.804 1.69161
1 633 T3_Eext_NOR_930 1 12 5.09419 9.79714 17.2955 18.212 23.2181 26.164 24.991 21.7239 16.308 10.5561 8.368 4.25806
1 634 T3_Eext_NOR_1030 1 12 7.94323 13.3071 21.2516 23.788 28.3394 31.948 29.3652 26.6671 21.016 15.1781 10.808 6.6929
1 635 T3_Eext_NOR_1130 1 12 9.43355 15.2957 23.7058 27.572 32.3419 34.688 33.329 29.8335 23.644 19.169 11.68 8.05935
1 636 T3_Eext_NOR_1230 1 12 9.47613 14.94 23.6245 30.588 34.5523 35.9 35.3652 32.3535 25.968 19.44 11.56 7.93548
1 637 T3_Eext_NOR_1330 1 12 7.93548 14.7729 21.5574 31.14 33.5071 35.66 35.4348 32.1484 24.544 18.4723 9.116 5.91871
1 638 T3_Eext_NOR_1430 1 12 5.69419 11.9657 19.2813 28.812 32.5742 35.132 33.9058 29.8839 23.296 16.2232 5.712 4.57548
1 639 T3_Eext_NOR_1530 1 12 2.53548 6.96 13.8735 25.14 27.6387 30.408 30.7277 26.2413 19.536 11.6516 2.1 0.685161
```

```
1 640 T3_Eext_NOR_1630 1 12 0.0967742 2.60143 8.05161 20.128 23.3845 25.68 25.2929 21.6116 14.124 6.36774 0.012 0
1 641 T3_Eext_NOR_1730 1 12 0 0.0428571 2.51226 13.684 17.6052 20.04 20.7329 15.9561 8.684 2.29161 0 0
1 642 T3_Eext_NOR_1830 1 12 0 0 0.0270968 7.276 11.4813 14.356 14.5703 9.72387 3.128 0.0503226 0 0
1 643 T3_Eext_NOR_1930 1 12 0 0 0 1.856 5.58581 7.944 8.12516 3.75097 0.156 0 0 0
1 644 T3_Eext_NOT_630 1 12 0 0 1.39742 2 5.51613 7.596 5.6129 3.31742 0.26 0 0 0
1 645 T3_Eext_NOT_730 1 12 0 0.788571 6.08129 6.436 11.2723 13.688 10.9432 9.48774 4.38 0.476129 0.56 0
1 646 T3_Eext_NOT_830 1 12 1.28516 4.44429 11.7174 11.328 15.8594 18.372 15.9174 15.6735 9.972 3.92903 3.88 1.06452
1 647 T3_Eext_NOT_930 1 12 4.62968 8.11286 16.0839 16.76 20.3071 24.472 21.8981 22.3587 15.252 7.64516 6.948 4.43226
1 648 T3_Eext_NOT_1030 1 12 7.09548 11.6014 20.2877 20.116 25.6219 28.8 27.2013 27.2942 20.552 10.9161 9.404 6.6929
1 649 T3_Eext_NOT_1130 1 12 9.11226 14.4943 22.9897 23.416 30.449 33.36 29.3226 30.5032 25.416 14.2065 10.668 8.19484
1 650 T3_Eext_NOT_1230 1 12 8.92645 14.1771 23.5587 27.892 33.6658 38.284 33.6232 34.5561 27.08 15.6852 10.664 8.24516
1 651 T3_Eext_NOT_1330 1 12 8.83742 14.6014 21.9871 29.692 31.2968 37.38 33.9329 34.4942 27.536 17.2065 8.732 7.29677
1 652 T3_Eext_NOT_1430 1 12 6.14323 11.1557 18.8361 26.84 32.4697 36.204 33.2245 31.4284 26.436 14.9381 5.808 3.87484
1 653 T3_Eext_NOT_1530 1 12 3.12774 7.26857 14.2258 25.044 29.0826 31.648 32.0013 28.3355 21.596 12.0968 2.672 1.07613
1 654 T3_Eext_NOT_1630 1 12 0.232258 3.20143 8.60516 20.144 23.9806 27.416 28.6065 23.8916 15.952 7.32 0.08 0
1 655 T3_Eext_NOT_1730 1 12 0 0.154286 3.20516 14.324 18.9794 21.416 21.2594 17.6323 10.388 3.0929 0 0
1 656 T3_Eext_NOT_1830 1 12 0 0 0.108387 7.952 11.6013 15.296 14.5548 10.8387 4.292 0.150968 0 0
1 657 T3_Eext_NOT_1930 1 12 0 0 0 2.72 6.10452 9.412 8.19871 4.66839 0.392 0 0 0
1 658 T3_Eext_PLY_630 1 12 0 0 0.80129 0.976 4.2929 5.552 4.8929 1.80774 0.036 0 0 0
1 659 T3_Eext_PLY_730 1 12 0 0.57 4.86968 4.956 9.66194 10.244 11.2839 7.11484 3.128 0.243871 0.444 0
1 660 T3_Eext_PLY_830 1 12 1.35484 4.37571 9.70839 10.548 15.5845 14.552 18.2787 14.191 9.084 3.67355 4.208 1.25032
1 661 T3_Eext_PLY_930 1 12 4.43226 8.58857 15.9058 15.912 21.2826 19.584 24.6194 19.9548 14.86 8.79097 8.136 4.58323
1 662 T3_Eext_PLY_1030 1 12 7.60645 12.1843 19.4439 21.096 25.2116 23.828 28.7071 24.9987 19.24 13.2116 10.984 7.7071
1 663 T3_Eext_PLY_1130 1 12 10.6219 14.85 21.6465 24.676 27.4761 26.392 32.2065 30.0194 23.82 16.2697 12.28 9.70452
1 664 T3_Eext_PLY_1230 1 12 10.7923 16.0543 24.5884 26.812 29.7677 29.128 35.8065 32.4852 25.76 17.8839 12.124 9.6
1 665 T3_Eext_PLY_1330 1 12 9.66581 14.61 22.5755 26.832 30.6116 30.588 35.9071 32.1406 27.024 19.6452 11.28 8.43484
1 666 T3_Eext_PLY_1430 1 12 7.40516 12.5614 20.0632 24.64 28.7729 31.368 34.44 31.3781 26.884 17.4542 8.22 5.92258
1 667 T3_Eext_PLY_1530 1 12 4.68387 8.07429 15.3987 24.028 27.2594 28.276 30.4606 28.0297 23.684 13.7535 4.344 2.92645
1 668 T3_Eext_PLY_1630 1 12 0.975484 4.56857 8.95355 19.016 20.6787 24.036 27.2826 23.4852 18.444 9.17419 0.46 0.0154839
1 669 T3_Eext_PLY_1730 1 12 0 0.591429 3.9329 13.752 18.0194 19.812 22.5948 17.9497 11.804 4.37419 0 0
1 670 T3_Eext_PLY_1830 1 12 0 0 0.298065 9.028 12.3987 14.532 15.6581 11.969 5.556 0.456774 0 0
1 671 T3_Eext_PLY_1930 1 12 0 0 0 3.368 6.72 8.896 9.47226 4.98194 0.636 0 0 0
1 672 T3_Eext_SOU_630 1 12 0 0 1.31613 1.756 5.01677 7.708 5.86452 2.81032 0.18 0 0 0
1 673 T3_Eext_SOU_730 1 12 0 1.01571 5.79484 6.504 10.6877 14.528 11.5548 8.07871 4.06 0.549677 0.84 0
1 674 T3_Eext_SOU_830 1 12 1.9471 5.46429 10.7613 13.332 16.3084 19.248 18.4297 14.8645 9.988 4.85032 4.452 1.71484
```

```
 1 675 T3_Eext_SOU_930 1 12 5.23742 9.9 15.36 18.096 21.0929 24.64 22.8735 20.9187 16.692 10.6723
8.148 4.63355
 1 676 T3_Eext_SOU_1030 1 12 8.65161 12.84 18.5613 23.76 26.64 30.44 27.2981 26.6206 21.696 14.8142
10.528 7.84645
 1 677 T3_Eext_SOU_1130 1 12 10.7226 16.4486 22.3703 27.096 29.4503 34.56 32.0361 31.0916 25.004
18.3445 12.348 9.31742
 1 678 T3_Eext_SOU_1230 1 12 11.1213 17.3143 22.5213 27.532 32.489 37.056 35.5006 33.2206 26.552
19.2232 12.004 9.66194
 1 679 T3_Eext_SOU_1330 1 12 10.4206 15.3343 20.8645 29.86 31.0335 35.096 34.6684 34.2232 27.972
19.2155 11.132 8.02452
 1 680 T3_Eext_SOU_1430 1 12 7.43613 12.3986 17.8103 27.828 29.4774 35.436 36.3484 32.3265 24.84
17.0594 7.416 4.60645
 1 681 T3_Eext_SOU_1530 1 12 4.09548 8.17714 14.2606 23.524 27.24 31.836 35.0361 28.1845 20.672
13.5174 3.644 1.84645
 1 682 T3_Eext_SOU_1630 1 12 0.464516 3.63857 8.89548 19.544 23.7948 29.284 28.0877 24.569 16.46
8.33032 0.172 0
 1 683 T3_Eext_SOU_1730 1 12 0 0.231429 3.48 14.616 18.8594 22.516 23.0671 17.6555 10.592 3.5071 0 0
 1 684 T3_Eext_SOU_1830 1 12 0 0 0.0929032 8.52 12.84 15.276 16.0955 11.3226 4.376 0.189677 0 0
 1 685 T3_Eext_SOU_1930 1 12 0 0 0 2.704 6.5071 8.856 8.71742 4.67613 0.304 0 0 0
 1 686 T3_Eext_SWI_630 1 12 0 0 2.24903 2.664 6.61548 8.732 7.8271 4.04903 0.52 0 0 0
 1 687 T3_Eext_SWI_730 1 12 0.0348387 1.85143 6.89032 7.128 12.36 14.836 14.3071 10.5135 5.24
1.13032 1.508 0.0309677
 1 688 T3_Eext_SWI_830 1 12 2.83742 5.71286 12.4723 11.808 19.7652 21.388 21.2323 16.8619 12.28
6.12387 5.02 2.53548
 1 689 T3_Eext_SWI_930 1 12 5.97677 9.89143 16.9316 17.22 23.9961 25.3 26.8955 22.6529 18.312
10.7458 8.624 5.5471
 1 690 T3_Eext_SWI_1030 1 12 8.86452 13.4871 21.6929 24.252 28.6723 30.228 31.6684 27.6774 22.888
15.0619 11.048 8.26452
 1 691 T3_Eext_SWI_1130 1 12 9.65032 16.2257 22.5755 25.32 32.9845 35.108 35.3342 31.1419 25.956
17.9458 11.548 10.0994
 1 692 T3_Eext_SWI_1230 1 12 10.6645 16.41 23.0245 28.064 34.1381 35.096 37.0606 33.6194 26.832
19.649 11.732 9.65419
 1 693 T3_Eext_SWI_1330 1 12 9.44903 15.1929 21.5148 27.336 33.9523 33.036 36.2594 33.3445 26.74
19.6026 9.548 7.68387
 1 694 T3_Eext_SWI_1430 1 12 6.36387 12.1329 19.2619 25.216 29.9961 33.236 35.0245 31.56 24.484
16.9355 6.296 4.08
 1 695 T3_Eext_SWI_1530 1 12 3.10452 7.07143 13.7884 22.06 28.9123 29.856 32.1329 28.32 20.024
12.4181 2.632 1.01419
 1 696 T3_Eext_SWI_1630 1 12 0.147097 2.72143 8.00516 18.308 23.0439 25.248 26.9729 22.9161 14.276
7.40516 0.028 0
 1 697 T3_Eext_SWI_1730 1 12 0 0.0514286 2.44645 11.928 17.3342 19.404 20.8142 16.5987 8.692
2.44645 0 0
 1 698 T3_Eext_SWI_1830 1 12 0 0 0.0116129 6.26 11.4271 13.376 14.3226 9.5071 3.024 0.0387097 0 0
 1 699 T3_Eext_SWI_1930 1 12 0 0 0 1.616 5.09032 7.604 7.2929 3.36 0.1 0 0 0
 1 700 tVinf_BEL 12 9 3.23 1.98 1.46 0.99 0.74 1.8 2.83 3.58 5.23 2.84 5.39 4.56 2.77 0.1 0.25 0.92 2.42 6.12
4.53 3.26 1.57 0.85 0.48 1.19 2.46 3.94 5.67 1.9 2.44 2.43 2.04 1.4 1.8 0.98 1.1 4.63 1.76 1.83 2.22 1.92 1.16
1.56 1.54 1.68 4.17 1.76 2 2.08 1.83 1.32 2.36 1.66 0.84 4.65 1.87 1.24 1.1 1.02 0.68 2.56 2.35 2.17 4.36 1.4 2
2.57 1.94 0.99 1.06 1.64 1.62 3.91 1.62 1.54 1.22 0.83 0.44 3.21 3.19 2.23 4.77 2.93 3.44 2.65 1.11 0.33 1.58
2.12 2.87 5.12 2.59 2.73 2.42 1.49 0.68 0.85 1.29 2.18 4.22 4.68 3.15 1.2 0.85 0.6 0.3 1.37 4.3 5.23
 1 701 tVinf_BIR 12 9 3.08 1.36 0.1 0.36 0.36 1.93 2.29 3.62 4.59 2.23 0.75 0.55 1.46 1.43 2.19 2.29 2.71
4.56 2.53 1.65 1.11 1.1 0.89 1.5 1.97 2.9 4.29 2.49 2.44 1.09 1.95 1.85 0.87 0.54 2.05 4.23 2.1 1.79 1.23 1.13
0.97 1.59 1.71 1.93 3.81 1.4 0.68 1.64 1.81 1.63 1.63 2.13 1.89 3.81 1.54 1 0.96 1.12 0.89 1.23 1.63 1.77 3.19
2.14 1.37 0.73 0.47 0.45 1.56 1.87 2.04 3.42 1.26 0.73 0.87 1.39 1.31 1.68 2.18 1.66 3.53 2.26 2.13 1.93 1.42
0.64 1.01 1.68 2.27 3.85 3.2 1.53 0.43 0.78 0.78 1.08 2.16 3.39 4.26 4.15 3.28 1.66 1.12 0.75 0.52 1.05 3.45
4.98
 1 702 tVinf_CAR 12 9 3.34 2.56 1.86 1.13 0.14 3.08 4.78 5.35 6.74 0.98 2.88 3.54 2.9 0.51 2.26 2.49 1.67
5.27 1.68 2.41 3.1 2.13 1.51 2.17 2.62 2.48 5.18 1.09 2.17 4.23 3.68 2.49 0.8 1.1 1.15 4.97 0.97 1.18 3 2.85
1.95 2.53 3.01 2.6 5.24 0.97 0.77 2.3 2.25 1.92 2.42 3.02 2.83 4.7 0.75 0.8 1.38 1.25 0.67 3.73 4.07 3.71 4.96
1.75 1.61 1.31 0.8 0.37 2.62 3.06 2.78 4.38 1.33 1.52 3.47 3.46 2.53 1.45 2.29 2.28 5.32 1.73 2.37 3.66 2.99
1.72 2.34 2.98 2.92 5.94 2.42 3.14 3.22 2.19 0.93 0.97 1.71 2.11 4.91 2.37 2.67 2.36 1.62 0.62 2.68 3.97 4.25
6.15
```

1 703 tVinf_EDI 12 9 3.5 1.22 1.17 1.33 0.71 0.78 2.97 3.96 4.82 4.64 2.14 1.46 1.03 0.41 0.15 2.25 4.59 5.37 2.18 1.63 2.03 1.94 0.59 2.14 3.57 3.71 5.23 2.01 0.85 1.38 1.69 1.29 2.1 2.96 3.27 4.85 1.16 1.33 3.11 3.09 1.68 1.28 2.21 2.03 4.65 2.23 0.48 1.07 1.21 1.07 1.31 3.21 3.26 4.18 1.26 0.28 0.79 0.92 0.82 2.37 3.65 3.63 4.22 1.94 0.61 1.03 1.16 1 1.13 2.83 3.08 3.91 1.66 0.25 0.34 0.42 0.34 2.66 4.18 4.04 4.49 2.11 1.66 2.22 2.06 0.89 0.61 2.43 2.7 4.3 3.04 0.81 1.15 1.21 0.92 1.05 3.1 3.65 4.53 2.9 0.73 0.78 0.61 0.2 2.22 4.25 4.48 5.13
1 704 tVinf_GLA 12 9 2.08 1.05 2.05 2.04 0.89 2.96 4.2 3.96 5.77 2.14 2.83 3.34 3.17 0.59 1.07 2.23 2.75 5.23 3.7 1.46 1.05 0.76 0.33 1.9 3.72 4.56 5.46 1.3 1.82 3.13 3.29 1.84 1.24 1.14 1.12 4.67 1.63 2.15 2.87 2.89 0.65 1.08 2.02 1.99 4.49 1.68 0.83 1.2 1.19 0.38 2.47 3.74 3.38 4.53 1.65 0.45 0.63 0.57 0.31 2.46 3.5 3 3.94 2.5 0.85 0.95 0.76 0.4 1.87 3.73 3.93 4.7 2.72 1.01 1.06 0.94 0.45 1.82 3.43 4.04 4.74 1.96 1.97 2.77 2.66 0.75 1.49 2.31 2.68 4.94 3.32 1.73 1.54 1.43 1.04 1.28 2.23 3.71 4.94 3.37 1.42 1.37 1.1 0.28 1.43 3 4.06 5.06
1 705 tVinf_LEE 12 9 1.66 1.35 1.11 0.38 0.12 3.07 3.92 3.24 4.74 0.69 0.9 0.89 0.59 0.06 4.01 4.12 2.67 4.69 1.42 1.57 1.62 0.9 0.62 3.02 3.56 3.21 4.73 1.25 1.77 1.64 1.17 0.42 1.96 2.35 2.33 3.77 0.92 1 1.18 1.21 0.85 1.51 2.03 1.58 3.19 0.61 0.78 0.83 0.75 0.31 3.24 3.52 2.92 4.14 0.64 0.89 0.99 0.97 0.58 2.95 3.1 2.64 3.99 0.68 0.79 1.06 1.09 0.73 2.62 3.01 2.57 3.87 0.52 0.46 0.62 0.81 0.64 2.84 3.02 2.06 3.64 1.15 1.24 1.3 0.94 0.67 3.24 3.46 2.85 4.54 0.59 0.93 1.06 0.9 0.6 3.01 2.93 1.89 3.87 1.74 0.92 0.59 0.54 0.41 3.07 3.62 3.62 4.5
1 706 tVinf_LON 12 9 3.64 1.95 0.8 0.58 0.34 0.89 2.6 3.92 4.58 2.43 0.32 0.79 1.71 1.65 1.92 3.15 3.26 4.85 2.84 1.16 1.36 1.48 1.21 1.33 2.58 3.4 4.74 2.43 1.3 0.67 0.65 0.52 1.54 2.17 2.77 3.81 1.91 0.81 0.91 1.8 1.7 0.9 1.76 2.37 3.76 1.8 1.2 1.06 0.81 0.3 1.55 2.08 2.29 3.5 2.21 1.57 1.45 1.27 0.75 0.85 1.71 2.5 3.58 2.15 1.45 0.86 0.55 0.29 1.12 1.6 2.14 3.23 2.35 1.14 0.56 0.67 0.45 0.54 1.69 2.68 3.18 2.11 1.91 1.95 1.62 0.86 0.6 1.54 2.21 3.73 2.59 1.4 0.52 0.34 0.06 0.94 2.17 3.15 3.6 1.88 1.31 1.57 1.58 1.2 1.69 2.51 2.61 4.25
1 707 tVinf_MAN 12 9 4.25 2.84 0.18 0.08 0.08 1.27 1.85 4.55 5 3.81 2.7 0.59 0.15 0.09 1.9 2.56 4.24 5.02 2.74 2.4 1.78 1.36 0.6 1.6 2.18 3.17 4.86 2.24 2.21 1.62 2.36 2.05 0.81 0.61 1.92 4.31 1.75 1.89 1.93 2.2 1.91 1.42 1.52 1.77 4.25 1.85 1.46 0.86 0.79 0.55 1.97 2.31 2.56 3.84 1.71 1.02 0.53 0.47 0.28 2.12 2.73 2.45 3.62 2.16 2.01 1 0.62 0.46 1.65 1.87 2.05 3.65 1.58 1.72 2.37 2.56 1.89 0.91 1.08 1.66 4.05 3.17 2.09 0.46 0.23 0.03 1.56 2.38 3.71 4.31 2.28 1.45 0.76 1 0.84 1.76 2.34 2.52 4.02 3.17 2.53 2.21 2.09 1.22 0.5 1.13 3.09 4.7
1 708 tVinf_NEW 12 9 2.85 2.04 0.82 0.45 0.23 1.98 2.64 3.7 4.51 1.42 0.34 0.15 0.13 0.07 2.91 3.91 3.5 4.24 1.36 0.96 0.98 1.08 0.74 2.28 2.59 2.43 3.87 0.96 1.72 1.84 2.45 1.69 1.68 1.48 0.92 4.15 1.25 1.17 1.08 1.45 1.14 1.34 1.13 1.23 3.16 1.63 1.12 1.29 1.32 1.04 1.11 1.82 2.19 3.39 1.93 0.94 0.87 0.96 0.57 0.76 1.91 2.09 3.04 1.47 1.06 0.98 0.68 0.53 1.81 2.59 2.69 3.55 1.31 1.04 1.44 1.76 1.4 1.35 1.55 1.33 3.42 2.55 1.47 0.75 0.76 0.64 1.53 2.1 3.13 3.96 1.85 1.68 1.44 0.91 0.78 1.54 1.69 1.94 3.52 3.12 1.81 0.48 0.22 0.21 2.21 2.96 3.86 4.67
1 709 tVinf_NOR 12 9 3.26 1.58 0.45 0.72 0.59 2.62 3.9 4.65 5.68 1.93 0.36 0.14 0.24 0.21 4.43 5.6 3.77 5.92 3.11 1.29 1.43 1.46 1.11 1.56 3.14 3.72 5.11 0.82 0.78 1.42 2.2 2.03 2.64 2.75 2.41 4.8 2.16 1.56 2.48 2.99 2.53 0.69 1.53 2.17 4.78 2.51 1.19 0.7 0.69 0.62 1.7 2.41 3.12 4.16 2.41 1.05 0.78 1.03 0.89 1.39 2.47 2.72 3.94 2.09 1.65 1.15 0.79 0.58 1.98 1.88 1.85 3.98 2.61 1.07 0.52 0.16 0.09 2.03 3.57 3.62 4.38 3.62 2.93 1.7 0.84 0.25 0.63 1.78 3.1 4.58 2.01 0.28 0.11 0.99 0.98 2.41 3.55 3.36 4.54 3.56 1.67 0.77 0.51 0.35 1.9 3.49 4.03 5.12
1 710 tVinf_NOT 12 9 3.75 1.82 0.64 0.42 0.35 1.64 2.97 3.99 5.11 2.37 0.21 0.33 0.31 0.2 2.56 4.02 3.46 4.51 2.43 1.35 1.69 1.53 1.43 0.91 2.22 2.65 4.12 1.98 1.25 1.48 1.91 1.65 1.25 1.39 1.91 4.07 2.72 0.8 0.6 0.33 0.15 1.27 2.88 3.5 3.89 1.15 0.57 0.45 0.94 0.82 2.44 2.86 2.16 3.78 1.39 0.79 1.25 1.28 0.86 1.43 2.24 2.19 3.45 1.43 0.62 1.36 1.5 1.23 1.18 2 2.2 3.45 2.92 0.94 0.54 0.57 0.35 0.86 2.42 3.27 3.78 2.12 1.56 2.02 1.87 0.58 0.96 2.52 2.7 4.13 1.78 0.55 0.78 1.47 1.32 1.5 2.41 2.32 3.8 4.26 1.18 0.48 0.36 0.32 0.62 2.97 4.4 4.83
1 711 tVinf_PLY 12 9 3.48 1.53 0.62 0.85 0.62 3.55 5.07 5.36 6.88 1.51 0.49 0.56 0.97 0.82 4.36 5.07 3.11 5.86 3.18 3.38 3.23 2.84 0.89 1.44 2.99 3.72 6.26 2.92 1.95 1.1 1.46 1.3 2.31 2.41 3.13 5.36 2.34 1.51 2.21 2.13 0.84 2.08 3.23 3.28 5.24 3.56 1.9 1.55 1.05 0.26 1.29 3.06 3.81 5.13 2.52 1.79 1.4 1.25 0.29 1.63 1.98 2.81 4.33 2.31 1.58 1 0.52 0.18 2.16 2.67 2.79 4.41 2.62 0.94 0.79 0.94 0.73 2.9 4.32 4.1 5.48 3.82 1.52 1.02 1.03 0.48 2.51 4.65 5.52 6.41 4.62 3.14 1.91 1.37 0.63 1.23 3.3 4.95 6.46 4.79 3.09 1.84 0.76 0.26 1.79 3.83 5.02 6.72
1 712 tVinf_SOU 12 9 0.82 1.21 1.62 1.37 0.6 1.52 1.69 1.7 3.14 0.42 0.3 0.35 0.3 0.19 2.77 2.77 1.28 3.07 1.25 0.75 0.53 0.34 0.29 1.68 2.22 2.12 2.94 1.01 1.06 1.56 1.92 1.67 0.27 0.44 0.82 2.77 1.41 1.01 1.1 1.4 1.24 1.35 1.3 1.34 3.21 0.37 0.28 0.66 1.49 1.43 1.38 1.17 0.7 2.6 1.07 0.39 0.42 0.39 0.33 1.93 2.51 2.15 2.93 0.83 0.65 0.52 0.43 0.31 1.64 1.78 1.59 2.43 1.64 0.78 0.53 0.3 0.22 1.29 2.31 2.2 2.98 1.82 1.72 1.27 0.95 0.5 0.8 1.33 1.85 3.07 0.84 0.44 0.62 1.03 0.9 1.52 1.81 1.72 2.82 1.54 0.86 0.61 0.69 0.55 1.53 2.52 2.85 3.45
1 713 tVinf_SWI 12 9 4.61 2.95 1.65 0.95 0.43 1.55 3.17 4.77 6.17 1.66 0.57 0.67 0.5 0.33 4.57 5.11 3.48 5.79 2.4 2.56 2.17 2.13 1.49 1.55 2.07 2.58 5.05 3.1 2.71 2.03 2.35 2.02 1.12 1.11 2.76 5.29 1.8 0.76 1.87 2.22 1.98 1.8 2.29 2.63 4.59 1.48 1.17 1.41 2.65 2.49 1.07 1.13 1.5 4.15 1.48 0.99 0.94 0.92 0.52 2.51 2.92 2.44 3.97 2.49 1.84 1.13 0.74 0.44 2.01 2.36 2.64 4.33 1.63 1.34 2.48 3.02 2.33 1.05 1.45 1.77 4.57 1.91 2.34 2.31 2.02 0.63 2.28 2.89 2.76 5.05 1.98 1.17 1.51 1.95 1.41 2.2 2.73 2.75 4.85 4.08 3.15 2.03 0.86 0.45 1.63 2.84

```
3.87 5.85
 1 714 tVinf_DUB 12 9 2.21 1.22 0.85 0.71 0.34 0.93 1.9 2.41 3.23 3.1 4.71 4.62 1.26 0.07 1.05 2.1 2.36 6.07
1.98 0.78 0.7 0.18 0.16 3.71 4.84 4.33 5.41 2.62 1.44 1.98 1.73 1.24 3.73 4.65 3.64 6.58 2.67 1.69 1.76 0.88
0.29 1.9 3.28 2.93 4.6 1.41 1.45 2.28 2.06 1.44 1.72 2.24 1.84 4.2 1.29 0.85 0.95 0.72 0.68 2.86 3.42 2.63 4.16
1.59 2.62 2.91 2.17 1.08 1.48 2.27 2.11 4.76 3.29 2.02 1.7 0.6 0.2 2.81 4.51 4.18 5.89 4.87 1.1 0.5 0.2 0.11
2.27 5.67 6.51 6.92 2.18 0.9 0.8 0.69 0.46 2.46 3.73 3.59 4.75 3.49 1.53 0.88 0.1 0.05 2.01 4.15 4.26 5.31
 1 715 tVinf_JER 12 9 3.82 4.72 5.77 3.59 2.35 0.03 0.15 1.53 6.79 3.71 3.72 3.1 1.87 1.07 0.87 1.29 2.42
5.37 2.93 1.65 1.38 1.04 0.97 5.19 5.95 4.95 7.58 2.29 1.58 2.53 2.68 2.06 3.7 4.3 3.63 6.92 1.63 1.54 2.05
1.97 1.15 2.29 3.01 2.71 4.93 1.01 0.88 2.5 2.8 2.32 2.11 2.28 1.74 4.72 0.5 1.11 3.63 4.22 3.58 1.65 1.62 1.18
5.45 0.81 1.39 3.61 3.81 3.03 1.92 1.89 1.38 5.36 2.57 1.68 1.03 1.39 1.36 3.91 4.26 3.4 6.32 3.45 1.28 0.77
1.6 1.49 4.65 5.91 5.6 7.88 1.78 2.77 3.78 2.88 1.38 2.18 2.65 2.31 5.83 3.58 2.36 1.03 0.21 0.1 2.28 3.18 3.86
5.19
 1 716 tWind_Vgen_BEL 1 12 4.89 5.93 5.42 4.34 3.98 4.45 4.15 3.75 4.5 4.76 3.93 4.96
 1 717 tWind_Vgen_BIR 1 12 4.33 4.33 4.04 4 3.64 3.61 3.05 3.26 3.33 3.61 4.03 4.73
 1 718 tWind_Vgen_CAR 1 12 6.38 5.1 4.98 4.77 4.95 4.5 4.78 4.18 5.07 5.64 4.68 5.83
 1 719 tWind_Vgen_EDI 1 12 4.5 5.02 4.89 4.57 4.45 3.97 3.95 3.63 4.18 3.94 4.2 4.7
 1 720 tWind_Vgen_GLA 1 12 5.3 4.78 5.16 4.37 4.26 4.21 3.72 4.41 4.33 4.55 4.51 4.67
 1 721 tWind_Vgen_LEE 1 12 4.47 4.43 4.45 3.57 3.03 3.93 3.77 3.67 3.42 4.14 3.6 4.18
 1 722 tWind_Vgen_LON 1 12 4.33 4.59 4.47 3.63 3.58 3.33 3.39 3.05 2.98 3.52 3.37 4.02
 1 723 tWind_Vgen_MAN 1 12 4.66 4.76 4.59 4.04 4.03 3.7 3.45 3.49 3.81 4.02 3.76 4.46
 1 724 tWind_Vgen_NEW 1 12 4.28 4.05 3.66 3.89 3.01 3.24 2.91 3.4 3.24 3.73 3.34 4.46
 1 725 tWind_Vgen_NOR 1 12 5.44 5.57 4.8 4.55 4.55 3.94 3.76 3.66 4.04 4.24 4.32 4.86
 1 726 tWind_Vgen_NOT 1 12 4.86 4.32 3.84 3.9 3.77 3.58 3.29 3.25 3.61 3.97 3.6 4.51
 1 727 tWind_Vgen_PLY 1 12 6.51 5.49 6.07 5.07 4.99 4.93 4.07 4.09 5.1 6 5.97 6.28
 1 728 tWind_Vgen_SOU 1 12 2.97 2.83 2.8 2.65 3.08 2.47 2.84 2.31 2.81 2.85 2.65 3.26
 1 729 tWind_Vgen_SWI 1 12 5.84 5.45 4.9 5.01 4.25 4.01 3.77 4.06 4.23 4.68 4.6 5.55
 1 730 tWind_Vgen_DUB 1 12 3.06 5.73 5.17 6.36 4.32 3.98 3.94 4.64 5.56 6.63 4.37 5.04
 1 731 tWind_Vgen_JER 1 12 6.64 5.19 7.31 6.67 4.72 4.48 5.23 5.19 6.09 7.52 5.47 4.9
 1 732 tWind_EPF_BEL 1 12 2.16 2.16 2.16 2.16 2.16 2.16 2.16 2.16 2.16 2.16 2.16 2.16
 1 733 tWind_EPF_BIR 1 12 2.26 2.26 2.26 2.26 2.26 2.26 2.26 2.26 2.26 2.26 2.26 2.26
 1 734 tWind_EPF_CAR 1 12 2.05 2.05 2.05 2.05 2.05 2.05 2.05 2.05 2.05 2.05 2.05 2.05
 1 735 tWind_EPF_EDI 1 12 2.51 2.51 2.51 2.51 2.51 2.51 2.51 2.51 2.51 2.51 2.51 2.51
 1 736 tWind_EPF_GLA 1 12 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66 2.66
 1 737 tWind_EPF_LEE 1 12 2.8 2.8 2.8 2.8 2.8 2.8 2.8 2.8 2.8 2.8 2.8 2.8
 1 738 tWind_EPF_LON 1 12 2.33 2.33 2.33 2.33 2.33 2.33 2.33 2.33 2.33 2.33 2.33 2.33
 1 739 tWind_EPF_MAN 1 12 2.27 2.27 2.27 2.27 2.27 2.27 2.27 2.27 2.27 2.27 2.27 2.27
 1 740 tWind_EPF_NEW 1 12 2.21 2.21 2.21 2.21 2.21 2.21 2.21 2.21 2.21 2.21 2.21 2.21
 1 741 tWind_EPF_NOR 1 12 2.37 2.37 2.37 2.37 2.37 2.37 2.37 2.37 2.37 2.37 2.37 2.37
 1 742 tWind_EPF_NOT 1 12 2.1 2.1 2.1 2.1 2.1 2.1 2.1 2.1 2.1 2.1 2.1 2.1
 1 743 tWind_EPF_PLY 1 12 2.35 2.35 2.35 2.35 2.35 2.35 2.35 2.35 2.35 2.35 2.35 2.35
 1 744 tWind_EPF_SOU 1 12 2.24 2.24 2.24 2.24 2.24 2.24 2.24 2.24 2.24 2.24 2.24 2.24
 1 745 tWind_EPF_SWI 1 12 2.24 2.24 2.24 2.24 2.24 2.24 2.24 2.24 2.24 2.24 2.24 2.24
 1 746 tWind_EPF_DUB 1 12 2.26 2.26 2.26 2.26 2.26 2.26 2.26 2.26 2.26 2.26 2.26 2.26
 1 747 tWind_EPF_JER 1 12 1.95 1.95 1.95 1.95 1.95 1.95 1.95 1.95 1.95 1.95 1.95 1.95
 2 9 tREC_Key_PROJECT 42 2 "EPC-C1" "COOLING" "EPC-C2" "COOLING" "EPC-C3"
"COOLING" "EPC-W1" "HOT-WATER" "EPC-W3" "HOT-WATER" "EPC-W4" "HOT-WATER"
"EPC-W2" "HOT-WATER" "EPC-E1" "ENVELOPE" "EPC-E2" "ENVELOPE" "EPC-E3"
"ENVELOPE" "EPC-E4" "ENVELOPE" "EPC-E5" "ENVELOPE" "EPC-E6" "ENVELOPE" "EPC-
E7" "ENVELOPE" "EPC-E8" "ENVELOPE" "EPC-F1" "FUEL-SWITCHING" "EPC-F2" "FUEL-
SWITCHING" "EPC-F3" "FUEL-SWITCHING" "EPC-F4" "FUEL-SWITCHING" "EPC-F5" "FUEL-
SWITCHING" "EPC-F6" "FUEL-SWITCHING" "EPC-H2" "HEATING" "EPC-H5" "HEATING"
"EPC-H6" "HEATING" "EPC-H7" "HEATING" "EPC-H8" "HEATING" "EPC-H1" "HEATING"
"EPC-H3" "HEATING" "EPC-H4" "HEATING" "EPC-L1" "LIGHTING" "EPC-L2" "LIGHTING"
"EPC-L3" "LIGHTING" "EPC-L4" "LIGHTING" "EPC-L5" "LIGHTING" "EPC-L6" "LIGHTING"
"EPC-L7" "LIGHTING" "EPC-V1" "OVERHEATING" "EPC-R1" "RENEWABLES" "EPC-R2"
"RENEWABLES" "EPC-R3" "RENEWABLES" "EPC-R4" "RENEWABLES" "EPC-R5"
"RENEWABLES"
 2 10 tREC_Key_HVAC 19 2 "EPC-F1" "FUEL-SWITCHING" "EPC-F2" "FUEL-SWITCHING"
"EPC-F3" "FUEL-SWITCHING" "EPC-F4" "FUEL-SWITCHING" "EPC-F5" "FUEL-SWITCHING"
"EPC-F6" "FUEL-SWITCHING" "EPC-H1" "HEATING" "EPC-H3" "HEATING" "EPC-H4"
"HEATING" "EPC-R1" "RENEWABLES" "EPC-R5" "RENEWABLES" "EPC-H2" "HEATING"
```

```
"EPC-H5" "HEATING" "EPC-H6" "HEATING" "EPC-H7" "HEATING" "EPC-H8" "HEATING" "EPC-C1" "COOLING" "EPC-C2" "COOLING" "EPC-C3" "COOLING"
 2 11 tREC_Key_DHW 4 2 "EPC-W1" "HOT-WATER" "EPC-W3" "HOT-WATER" "EPC-W4" "HOT-WATER" "EPC-W2" "HOT-WATER"
 1 748 tPayBacks 1 3 10 5 3
 1 749 tFuel_Payback 1 13 1 0.36 0.68 0.58 1.64 1.64 1.64 0.68 0.68 1.22 0 0.1 1.1
 1 750 tRec_Pot_Impact 3 3 2 2 1 3 2 1 3 3 2
 1 751 tRel_Carbon 13 13 1 0.829059829059829 7.76 0.732075471698113 0.666666666666667 0.611987381703470 0.494897959183673 1.03743315508021 7.76 0.459715639810427 1 10.7777777777778 1 1.20618556701031 1 9.36 0.883018867924528 0.804123711340206 0.738170347003155 0.596938775510204 1.25133689839572 9.36 0.554502369668247 1 13 1 0.128865979381443 0.106837606837607 1 9.43396226415094E-02 8.59106529209622E-02 7.88643533123028E-02 6.37755102040816E-02 0.133689839572193 1 5.92417061611374E-02 1 1.38888888888889 1 1.3659793814433 1.13247863247863 10.6 1 0.910652920962199 0.83596214511041 0.676020408163265 1.41711229946524 10.6 0.627962085308057 1 14.7222222222222 1 1.5 1.24358974358974 11.64 1.09811320754717 1 0.917981072555205 0.74234693877551 1.55614973262032 11.64 0.68957345971564 1 16.1666666666667 1 1.6340206185567 1.35470085470085 12.68 1.19622641509434 1.0893470790378 1 0.808673469387755 1.6951871657754 12.68 0.751184834123223 1 17.6111111111111 1 2.02061855670103 1.67521367521368 15.68 1.47924528301887 1.34707903780069 1.23659305993691 1 2.09625668449198 15.68 0.928909952606635 1 21.7777777777778 1 0.963917525773196 0.799145299145299 7.48 0.705660377358491 0.642611683848797 0.589905362776025 0.477040816326531 1 7.48 0.443127962085308 1 10.3888888888889 1 0.128865979381443 0.106837606837607 1 9.43396226415094E-02 8.59106529209622E-02 7.88643533123028E-02 6.37755102040816E-02 0.133689839572193 1 5.92417061611374E-02 1 1.38888888888889 1 2.17525773195876 1.80341880341880 16.88 1.59245283018868 1.45017182130584 1.33123028391167 1.07653061224490 2.25668449197861 16.88 1 1 23.4444444444444 1 1 1 1 1 1 1 1 1 1 1 1 1 9.27835051546392E-02 7.69230769230769E-02 0.726 7.92452830188680E-02 6.18556701030928E-02 0.056782334384858 4.59183673469388E-02 9.62566844919786E-02 0.72 0.042654028436019 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 752 tFuel_Price_Weighting 1 13 1 2.74 1.48 1.72 0.61 1.07 0.61 1.48 1.48 3.43 1 0.2 0.875
 1 753 TdlH_BEL 1 12 9 10 11 12 12 12 12 12 12 11 9 8
 1 754 TdlH_BIR 1 12 9 10 11 12 12 12 12 12 12 11 9 8
 1 755 TdlH_CAR 1 12 9 10 11 12 12 12 12 12 12 11 9 8
 1 756 TdlH_EDI 1 12 9 10 11 12 12 12 12 12 12 11 9 8
 1 757 TdlH_GLA 1 12 9 10 11 12 12 12 12 12 12 11 9 8
 1 758 TdlH_LEE 1 12 9 10 11 12 12 12 12 12 12 11 9 8
 1 759 TdlH_LON 1 12 9 10 11 12 12 12 12 12 12 11 9 8
 1 760 TdlH_MAN 1 12 9 10 11 12 12 12 12 12 12 11 9 8
 1 761 TdlH_NEW 1 12 9 10 11 12 12 12 12 12 12 11 9 8
 1 762 TdlH_NOR 1 12 9 10 11 12 12 12 12 12 12 11 9 8
 1 763 TdlH_NOT 1 12 9 10 11 12 12 12 12 12 12 11 9 8
 1 764 TdlH_PLY 1 12 9 10 11 12 12 12 12 12 12 11 9 8
 1 765 TdlH_SOU 1 12 9 10 11 12 12 12 12 12 12 11 9 8
 1 766 TdlH_SWI 1 12 9 10 11 12 12 12 12 12 12 11 9 8
 1 767 TdlH_DUB 1 12 9 10 11 12 12 12 12 12 12 11 9 8
 1 768 TdlH_JER 1 12 9 10 11 12 12 12 12 12 12 11 9 8
```

ActlibraryTab

Coded file

```
$ File with default Activity Tables (monthly values) in NCM
$ 10-11-10
$ CONF order
$ int cLIGHT_ILLUM=1;
$ int cLIGHT_DIS=2;
$ int cBR_LIGHT_CHECK=3;
$ int cPERS_HG_SEN=4;
$ int cPERS_HG_LAT=5;
$ int cEQUP_HG_SEN=6;
$ int cEQUP_HG_LAT=7;
$ int cCOOL_SET_POINT=8;
$ int cHEAT_SET_POINT=9;
```

```
$ int cHEAT_SET_AVE=10;
$ int cSB_SET_POINT=11;
$ int cOA_FLOW_PERSON=12;
$ int cOA_FLOW_PERSON_DES=13;
$ int cHWS=14;
$ int cMAX_TIME_REDUCED=15;
$ int cMIN_TIME_REDUCED=16;
$ int cOCC_T_START=17;
$ int cOCC_T_END=18;
$ int cLIGHT_T_START=19;
$ int cLIGHT_T_END=20;
$ int cSYS_T_START=21;
$ int cSYS_T_END=22;
$ int cSYS_T_HOURS=23;
$ int cSYS_HEAT_T_START=24;
$ int cSYS_HEAT_T_END=25;
$ int cSYS_HEAT_T_HOURS=26;
$ int cHEAT_MODE=27;
$ int cCOOL_MODE=28;
$ int cOCC_T_HOURS=29;
$ int cSERVICE=30;
$ int cDRIVER2A=31;
$ int cDRIVER2B=32;
$ int cSOLAR_GAIN_CHECK=33;
$ int cPEAK_INTERNAL_GAINS=34;
$ int cDIMMING=35;
$ int cSYS_COOL_DAYS=36;
 1 0 CONF 36 2 1 1 2 1 3 12 15 12 27 12 39 12 51 12 63 12 75 12 87 12 99 12 111 12 123 12 135 12 147 12
159 12 171 12 183 12 195 12 207 12 219 12 231 12 243 12 255 12 267 12 279 12 291 12 303 12 315 12 327
1 328 1 329 1 330 1 331 1 332 1 333 12
 1 1000 A1000 1 344 50.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 1694.00 1540.00 1694.00 1463.00 1617.00 1617.00
1694.00 1694.00 1540.00 1771.00 1694.00 1463.00 1694.00 1540.00 1694.00 1463.00 1617.00 1617.00
1694.00 1694.00 1540.00 1771.00 1694.00 1463.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00
16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 242.00
220.00 242.00 209.00 231.00 231.00 242.00 242.00 220.00 253.00 242.00 209.00 242.00 220.00 242.00
209.00 231.00 231.00 242.00 242.00 220.00 253.00 242.00 209.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0
6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0
6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0
264.0 264.0 240.0 276.0 264.0 228.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 220.0 200.0 220.0 190.0
210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 3 0 2 0 7.7 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0
23.0 22.0 19.0
 1 1001 A1001 1 344 100.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 1626.24 1478.40 1626.24 1404.48 1552.32
1552.32 1626.24 1626.24 1478.40 1700.16 1626.24 1404.48 1626.24 1478.40 1626.24 1404.48 1552.32
1552.32 1626.24 1626.24 1478.40 1700.16 1626.24 1404.48 574.80 522.11 574.80 504.15 552.05 549.66
574.80 574.80 526.90 597.56 572.41 506.55 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 232.32 211.20 232.32 200.64 221.76 221.76 232.32 232.32 211.20 242.88
232.32 200.64 309.76 281.60 309.76 267.52 295.68 295.68 309.76 309.76 281.60 323.84 309.76 267.52 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0
10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0
5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0
```

```
280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 3 1 0 0 10.1 0 22.0 20.0 22.0
19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1002 A1002 1 344 200.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 1558.48 1416.80 1558.48 1345.96 1487.64
1487.64 1558.48 1558.48 1416.80 1629.32 1558.48 1345.96 1558.48 1416.80 1558.48 1345.96 1487.64
1487.64 1558.48 1558.48 1416.80 1629.32 1558.48 1345.96 1336.72 1214.11 1336.72 1173.82 1284.42
1278.42 1336.72 1336.72 1226.12 1389.02 1330.71 1179.83 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 267.17 242.88 267.17 230.74 255.02 255.02 267.17 267.17
242.88 279.31 267.17 230.74 356.22 323.84 356.22 307.65 340.03 340.03 356.22 356.22 323.84 372.42
356.22 307.65 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0
10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0
322.0 308.0 266.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 1 1 1 0
13.4 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1003 A1003 1 344 200.00 9.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 1705.01 1550.01 1705.01 1472.51 1627.51
1627.51 1705.01 1705.01 1550.01 1782.51 1705.01 1472.51 1090.09 990.99 1090.09 941.44 1040.54 1040.54
1090.09 1090.09 990.99 1139.64 1090.09 941.44 1231.50 1118.45 1231.50 1082.94 1183.98 1177.98 1231.50
1231.50 1130.46 1279.02 1225.49 1088.94 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 199.65 181.50 199.65 172.42 190.57 190.57 199.65 199.65 181.50 208.72
199.65 172.42 266.20 242.00 266.20 229.90 254.10 254.10 266.20 266.20 242.00 278.30 266.20 229.90 0.63
0.57 0.63 0.54 0.60 0.60 0.63 0.63 0.57 0.65 0.63 0.54 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0
10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0
5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0
280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 1 1 0 0 14.8 0 22.0 20.0 22.0
19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1004 A1004 1 344 200.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 18160.00 16470.00 18160.00 16377.50
17637.50 17422.50 18160.00 18160.00 16900.00 18682.50 17945.00 16592.50 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 2 0 2 0 62.4 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1005 A1005 1 344 400.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 1604.40 1458.55 1604.40 1385.62 1531.48
1531.48 1604.40 1604.40 1458.55 1677.33 1604.40 1385.62 1098.89 998.99 1098.89 949.04 1048.94 1048.94
1098.89 1098.89 998.99 1148.84 1098.89 949.04 3629.57 3296.86 3629.57 3183.44 3485.90 3470.78 3629.57
3629.57 3327.11 3773.25 3614.45 3198.56 27.43 24.91 27.43 24.06 26.34 26.23 27.43 27.43 25.14 28.51
27.31 24.17 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 219.78 199.80 219.78 189.81 209.79 209.79 219.78 219.78
199.80 229.77 219.78 189.81 293.04 266.40 293.04 253.08 279.72 279.72 293.04 293.04 266.40 306.36
293.04 253.08 4.32 3.93 4.32 3.73 4.13 4.13 4.32 4.32 3.93 4.52 4.32 3.73 10.0 10.0 10.0 10.0 10.0 10.0 10.0
10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
```

7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 1 1 0 1 19.8 -1 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1006 A1006 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 3051.84 2774.40 3051.84 2635.68 2913.12 2913.12 3051.84 3051.84 2774.40 3190.56 3051.84 2635.68 5924.16 5385.60 5924.16 5116.32 5654.88 5654.88 5924.16 5924.16 5385.60 6193.44 5924.16 5116.32 4633.50 4209.00 4633.50 4059.75 4448.25 4430.25 4633.50 4633.50 4245.00 4818.75 4615.50 4077.75 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 897.60 816.00 897.60 775.20 856.80 856.80 897.60 897.60 816.00 938.40 897.60 775.20 1305.60 1186.91 1305.60 1127.56 1246.25 1246.25 1305.60 1305.60 1186.91 1364.95 1305.60 1127.56 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 374.0 340.0 374.0 323.0 357.0 357.0 374.0 374.0 340.0 391.0 374.0 323.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 374.0 340.0 374.0 323.0 357.0 357.0 374.0 374.0 340.0 391.0 374.0 323.0 1 1 1 2 2 2 1 1 2 1 1 2 1 1 1 2 2 2 1 1 2 1 1 2 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 1 1 0 1 31.8 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1007 A1007 1 344 100.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 2002.00 1820.00 2002.00 1729.00 1911.00 1911.00 2002.00 2002.00 1820.00 2093.00 2002.00 1729.00 2002.00 1820.00 2002.00 1729.00 1911.00 1911.00 2002.00 2002.00 1820.00 2093.00 2002.00 1729.00 1544.50 1403.00 1544.50 1353.25 1482.75 1476.75 1544.50 1544.50 1415.00 1606.25 1538.50 1359.25 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 286.00 260.00 286.00 247.00 273.00 273.00 286.00 286.00 260.00 299.00 286.00 247.00 286.00 260.00 286.00 247.00 273.00 273.00 286.00 286.00 260.00 299.00 286.00 247.00 2640.00 2400.00 2640.00 2280.00 2520.00 2520.00 2640.00 2640.00 2400.00 2760.00 2640.00 2280.00 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 330.0 300.0 330.0 285.0 315.0 315.0 330.0 330.0 300.0 345.0 330.0 285.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 330.0 300.0 330.0 285.0 315.0 315.0 330.0 330.0 300.0 345.0 330.0 285.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 286.0 260.0 286.0 247.0 273.0 273.0 286.0 286.0 260.0 299.0 286.0 247.0 1 0 1 0 12.0 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1008 A1008 1 344 500.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 2953.91 2668.05 2953.91 2858.62 2953.91 2858.62 2953.91 2953.91 2858.62 2953.91 2858.62 2953.91 5485.84 4954.95 5485.84 5308.87 5485.84 5308.87 5485.84 5485.84 5308.87 5485.84 5308.87 5485.84 14838.50 13402.52 14838.50 14359.84 14838.50 14359.84 14838.50 14838.50 14359.84 14838.50 14359.84 14838.50 6982.82 6307.07 6982.82 6757.57 6982.82 6757.57 6982.82 6982.82 6757.57 6982.82 6757.57 6982.82 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 1172.19 1058.75 1172.19 1134.37 1172.19 1134.37 1172.19 1172.19 1134.37 1172.19 1134.37 1172.19 1507.10 1361.25 1507.10 1458.48 1507.10 1458.48 1507.10 1507.10 1458.48 1507.10 1458.48 1507.10 10.23 9.24 10.23 9.90 10.23 9.90 10.23 10.23 9.90 10.23 9.90 10.23 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 620.0 560.0 620.0 600.0 620.0 600.0 620.0 620.0 600.0 620.0 600.0 620.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 620.0 560.0 620.0 600.0 620.0 600.0 620.0 620.0 600.0 620.0 600.0 620.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 558.0 504.0 558.0 540.0 558.0 540.0 558.0 558.0 540.0 558.0 540.0 558.0 1 1 0 0 35.5 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1009 A1009 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 1279.37 1163.07 1279.37 1104.91 1221.22 1221.22 1279.37 1279.37 1163.07 1337.53 1279.37 1104.91 817.96 743.60 817.96 706.42 780.78 780.78

817.96 817.96 743.60 855.14 817.96 706.42 4969.69 4513.54 4969.69 4369.45 4777.61 4753.62 4969.69 4969.69 4561.53 5161.78 4945.70 4393.45 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 190.67 173.33 190.67 164.67 182.00 182.00 190.67 190.67 173.33 199.33 190.67 164.67 254.22 231.11 254.22 219.56 242.67 242.67 254.22 254.22 231.11 265.78 254.22 219.56 88.00 80.00 88.00 76.00 84.00 84.00 88.00 88.00 80.00 92.00 88.00 76.00 66.0 66.0 66.0 66.0 66.0 66.0 66.0 66.0 66.0 66.0 66.0 66.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 18.0 9.0 9.0 9.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 132.0 120.0 132.0 114.0 126.0 126.0 132.0 132.0 120.0 138.0 132.0 114.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 132.0 120.0 132.0 114.0 126.0 126.0 132.0 132.0 120.0 138.0 132.0 114.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 88.0 80.0 88.0 76.0 84.0 84.0 88.0 88.0 80.0 92.0 88.0 76.0 1 1 0 0 34.1 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1010 A1010 1 344 300.00 0.00 1 1 0 0 3 1 0 3 1 -1 -1 -1 1785.60 1612.80 1785.60 1728.00 1785.60 1728.00 1785.60 1785.60 1728.00 1785.60 1728.00 1785.60 2678.40 2419.20 2678.40 2592.00 2678.40 2592.00 2678.40 2678.40 2592.00 2678.40 2592.00 2678.40 2536.62 2291.14 2536.62 2454.79 2536.62 2454.79 2536.62 2536.62 2454.79 2536.62 2454.79 2536.62 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 248.00 224.00 248.00 240.00 248.00 240.00 248.00 248.00 240.00 248.00 240.00 248.00 248.00 224.00 248.00 240.00 248.00 240.00 248.00 248.00 240.00 248.00 240.00 248.00 6.20 5.60 6.20 6.00 6.20 6.00 6.20 6.20 6.00 6.20 6.00 6.20 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0 496.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0 496.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0 496.0 3 2 2 1 8.6 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1012 A1012 1 344 300.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 2782.56 2513.28 2782.56 2692.80 2782.56 2692.80 2782.56 2782.56 2692.80 2782.56 2692.80 2782.56 2782.56 2513.28 2782.56 2692.80 2782.56 2692.80 2782.56 2782.56 2692.80 2782.56 2692.80 2782.56 18555.04 16759.39 18555.04 17956.49 18555.04 17956.49 18555.04 18555.04 17956.49 18555.04 17956.49 18555.04 976.58 882.07 976.58 945.08 976.58 945.08 976.58 976.58 945.08 976.58 945.08 976.58 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 417.38 376.99 417.38 403.92 417.38 403.92 417.38 417.38 403.92 417.38 403.92 417.38 556.51 502.66 556.51 538.56 556.51 538.56 556.51 556.51 538.56 556.51 538.56 556.51 4960.00 4480.00 4960.00 4800.00 4960.00 4800.00 4960.00 4960.00 4800.00 4960.00 4800.00 4960.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 1 0 0 54.0 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1013 A1013 1 344 300.00 0.00 0 1 0 0 1 0 0 2 2 -1 -1 -1 5824.28 5260.64 5824.28 5636.40 5824.28 5636.40 5824.28 5824.28 5636.40 5824.28 5636.40 5824.28 3723.72 3363.36 3723.72 3603.60 3723.72 3603.60 3723.72 3723.72 3603.60 3723.72 3603.60 3723.72 721.65 651.81 721.65 698.37 721.65 698.37 721.65 721.65 698.37 721.65 698.37 721.65 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 682.00 616.00 682.00 660.00 682.00 660.00 682.00 682.00 660.00 682.00 660.00 682.00 682.00 616.00 682.00 660.00 682.00 660.00 682.00 682.00 660.00 682.00 660.00 682.00 18.60 16.80 18.60 18.00 18.60 18.00 18.60 18.60 18.00 18.60 18.00 18.60 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0

```
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0
434.0 420.0 434.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 1 2 0
17.5 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1014 A1014 1 344 100.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 1566.40 1414.81 1566.40 1515.87 1566.40
1515.87 1566.40 1566.40 1515.87 1566.40 1515.87 1566.40 1097.55 991.33 1097.55 1062.14 1097.55
1062.14 1097.55 1097.55 1062.14 1097.55 1062.14 1097.55 639.58 577.69 639.58 618.95 639.58 618.95
639.58 639.58 618.95 639.58 618.95 639.58 114.64 103.55 114.64 110.94 114.64 110.94 114.64 114.64
110.94 114.64 110.94 114.64 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 21.0 21.0 21.0 21.0
21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 16.50 16.50 16.50 16.50 16.50 16.50 16.50 16.50 16.50 16.50 16.50
16.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 266.39 240.61 266.39 257.80 266.39 257.80
266.39 266.39 257.80 266.39 257.80 266.39 334.90 302.49 334.90 324.09 334.90 324.09 334.90 334.90
324.09 334.90 324.09 334.90 347.20 313.60 347.20 336.00 347.20 336.00 347.20 347.20 336.00 347.20
336.00 347.20 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0
11.0 11.0 11.0 11.0 11.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 403.0 364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0 403.0 390.0 403.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 403.0 364.0 403.0 390.0 403.0
390.0 403.0 403.0 390.0 403.0 390.0 403.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 341.0 308.0 341.0
330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 1 1 0 1 8.4 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0
30.0 31.0 30.0 31.0
 1 1015 A1015 1 344 100.00 0.00 0 1 0 0 1 0 0 2 2 -1 -1 -1 2800.12 2533.44 2800.12 2633.44 2766.78
2700.11 2800.12 2800.12 2666.78 2833.45 2733.45 2700.11 2800.12 2533.44 2800.12 2633.44 2766.78
2700.11 2800.12 2800.12 2666.78 2833.45 2733.45 2700.11 2016.26 1822.47 2016.26 1927.54 2005.92
1948.22 2016.26 2016.26 1937.88 2026.60 1958.56 1985.24 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 400.02 361.92 400.02 376.21 395.25 385.73 400.02 400.02
380.97 404.78 390.49 385.73 410.73 371.44 410.73 389.30 407.16 396.45 410.73 410.73 392.87 414.30
400.02 400.02 3360.00 3040.00 3360.00 3160.00 3320.00 3240.00 3360.00 3360.00 3200.00 3400.00 3280.00
3240.00 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 3.0 3.0 6.0 3.0 3.0 6.0 3.0 3.0 6.0 3.0 3.0
3.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 407.0
368.0 407.0 387.0 404.0 393.0 407.0 407.0 390.0 410.0 396.0 398.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 407.0 368.0 407.0 387.0 404.0 393.0 407.0
407.0 390.0 410.0 396.0 398.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 345.0 312.0 345.0 327.0 342.0
333.0 345.0 345.0 330.0 348.0 336.0 336.0 1 0 1 0 13.8 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0
30.0 31.0
 1 1016 A1016 1 344 100.00 0.00 0 1 0 1 1 0 0 2 1 -1 -1 -1 4057.90 3665.20 4057.90 3927.00 4057.90
3927.00 4057.90 4057.90 3927.00 4057.90 3927.00 4057.90 4057.90 3665.20 4057.90 3927.00 4057.90
3927.00 4057.90 4057.90 3927.00 4057.90 3927.00 4057.90 1034.47 934.36 1034.47 1001.10 1034.47
1001.10 1034.47 1034.47 1001.10 1034.47 1001.10 1034.47 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 579.70 523.60 579.70 561.00 579.70 561.00 579.70 579.70
561.00 579.70 561.00 579.70 604.60 546.09 604.60 585.09 604.60 585.09 604.60 604.60 585.09 604.60
585.09 604.60 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0
5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 589.0 532.0 589.0 570.0 589.0 570.0 589.0 589.0 570.0 589.0 570.0
589.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
589.0 532.0 589.0 570.0 589.0 570.0 589.0 589.0 570.0 589.0 570.0 589.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 527.0 476.0 527.0 510.0 527.0 510.0 527.0 527.0 510.0 527.0 510.0 527.0 3 1 0 0 10.0 0 31.0 28.0
31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1017 A1017 1 344 500.00 0.00 0 0 0 0 1 0 0 2 2 -1 -1 -1 2953.91 2668.05 2953.91 2858.62 2953.91
2858.62 2953.91 2953.91 2858.62 2953.91 2858.62 2953.91 5485.84 4954.95 5485.84 5308.87 5485.84
```

5308.87 5485.84 5485.84 5308.87 5485.84 5308.87 5485.84 14838.50 13402.52 14838.50 14359.84 14838.50 14359.84 14838.50 14838.50 14359.84 14838.50 14359.84 14838.50 6982.82 6307.07 6982.82 6757.57 6982.82 6757.57 6982.82 6982.82 6757.57 6982.82 6757.57 6982.82 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 1172.19 1058.75 1172.19 1134.37 1172.19 1134.37 1172.19 1172.19 1134.37 1172.19 1134.37 1172.19 1507.10 1361.25 1507.10 1458.48 1507.10 1458.48 1507.10 1507.10 1458.48 1507.10 1458.48 1507.10 10.23 9.24 10.23 9.90 10.23 9.90 10.23 10.23 9.90 10.23 9.90 10.23 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 620.0 560.0 620.0 600.0 620.0 600.0 620.0 620.0 600.0 620.0 600.0 620.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 620.0 560.0 620.0 600.0 620.0 600.0 620.0 620.0 600.0 620.0 600.0 620.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 558.0 504.0 558.0 540.0 558.0 540.0 558.0 558.0 540.0 558.0 540.0 558.0 1 1 0 0 35.5 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1018 A1018 1 344 150.00 10.00 0 1 0 0 1 0 0 2 2 -1 -1 -1 3605.51 3256.59 3605.51 3489.20 3605.51 3489.20 3605.51 3605.51 3489.20 3605.51 3489.20 3605.51 2305.16 2082.08 2305.16 2230.80 2305.16 2230.80 2305.16 2305.16 2230.80 2305.16 2230.80 2305.16 6698.43 6050.20 6698.43 6482.35 6698.43 6482.35 6698.43 6698.43 6482.35 6698.43 6482.35 6698.43 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 537.33 485.33 537.33 520.00 537.33 520.00 537.33 537.33 520.00 537.33 520.00 537.33 929.44 839.50 929.44 899.46 929.44 899.46 929.44 929.44 899.46 929.44 899.46 929.44 248.00 224.00 248.00 240.00 248.00 240.00 248.00 248.00 240.00 248.00 240.00 248.00 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 558.0 504.0 558.0 540.0 558.0 540.0 558.0 558.0 540.0 558.0 540.0 558.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 558.0 504.0 558.0 540.0 558.0 540.0 558.0 558.0 540.0 558.0 540.0 558.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0 496.0 1 1 0 0 27.3 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1019 A1019 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 4804.09 4346.56 4804.09 4518.13 4746.90 4632.52 4804.09 4804.09 4575.33 4861.28 4689.71 4632.52 9325.59 8437.44 9325.59 8770.50 9214.57 8992.53 9325.59 9325.59 8881.52 9436.61 9103.55 8992.53 6045.06 5464.05 6045.06 5779.06 6014.06 5841.05 6045.06 6045.06 5810.05 6076.06 5872.05 5952.06 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 1412.97 1278.40 1412.97 1328.86 1396.15 1362.51 1412.97 1412.97 1345.68 1429.79 1379.33 1362.51 1450.82 1312.04 1450.82 1375.12 1438.20 1400.35 1450.82 1450.82 1387.74 1463.43 1412.97 1412.97 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 3.0 3.0 6.0 3.0 3.0 6.0 3.0 3.0 6.0 3.0 3.0 3.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 407.0 368.0 407.0 387.0 404.0 393.0 407.0 407.0 390.0 410.0 396.0 398.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 407.0 368.0 407.0 387.0 404.0 393.0 407.0 407.0 390.0 410.0 396.0 398.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 345.0 312.0 345.0 327.0 342.0 333.0 345.0 345.0 330.0 348.0 336.0 336.0 1 1 0 1 30.5 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1020 A1020 1 344 300.00 0.00 0 0 0 0 1 0 0 2 2 -1 -1 -1 3375.90 3049.20 3375.90 3267.00 3375.90 3267.00 3375.90 3375.90 3267.00 3375.90 3267.00 3375.90 3375.90 3049.20 3375.90 3267.00 3375.90 3267.00 3375.90 3375.90 3267.00 3375.90 3267.00 3375.90 17154.62 15494.50 17154.62 16601.25 17154.62 16601.25 17154.62 17154.62 16601.25 17154.62 16601.25 17154.62 902.87 815.50 902.87 873.75 902.87 873.75 902.87 902.87 873.75 902.87 873.75 902.87 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 450.12 406.56 450.12 435.60 450.12 435.60 450.12 450.12 435.60 450.12 435.60 450.12 450.12 406.56 450.12 435.60 450.12 435.60 450.12 450.12 435.60 450.12 435.60 450.12 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0

8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0
6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0
336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0
6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0
372.0 360.0 372.0 360.0 372.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 310.0 280.0 310.0 300.0 310.0
300.0 310.0 310.0 300.0 310.0 300.0 310.0 1 1 0 0 62.9 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
1 1021 A1021 1 344 200.00 0.00 1 0 0 0 1 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 37200.00 33600.00 37200.00 36000.00
37200.00 36000.00 37200.00 37200.00 36000.00 37200.00 36000.00 37200.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 2 0 2 0 59.9 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
1 1022 A1022 1 344 200.00 9.00 0 0 0 0 1 0 0 2 2 -1 -1 -1 4513.82 4077.00 4513.82 4368.21 4513.82
4368.21 4513.82 4513.82 4368.21 4513.82 4368.21 4513.82 2885.88 2606.60 2885.88 2792.79 2885.88
2792.79 2885.88 2885.88 2792.79 2885.88 2792.79 2885.88 2444.68 2208.10 2444.68 2365.82 2444.68
2365.82 2444.68 2444.68 2365.82 2444.68 2365.82 2444.68 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 528.55 477.40 528.55 511.50 528.55 511.50 528.55 528.55
511.50 528.55 511.50 528.55 551.25 497.90 551.25 533.47 551.25 533.47 551.25 551.25 533.47 551.25
533.47 551.25 0.92 0.83 0.92 0.89 0.92 0.89 0.92 0.92 0.89 0.92 0.89 0.92 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0
5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 589.0 532.0 589.0 570.0 589.0 570.0 589.0 589.0 570.0 589.0 570.0
589.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
589.0 532.0 589.0 570.0 589.0 570.0 589.0 589.0 570.0 589.0 570.0 589.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 527.0 476.0 527.0 510.0 527.0 510.0 527.0 527.0 510.0 527.0 510.0 527.0 1 1 0 0 13.7 0 31.0 28.0
31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
1 1023 A1023 1 344 200.00 0.00 0 0 0 0 1 0 0 2 1 -1 -1 -1 3580.50 3234.00 3580.50 3465.00 3580.50
3465.00 3580.50 3580.50 3465.00 3580.50 3465.00 3580.50 3580.50 3234.00 3580.50 3465.00 3580.50
3465.00 3580.50 3580.50 3465.00 3580.50 3465.00 3580.50 2395.07 2163.29 2395.07 2317.81 2395.07
2317.81 2395.07 2395.07 2317.81 2395.07 2317.81 2395.07 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 613.80 554.40 613.80 594.00 613.80 594.00 613.80 613.80
594.00 613.80 594.00 613.80 613.80 554.40 613.80 594.00 613.80 594.00 613.80 613.80 594.00 613.80
594.00 613.80 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0
8.0 8.0 8.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0
496.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0 496.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 1 1 1 0 13.6 0 31.0 28.0
31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
1 1024 A1024 1 344 300.00 0.00 0 1 0 0 1 0 0 2 2 -1 -1 -1 2823.88 2550.60 2823.88 2732.78 2823.88
2732.78 2823.88 2823.88 2732.78 2823.88 2732.78 2823.88 5481.64 4951.16 5481.64 5304.82 5481.64
5304.82 5481.64 5481.64 5304.82 5481.64 5304.82 5481.64 1488.00 1344.00 1488.00 1440.00 1488.00
1440.00 1488.00 1488.00 1440.00 1488.00 1440.00 1488.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0
28.0 28.0 28.0 28.0 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 778.64 703.29 778.64 753.53 778.64 753.53 778.64 778.64

```
753.53 778.64 753.53 778.64 778.64 703.29 778.64 753.53 778.64 753.53 778.64 778.64 753.53 778.64
753.53 778.64 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0
10.0 10.0 10.0 10.0 10.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0
434.0 420.0 434.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 1 2 0 9.6
0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1025 A1025 1 344 300.00 0.00 0 1 0 0 1 0 0 2 2 -1 -1 -1 1412.97 1278.40 1412.97 1328.86 1396.15
1362.51 1412.97 1412.97 1345.68 1429.79 1379.33 1362.51 2742.82 2481.60 2742.82 2579.56 2710.17
2644.86 2742.82 2742.82 2612.21 2775.47 2677.52 2644.86 807.25 729.66 807.25 771.73 803.11 780.01
807.25 807.25 775.87 811.39 784.15 794.83 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0
25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0
16.0 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 138.53 125.33 138.53 130.28 136.88 133.58 138.53 138.53 131.93 140.18
135.23 133.58 142.24 128.63 142.24 134.82 141.00 137.29 142.24 142.24 136.05 143.47 138.53 138.53 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0
13.0 3.0 3.0 6.0 3.0 3.0 6.0 3.0 3.0 6.0 3.0 3.0 3.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 407.0 368.0 407.0 387.0 404.0 393.0 407.0 407.0 390.0 410.0 396.0 398.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 407.0
368.0 407.0 387.0 404.0 393.0 407.0 407.0 390.0 410.0 396.0 398.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 345.0 312.0 345.0 327.0 342.0 333.0 345.0 345.0 330.0 348.0 336.0 336.0 3 2 2 1 6.4 -1 31.0 28.0 31.0
30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1026 A1026 1 344 50.00 0.00 1 0 0 0 1 0 0 2 1 -1 -1 -1 3341.80 3018.40 3341.80 3234.00 3341.80 3234.00
3341.80 3341.80 3234.00 3341.80 3234.00 3341.80 3341.80 3018.40 3341.80 3234.00 3341.80 3234.00
3341.80 3341.80 3234.00 3341.80 3234.00 3341.80 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00
16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 477.40
431.20 477.40 462.00 477.40 462.00 477.40 477.40 462.00 477.40 462.00 477.40 477.40 431.20 477.40
462.00 477.40 462.00 477.40 477.40 462.00 477.40 462.00 477.40 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 5.0 5.0
5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 496.0 448.0
496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0 496.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0
480.0 496.0 480.0 496.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 434.0 392.0 434.0 420.0 434.0 420.0
434.0 434.0 420.0 434.0 420.0 434.0 3 0 2 0 7.7 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1028 A1028 1 344 400.00 0.00 1 0 0 0 1 0 0 2 2 -1 -1 -1 2160.27 1951.22 2160.27 2090.59 2160.27
2090.59 2160.27 2160.27 2090.59 2160.27 2090.59 2160.27 1479.62 1336.43 1479.62 1431.89 1479.62
1431.89 1479.62 1479.62 1431.89 1479.62 1431.89 1479.62 5294.63 4782.24 5294.63 5123.83 5294.63
5123.83 5294.63 5294.63 5123.83 5294.63 5123.83 5294.63 40.01 36.14 40.01 38.72 40.01 38.72 40.01 40.01
38.72 40.01 38.72 40.01 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 295.93 267.29 295.93 286.38 295.93 286.38
295.93 295.93 286.38 295.93 286.38 295.93 394.57 356.38 394.57 381.84 394.57 381.84 394.57 394.57
381.84 394.57 381.84 394.57 6.85 6.19 6.85 6.63 6.85 6.63 6.85 6.85 6.63 6.85 6.63 6.85 10.0 10.0 10.0 10.0
10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0
5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 434.0 392.0 434.0 420.0 434.0 420.0
434.0 434.0 420.0 434.0 420.0 434.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0
360.0 372.0 1 1 0 1 20.3 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1029 A1029 1 344 100.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 2562.05 2314.11 2562.05 2479.40 2562.05
```

2479.40 2562.05 2562.05 2479.40 2562.05 2479.40 2562.05 2562.05 2314.11 2562.05 2479.40 2562.05 2479.40 2562.05 2562.05 2479.40 2562.05 2479.40 2562.05 978.91 884.18 978.91 947.33 978.91 947.33 978.91 978.91 947.33 978.91 947.33 978.91 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 366.01 330.59 366.01 354.20 366.01 354.20 366.01 366.01 354.20 366.01 354.20 366.01 633.09 571.83 633.09 612.67 633.09 612.67 633.09 633.09 612.67 633.09 612.67 633.09 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 558.0 504.0 558.0 540.0 558.0 540.0 558.0 558.0 540.0 558.0 540.0 558.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 558.0 504.0 558.0 540.0 558.0 540.0 558.0 558.0 540.0 558.0 540.0 558.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0 496.0 3 1 0 0 10.8 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1030 A1030 1 344 150.00 10.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 3848.18 3475.78 3848.18 3724.05 3848.18 3724.05 3848.18 3848.18 3724.05 3848.18 3724.05 3848.18 2460.32 2222.22 2460.32 2380.95 2460.32 2380.95 2460.32 2460.32 2380.95 2460.32 2380.95 2460.32 9778.73 8832.41 9778.73 9463.29 9778.73 9463.29 9778.73 9778.73 9463.29 9778.73 9463.29 9778.73 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 573.50 518.00 573.50 555.00 573.50 555.00 573.50 573.50 555.00 573.50 555.00 573.50 992.00 896.00 992.00 960.00 992.00 960.00 992.00 992.00 960.00 992.00 960.00 992.00 176.46 159.38 176.46 170.77 176.46 170.77 176.46 176.46 170.77 176.46 170.77 176.46 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 558.0 504.0 558.0 540.0 558.0 540.0 558.0 558.0 540.0 558.0 540.0 558.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 558.0 504.0 558.0 540.0 558.0 540.0 558.0 558.0 540.0 558.0 540.0 558.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0 496.0 1 1 0 0 32.3 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1031 A1031 1 344 500.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 3007.62 2716.56 3007.62 2910.60 3007.62 2910.60 3007.62 3007.62 2910.60 3007.62 2910.60 3007.62 5585.58 5045.04 5585.58 5405.40 5585.58 5405.40 5585.58 5585.58 5405.40 5585.58 5405.40 5585.58 14838.50 13402.52 14838.50 14359.84 14838.50 14359.84 14838.50 14838.50 14359.84 14838.50 14359.84 14838.50 6982.82 6307.07 6982.82 6757.57 6982.82 6757.57 6982.82 6982.82 6757.57 6982.82 6757.57 6982.82 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 1193.50 1078.00 1193.50 1155.00 1193.50 1155.00 1193.50 1193.50 1155.00 1193.50 1155.00 1193.50 1534.50 1386.00 1534.50 1485.00 1534.50 1485.00 1534.50 1534.50 1485.00 1534.50 1485.00 1534.50 10.23 9.24 10.23 9.90 10.23 9.90 10.23 10.23 9.90 10.23 9.90 10.23 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 620.0 560.0 620.0 600.0 620.0 600.0 620.0 620.0 600.0 620.0 600.0 620.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 620.0 560.0 620.0 600.0 620.0 600.0 620.0 620.0 600.0 620.0 600.0 620.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 558.0 504.0 558.0 540.0 558.0 540.0 558.0 558.0 540.0 558.0 540.0 558.0 1 1 0 0 35.7 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1032 A1032 1 344 50.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 2148.30 1940.40 2148.30 2079.00 2148.30 2079.00 2148.30 2148.30 2079.00 2148.30 2079.00 2148.30 2148.30 1940.40 2148.30 2079.00 2148.30 2079.00 2148.30 2148.30 2079.00 2148.30 2079.00 2148.30 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 306.90 277.20 306.90 297.00 306.90 297.00 306.90 306.90 297.00 306.90 297.00 306.90 345.26 311.85 345.26 334.12 345.26 334.12 345.26 345.26 334.12 345.26 334.12 345.26 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0

```
17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0
341.0 341.0 330.0 341.0 330.0 341.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0
279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 3 0 2 0 8.7 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0
31.0 30.0 31.0
 1 1033 A1033 1 344 750.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 536.41 484.50 536.41 519.11 536.41 519.11
536.41 536.41 519.11 536.41 519.11 536.41 1041.27 940.50 1041.27 1007.68 1041.27 1007.68 1041.27
1041.27 1007.68 1041.27 1007.68 1041.27 360.78 325.87 360.78 349.15 360.78 349.15 360.78 360.78 349.15
360.78 349.15 360.78 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0
25.0 25.0 25.0 25.0 25.0 25.0 25.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00
17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 63.11 57.00 63.11 61.07 63.11 61.07 63.11 63.11 61.07 63.11 61.07 63.11 77.96 70.41 77.96
75.44 77.96 75.44 77.96 77.96 75.44 77.96 75.44 77.96 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 22.0 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 14.0
14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 217.0 196.0 217.0 210.0 217.0 210.0 217.0 217.0 210.0 217.0 210.0 217.0 1 0 0 0 4.8 0 31.0
28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1034 A1034 1 344 200.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 30477.03 27527.64 30477.03 29493.90
30477.03 29493.90 30477.03 30477.03 29493.90 30477.03 29493.90 30477.03 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 2 0 2 0 64.3 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1036 A1036 1 344 200.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 3126.97 2824.36 3126.97 3026.10 3126.97
3026.10 3126.97 3126.97 3026.10 3126.97 3026.10 3126.97 3126.97 2824.36 3126.97 3026.10 3126.97
3026.10 3126.97 3126.97 3026.10 3126.97 3026.10 3126.97 2444.68 2208.10 2444.68 2365.82 2444.68
2365.82 2444.68 2444.68 2365.82 2444.68 2365.82 2444.68 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 536.05 484.18 536.05 518.76 536.05 518.76 536.05 536.05
518.76 536.05 518.76 536.05 559.07 504.97 559.07 541.04 559.07 541.04 559.07 559.07 541.04 559.07
541.04 559.07 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0
5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 589.0 532.0 589.0 570.0 589.0 570.0 589.0 589.0 570.0 589.0 570.0
589.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
589.0 532.0 589.0 570.0 589.0 570.0 589.0 589.0 570.0 589.0 570.0 589.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 527.0 476.0 527.0 510.0 527.0 510.0 527.0 527.0 510.0 527.0 510.0 527.0 1 1 1 0 10.9 0 31.0 28.0
31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1038 A1038 1 344 400.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 1758.36 1588.20 1758.36 1701.64 1758.36
1701.64 1758.36 1758.36 1701.64 1758.36 1701.64 1758.36 1204.34 1087.79 1204.34 1165.49 1204.34
1165.49 1204.34 1204.34 1165.49 1204.34 1165.49 1204.34 4181.00 3776.38 4181.00 4046.12 4181.00
4046.12 4181.00 4181.00 4046.12 4181.00 4046.12 4181.00 31.59 28.54 31.59 30.58 31.59 30.58 31.59 31.59
30.58 31.59 30.58 31.59 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00
```

```
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 240.87 217.56 240.87 233.10 240.87 233.10
240.87 240.87 233.10 240.87 233.10 240.87 373.18 337.06 373.18 361.14 373.18 361.14 373.18 373.18
361.14 373.18 361.14 373.18 6.18 5.58 6.18 5.98 6.18 5.98 6.18 6.18 5.98 6.18 5.98 6.18 11.0 11.0 11.0 11.0
11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0
5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 403.0 364.0 403.0 390.0 403.0 390.0
403.0 403.0 390.0 403.0 390.0 403.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 403.0 364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0 403.0 390.0 403.0 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0
330.0 341.0 1 1 0 1 20.6 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1039 A1039 1 344 100.00 0.00 0 1 0 1 3 0 0 3 1 -1 -1 -1 1524.60 1386.00 1524.60 1316.70 1455.30
1455.30 1524.60 1524.60 1386.00 1593.90 1524.60 1316.70 1524.60 1386.00 1524.60 1316.70 1455.30
1455.30 1524.60 1524.60 1386.00 1593.90 1524.60 1316.70 451.10 409.66 451.10 397.34 433.98 431.58
451.10 451.10 414.46 468.22 448.70 399.75 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 217.80 198.00 217.80 188.10 207.90 207.90 217.80 217.80 198.00 227.70
217.80 188.10 217.80 198.00 217.80 188.10 207.90 207.90 217.80 217.80 198.00 227.70 217.80 188.10 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0
200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 176.0 160.0 176.0 152.0 168.0 168.0 176.0 176.0 160.0 184.0 176.0 152.0 3 1 0 0 10.9 0 22.0 20.0 22.0
19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1040 A1040 1 344 150.00 0.00 0 1 0 0 3 0 0 3 2 -1 -1 -1 787.31 715.73 787.31 679.95 751.52 751.52
787.31 787.31 715.73 823.09 787.31 679.95 503.36 457.60 503.36 434.72 480.48 480.48 503.36 503.36
457.60 526.24 503.36 434.72 1879.02 1703.84 1879.02 1700.27 1827.44 1803.43 1879.02 1879.02 1751.85
1930.61 1855.02 1724.27 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0
25.0 25.0 25.0 25.0 25.0 25.0 25.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 17.50 17.50
17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 117.33 106.67 117.33 101.33 112.00 112.00 117.33 117.33 106.67 122.67 117.33 101.33
156.44 142.22 156.44 135.11 149.33 149.33 156.44 156.44 142.22 163.56 156.44 135.11 176.00 160.00
176.00 152.00 168.00 168.00 176.00 176.00 160.00 184.00 176.00 152.00 18.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0
11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 11.0 11.0 11.0 11.0 11.0 11.0
11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 9.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 132.0 120.0 132.0 114.0
126.0 126.0 132.0 132.0 120.0 138.0 132.0 114.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 132.0 120.0 132.0 114.0 126.0 126.0 132.0 132.0 120.0 138.0
132.0 114.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 88.0 80.0 88.0 76.0 84.0 84.0 88.0 88.0 80.0 92.0
88.0 76.0 1 1 0 0 23.8 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1041 A1041 1 344 500.00 0.00 0 0 0 0 3 0 0 3 2 -1 -1 -1 884.27 803.88 884.27 763.69 844.07 844.07
884.27 884.27 803.88 924.46 884.27 763.69 1642.21 1492.92 1642.21 1418.27 1567.57 1567.57 1642.21
1642.21 1492.92 1716.86 1642.21 1418.27 6696.64 6081.92 6696.64 5888.80 6438.24 6405.60 6696.64
6696.64 6147.20 6955.04 6664.00 5921.44 3151.36 2862.08 3151.36 2771.20 3029.76 3014.40 3151.36
3151.36 2892.80 3272.96 3136.00 2786.56 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50
14.50 14.50 14.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 350.90 319.00 350.90 303.05
334.95 334.95 350.90 350.90 319.00 366.85 350.90 303.05 467.87 425.33 467.87 404.07 446.60 446.60
467.87 467.87 425.33 489.13 467.87 404.07 9.68 8.80 9.68 8.36 9.24 9.24 9.68 9.68 8.80 10.12 9.68 8.36 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 11.0 11.0
11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 132.0 120.0 132.0 114.0 126.0 126.0 132.0 132.0 120.0 138.0 132.0 114.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 132.0 120.0 132.0 114.0 126.0
126.0 132.0 132.0 120.0 138.0 132.0 114.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 88.0 80.0 88.0 76.0
```

```
84.0 84.0 88.0 88.0 80.0 92.0 88.0 76.0 1 1 0 0 47.1 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0
19.0
 1 1042 A1042 1 344 300.00 0.00 1 1 0 0 3 1 0 3 1 -1 -1 -1 1785.60 1612.80 1785.60 1728.00 1785.60
1728.00 1785.60 1785.60 1728.00 1785.60 1728.00 1785.60 2678.40 2419.20 2678.40 2592.00 2678.40
2592.00 2678.40 2678.40 2592.00 2678.40 2592.00 2678.40 2536.62 2291.14 2536.62 2454.79 2536.62
2454.79 2536.62 2536.62 2454.79 2536.62 2454.79 2536.62 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 248.00 224.00 248.00 240.00 248.00 240.00 248.00 248.00
240.00 248.00 240.00 248.00 248.00 224.00 248.00 240.00 248.00 240.00 248.00 248.00 240.00 248.00
240.00 248.00 6.20 5.60 6.20 6.00 6.20 6.00 6.20 6.20 6.00 6.20 6.00 6.20 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0
8.0 8.0 8.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0
496.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0 496.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0 496.0 3 2 2 1 8.6 -1 31.0 28.0
31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1043 A1043 1 344 300.00 0.00 1 1 0 0 3 1 0 3 1 -1 -1 -1 1785.60 1612.80 1785.60 1728.00 1785.60
1728.00 1785.60 1785.60 1728.00 1785.60 1728.00 1785.60 2678.40 2419.20 2678.40 2592.00 2678.40
2592.00 2678.40 2678.40 2592.00 2678.40 2592.00 2678.40 2536.62 2291.14 2536.62 2454.79 2536.62
2454.79 2536.62 2536.62 2454.79 2536.62 2454.79 2536.62 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 248.00 224.00 248.00 240.00 248.00 240.00 248.00 248.00
240.00 248.00 240.00 248.00 248.00 224.00 248.00 240.00 248.00 240.00 248.00 248.00 240.00 248.00
240.00 248.00 6.20 5.60 6.20 6.00 6.20 6.00 6.20 6.20 6.00 6.20 6.00 6.20 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0
8.0 8.0 8.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0
496.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0 496.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0 496.0 3 2 2 1 8.6 -1 31.0 28.0
31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1044 A1044 1 344 200.00 0.00 1 0 0 0 3 0 0 3 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23400.00 21150.00 23400.00 22387.50
23287.50 22612.50 23400.00 23400.00 22500.00 23512.50 22725.00 23062.50 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 2 0 2 0 59.9 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1045 A1045 1 344 200.00 9.00 0 0 0 0 3 0 0 3 2 -1 -1 -1 1653.34 1503.04 1653.34 1427.89 1578.19
1578.19 1653.34 1653.34 1503.04 1728.50 1653.34 1427.89 1057.06 960.96 1057.06 912.91 1009.01 1009.01
1057.06 1057.06 960.96 1105.10 1057.06 912.91 1125.74 1022.31 1125.74 991.58 1083.02 1077.03 1125.74
1125.74 1034.31 1168.47 1119.75 997.58 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 193.60 176.00 193.60 167.20 184.80 184.80 193.60 193.60 176.00 202.40
193.60 167.20 193.60 176.00 193.60 167.20 184.80 184.80 193.60 193.60 176.00 202.40 193.60 167.20 1.32
1.20 1.32 1.14 1.26 1.26 1.32 1.32 1.20 1.38 1.32 1.14 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0
```

16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 176.0 160.0 176.0 152.0 168.0 168.0 176.0 176.0 160.0 184.0 176.0 152.0 1 1 0 0 15.0 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1046 A1046 1 344 50.00 0.00 1 0 0 0 3 0 0 3 1 -1 -1 -1 1524.60 1386.00 1524.60 1316.70 1455.30 1455.30 1524.60 1524.60 1386.00 1593.90 1524.60 1316.70 1524.60 1386.00 1524.60 1316.70 1455.30 1455.30 1524.60 1524.60 1386.00 1593.90 1524.60 1316.70 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 217.80 198.00 217.80 188.10 207.90 207.90 217.80 217.80 198.00 227.70 217.80 188.10 217.80 198.00 217.80 188.10 207.90 207.90 217.80 217.80 198.00 227.70 217.80 188.10 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 176.0 160.0 176.0 152.0 168.0 168.0 176.0 176.0 160.0 184.0 176.0 152.0 3 0 2 0 8.7 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1049 A1049 1 344 200.00 0.00 0 0 0 0 3 0 0 3 1 -1 -1 -1 1524.60 1386.00 1524.60 1316.70 1455.30 1455.30 1524.60 1524.60 1386.00 1593.90 1524.60 1316.70 1524.60 1386.00 1524.60 1316.70 1455.30 1455.30 1524.60 1524.60 1386.00 1593.90 1524.60 1316.70 1125.74 1022.31 1125.74 991.58 1083.02 1077.03 1125.74 1125.74 1034.31 1168.47 1119.75 997.58 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 261.36 237.60 261.36 225.72 249.48 249.48 261.36 261.36 237.60 273.24 261.36 225.72 261.36 237.60 261.36 225.72 249.48 249.48 261.36 261.36 237.60 273.24 261.36 225.72 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 176.0 160.0 176.0 152.0 168.0 168.0 176.0 176.0 160.0 184.0 176.0 152.0 3 0 2 0 8.7 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1050 A1050 1 344 300.00 0.00 1 1 0 0 3 1 0 3 1 -1 -1 -1 1785.60 1612.80 1785.60 1728.00 1785.60 1728.00 1785.60 1785.60 1728.00 1785.60 1728.00 1785.60 2678.40 2419.20 2678.40 2592.00 2678.40 2592.00 2678.40 2678.40 2592.00 2678.40 2592.00 2678.40 2536.62 2291.14 2536.62 2454.79 2536.62 2454.79 2536.62 2536.62 2454.79 2536.62 2454.79 2536.62 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 248.00 224.00 248.00 240.00 248.00 240.00 248.00 248.00 240.00 248.00 240.00 248.00 248.00 224.00 248.00 240.00 248.00 240.00 248.00 248.00 240.00 248.00 240.00 248.00 6.20 5.60 6.20 6.00 6.20 6.00 6.20 6.20 6.00 6.20 6.00 6.20 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0 496.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0 496.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0 496.0 3 2 2 1 8.6 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1052 A1052 1 344 400.00 0.00 1 0 0 0 3 0 0 3 2 -1 -1 -1 1559.84 1418.03 1559.84 1347.13 1488.94 1488.94 1559.84 1559.84 1418.03 1630.74 1559.84 1347.13 1068.36 971.24 1068.36 922.68 1019.80 1019.80 1068.36 1068.36 971.24 1116.93 1068.36 922.68 3367.58 3058.68 3367.58 2957.19 3235.82 3220.69 3367.58 3367.58 3088.94 3499.33 3352.45 2972.32 25.45 23.11 25.45 22.35 24.45 24.34 25.45 25.45 23.34 26.44

25.33 22.46 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 213.68 194.25 213.68 184.54 203.96 203.96 213.68 213.68 194.25 223.39 213.68 184.54 284.90 259.00 284.90 246.05 271.95 271.95 284.90 284.90 259.00 297.85 284.90 246.05 4.86 4.42 4.86 4.20 4.64 4.64 4.86 4.86 4.42 5.08 4.86 4.20 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 1 1 0 1 21.7 -1 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1053 A1053 1 344 100.00 0.00 0 1 0 1 3 0 0 3 1 -1 -1 -1 2046.00 1848.00 2046.00 1980.00 2046.00 1980.00 2046.00 2046.00 1980.00 2046.00 1980.00 2046.00 2046.00 1848.00 2046.00 1980.00 2046.00 1980.00 2046.00 2046.00 1980.00 2046.00 1980.00 2046.00 629.27 568.37 629.27 608.97 629.27 608.97 629.27 629.27 608.97 629.27 608.97 629.27 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 292.29 264.00 292.29 282.86 292.29 282.86 292.29 292.29 282.86 292.29 282.86 292.29 292.29 264.00 292.29 282.86 292.29 282.86 292.29 292.29 282.86 292.29 282.86 292.29 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 310.0 280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 3 1 0 0 8.5 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1054 A1054 1 344 500.00 0.00 0 0 0 0 3 0 0 3 2 -1 -1 -1 1530.41 1381.20 1530.41 1500.72 1539.00 1483.54 1530.41 1530.41 1492.13 1521.82 1474.94 1556.19 2842.19 2565.08 2842.19 2787.06 2858.15 2755.14 2842.19 2842.19 2771.10 2826.23 2739.18 2890.07 8564.83 7735.97 8564.83 8288.54 8564.83 8288.54 8564.83 8564.83 8288.54 8564.83 8288.54 8564.83 4030.51 3640.46 4030.51 3900.49 4030.51 3900.49 4030.51 4030.51 3900.49 4030.51 3900.49 4030.51 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 607.31 548.09 607.31 595.52 610.72 588.70 607.31 607.31 592.11 603.89 585.29 617.54 784.32 706.82 784.32 787.42 796.72 762.62 784.32 784.32 775.02 771.92 750.22 821.52 10.77 9.72 10.77 10.57 10.84 10.44 10.77 10.77 10.51 10.71 10.38 10.96 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 7.0 7.0 6.0 5.0 6.0 6.0 5.0 6.0 5.0 7.0 7.0 5.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 315.0 284.0 315.0 314.0 319.0 306.0 315.0 315.0 310.0 311.0 302.0 327.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 315.0 284.0 315.0 314.0 319.0 306.0 315.0 315.0 310.0 311.0 302.0 327.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 253.0 228.0 253.0 254.0 257.0 246.0 253.0 253.0 250.0 249.0 242.0 265.0 1 1 0 0 33.5 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1055 A1055 1 344 150.00 0.00 0 1 0 0 3 0 0 3 2 -1 -1 -1 2017.64 1825.48 2017.64 1897.54 1993.62 1945.58 2017.64 2017.64 1921.56 2041.66 1969.60 1945.58 1289.97 1167.11 1289.97 1213.18 1274.61 1243.90 1289.97 1289.97 1228.54 1305.32 1259.25 1243.90 6212.33 5611.14 6212.33 6011.94 6212.33 6011.94 6212.33 6212.33 6011.94 6212.33 6011.94 6212.33 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 300.69 272.05 300.69 282.79 297.11 289.95 300.69 300.69 286.37 304.27 293.53 289.95 572.75 515.47 572.75 587.06 587.06 558.43 572.75 572.75 572.75 558.43 544.11 615.70 224.00 202.67 224.00 210.67 221.33 216.00 224.00 224.00 213.33 226.67 218.67 216.00 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 9.0 9.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 7.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0

```
14.0 222.0 200.0 222.0 224.0 226.0 216.0 222.0 222.0 220.0 218.0 212.0 234.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 222.0 200.0 222.0 224.0 226.0
216.0 222.0 222.0 220.0 218.0 212.0 234.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 160.0 144.0 160.0
164.0 164.0 156.0 160.0 160.0 160.0 156.0 152.0 172.0 1 1 0 0 42.7 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0
30.0 31.0 30.0 31.0
 1 1056 A1056 1 344 200.00 0.00 1 0 0 0 3 0 0 3 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23747.40 21449.27 23747.40 22981.36
23747.40 22981.36 23747.40 23747.40 22981.36 23747.40 22981.36 23747.40 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 2 0 2 0 57.9 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1057 A1057 1 344 200.00 9.00 0 0 0 0 2 0 0 3 2 -1 -1 -1 2514.56 2273.70 2514.56 2466.38 2524.19
2437.48 2524.19 2514.56 2456.75 2504.92 2427.85 2553.09 1607.67 1453.68 1607.67 1576.87 1613.83
1558.39 1613.83 1607.67 1570.71 1601.51 1552.23 1632.31 1371.63 1219.23 1371.63 1270.03 1371.63
1320.83 1320.83 1371.63 1270.03 1371.63 1320.83 1320.83 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 294.44 266.24 294.44 288.80 295.57 285.42 295.57 294.44
287.68 293.32 284.29 298.96 311.37 279.78 311.37 309.11 315.88 302.34 309.11 311.37 304.60 306.85
297.83 322.65 3.92 3.54 3.92 3.84 3.93 3.80 3.93 3.92 3.83 3.90 3.78 3.98 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 14.0 7.0 7.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 5.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 346.0 312.0 346.0 344.0 350.0 336.0 346.0 346.0 340.0
342.0 332.0 358.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0
16.0 16.0 16.0 346.0 312.0 346.0 344.0 350.0 336.0 346.0 346.0 340.0 342.0 332.0 358.0 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 276.0 248.0 276.0 274.0 280.0 268.0 274.0 276.0 270.0 272.0 264.0 286.0 1 1 0 0
14.4 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1059 A1059 1 344 50.00 0.00 1 0 0 0 3 0 0 3 1 -1 -1 -1 2250.60 2032.80 2250.60 2178.00 2250.60 2178.00
2250.60 2250.60 2178.00 2250.60 2178.00 2250.60 2250.60 2032.80 2250.60 2178.00 2250.60 2178.00
2250.60 2250.60 2178.00 2250.60 2178.00 2250.60 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00
16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 321.51
290.40 321.51 311.14 321.51 311.14 321.51 321.51 311.14 321.51 311.14 321.51 321.51 290.40 321.51
311.14 321.51 311.14 321.51 321.51 311.14 321.51 311.14 321.51 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0
6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0
6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0
372.0 372.0 360.0 372.0 360.0 372.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 310.0 280.0 310.0 300.0
310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 3 0 2 0 7.3 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0
31.0 30.0 31.0
 1 1060 A1060 1 344 200.00 0.00 0 0 0 0 3 0 0 3 1 -1 -1 -1 2250.60 2032.80 2250.60 2178.00 2250.60
2178.00 2250.60 2250.60 2178.00 2250.60 2178.00 2250.60 2250.60 2032.80 2250.60 2178.00 2250.60
2178.00 2250.60 2250.60 2178.00 2250.60 2178.00 2250.60 1574.83 1422.43 1574.83 1524.03 1574.83
1524.03 1574.83 1574.83 1524.03 1574.83 1524.03 1574.83 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 385.82 348.48 385.82 373.37 385.82 373.37 385.82 385.82
373.37 385.82 373.37 385.82 385.82 348.48 385.82 373.37 385.82 373.37 385.82 385.82 373.37 385.82
```

```
373.37 385.82 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0
8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0
372.0 360.0 372.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 310.0 280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 1 1 1 0
12.0 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1063 A1063 1 344 300.00 0.00 1 1 0 0 3 1 0 3 1 -1 -1 -1 1785.60 1612.80 1785.60 1728.00 1785.60
1728.00 1785.60 1785.60 1728.00 1785.60 1728.00 1785.60 2678.40 2419.20 2678.40 2592.00 2678.40
2592.00 2678.40 2678.40 2592.00 2678.40 2592.00 2678.40 2536.62 2291.14 2536.62 2454.79 2536.62
2454.79 2536.62 2536.62 2454.79 2536.62 2454.79 2536.62 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 248.00 224.00 248.00 240.00 248.00 240.00 248.00 248.00
240.00 248.00 240.00 248.00 248.00 224.00 248.00 240.00 248.00 240.00 248.00 248.00 240.00 248.00
240.00 248.00 6.20 5.60 6.20 6.00 6.20 6.00 6.20 6.20 6.00 6.20 6.00 6.20 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0
8.0 8.0 8.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0
496.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0 496.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 496.0 448.0 496.0 480.0 496.0 480.0 496.0 496.0 480.0 496.0 480.0 496.0 3 2 2 1 8.6 -1 31.0 28.0
31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1064 A1064 1 344 100.00 0.00 0 1 0 0 3 0 0 3 2 -1 -1 -1 1779.05 1601.60 1779.05 1815.45 1820.00
1733.55 1779.05 1779.05 1774.50 1738.10 1692.60 1901.90 1779.05 1601.60 1779.05 1815.45 1820.00
1733.55 1779.05 1779.05 1774.50 1738.10 1692.60 1901.90 1870.62 1689.59 1870.62 1810.28 1870.62
1810.28 1870.62 1870.62 1810.28 1870.62 1810.28 1870.62 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 254.15 228.80 254.15 259.35 260.00 247.65 254.15 254.15
253.50 248.30 241.80 271.70 776.75 702.00 776.75 744.25 773.50 750.75 776.75 776.75 747.50 780.00
754.00 767.00 2346.00 2112.00 2346.00 2394.00 2400.00 2286.00 2346.00 2346.00 2340.00 2292.00 2232.00
2508.00 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 373.0
336.0 373.0 377.0 380.0 363.0 373.0 373.0 370.0 366.0 356.0 394.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 373.0 336.0 373.0 377.0 380.0 363.0 373.0
373.0 370.0 366.0 356.0 394.0 1 2 1 1 1 1 1 1 1 2 2 1 1 2 1 1 1 1 1 1 1 2 2 1 239.0 216.0 239.0 229.0 238.0
231.0 239.0 239.0 230.0 240.0 232.0 236.0 1 0 1 0 27.2 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0
30.0 31.0
 1 1066 A1066 1 344 400.00 0.00 1 0 0 0 3 0 0 3 2 -1 -1 -1 2153.10 1944.73 2153.10 2083.64 2153.10
2083.64 2153.10 2153.10 2083.64 2153.10 2083.64 2153.10 1474.70 1331.99 1474.70 1427.13 1474.70
1427.13 1474.70 1474.70 1427.13 1474.70 1427.13 1474.70 3968.69 3584.62 3968.69 3840.66 3968.69
3840.66 3968.69 3968.69 3840.66 3968.69 3840.66 3968.69 29.99 27.09 29.99 29.02 29.99 29.02 29.99 29.99
29.02 29.99 29.02 29.99 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 294.94 266.40 294.94 285.43 294.94 285.43
294.94 294.94 285.43 294.94 285.43 294.94 294.94 266.40 294.94 285.43 294.94 285.43 294.94 294.94
285.43 294.94 285.43 294.94 6.77 6.11 6.77 6.55 6.77 6.55 6.77 6.77 6.55 6.77 6.55 6.77 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0
8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0
8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0
6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0
372.0 372.0 360.0 372.0 360.0 372.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 310.0 280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0
300.0 310.0 1 1 0 1 18.9 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
```

```
 1 1067 A1067 1 344 100.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 2250.60 2032.80 2250.60 2178.00 2250.60
2178.00 2250.60 2250.60 2178.00 2250.60 2178.00 2250.60 2250.60 2032.80 2250.60 2178.00 2250.60
2178.00 2250.60 2250.60 2178.00 2250.60 2178.00 2250.60 688.17 621.57 688.17 665.97 688.17 665.97
688.17 688.17 665.97 688.17 665.97 688.17 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 321.51 290.40 321.51 311.14 321.51 311.14 321.51 321.51 311.14 321.51
311.14 321.51 321.51 290.40 321.51 311.14 321.51 311.14 321.51 321.51 311.14 321.51 311.14 321.51 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0
6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0
336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 310.0 280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 3 1 0 0 9.3 0 31.0 28.0 31.0
30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1068 A1068 1 344 500.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 1020.60 925.34 1020.60 925.34 993.38 979.78
1020.60 1020.60 952.56 1047.82 1006.99 938.95 1895.40 1718.50 1895.40 1718.50 1844.86 1819.58 1895.40
1895.40 1769.04 1945.94 1870.13 1743.77 7114.91 6426.37 7114.91 6885.40 7114.91 6885.40 7114.91
7114.91 6885.40 7114.91 6885.40 7114.91 3348.19 3024.18 3348.19 3240.19 3348.19 3240.19 3348.19
3348.19 3240.19 3348.19 3240.19 3348.19 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50
14.50 14.50 14.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 405.00 367.20 405.00 367.20
394.20 388.80 405.00 405.00 378.00 415.80 399.60 372.60 610.20 550.80 610.20 596.70 612.90 591.30
610.20 610.20 594.00 607.50 588.60 618.30 8.25 7.48 8.25 7.48 8.03 7.92 8.25 8.25 7.70 8.47 8.14 7.59 15.0
15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 288.0 260.0 288.0 281.0
289.0 279.0 288.0 288.0 280.0 287.0 278.0 291.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 15.0 15.0 15.0
15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 288.0 260.0 288.0 281.0 289.0 279.0 288.0 288.0 280.0 287.0
278.0 291.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 226.0 204.0 226.0 221.0 227.0 219.0 226.0 226.0
220.0 225.0 218.0 229.0 1 1 0 0 32.8 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1069 A1069 1 344 150.00 10.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 1862.88 1685.46 1862.88 1751.99 1840.70
1796.35 1862.88 1862.88 1774.17 1885.06 1818.52 1796.35 1191.02 1077.59 1191.02 1120.13 1176.84
1148.48 1191.02 1191.02 1134.31 1205.20 1162.66 1148.48 5232.92 4726.51 5232.92 5064.12 5232.92
5064.12 5232.92 5232.92 5064.12 5232.92 5064.12 5232.92 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 277.63 251.19 277.63 261.10 274.32 267.71 277.63 277.63
264.41 280.93 271.02 267.71 528.81 475.93 528.81 542.03 542.03 515.59 528.81 528.81 528.81 515.59
502.37 568.47 168.00 152.00 168.00 158.00 166.00 162.00 168.00 168.00 160.00 170.00 164.00 162.00 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 9.0 9.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 7.0 11.0 11.0
11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 222.0 200.0 222.0 224.0 226.0 216.0 222.0 222.0 220.0 218.0 212.0 234.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 222.0 200.0 222.0 224.0 226.0
216.0 222.0 222.0 220.0 218.0 212.0 234.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 160.0 144.0 160.0
164.0 164.0 156.0 160.0 160.0 160.0 156.0 152.0 172.0 1 1 0 0 41.3 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0
30.0 31.0 30.0 31.0
 1 1070 A1070 1 344 200.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 22752.22 20550.39 22752.22 22018.28
22752.22 22018.28 22752.22 22752.22 22018.28 22752.22 22018.28 22752.22 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
```

```
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 2 0 2 0 55.4 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1071 A1071 1 344 600.00 10.00 0 1 0 1 2 0 0 2 2 -1 -1 -1 2028.95 1832.60 2028.95 1963.50 2028.95
1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1832.60 2028.95 1963.50 2028.95
1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 1631.32 1474.73 1631.32 1556.02 1621.42
1575.82 1631.32 1631.32 1565.92 1641.22 1585.73 1601.62 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 289.85 261.80 289.85 280.50 289.85 280.50 289.85 289.85
280.50 289.85 280.50 289.85 326.08 294.52 326.08 315.56 326.08 315.56 326.08 326.08 315.56 326.08
315.56 326.08 2.48 2.24 2.48 2.40 2.48 2.40 2.48 2.48 2.40 2.48 2.40 2.48 13.0 13.0 13.0 13.0 13.0 13.0 13.0
13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0
341.0 330.0 341.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 1 1 2 1
13.4 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1072 A1072 1 344 600.00 10.00 0 1 0 1 2 0 0 2 2 -1 -1 -1 2028.95 1832.60 2028.95 1963.50 2028.95
1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1832.60 2028.95 1963.50 2028.95
1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 18600.00 16800.00 18600.00 18000.00 18600.00
18000.00 18600.00 18600.00 18000.00 18600.00 18000.00 18600.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 289.85 261.80 289.85 280.50 289.85 280.50 289.85
289.85 280.50 289.85 280.50 289.85 326.08 294.52 326.08 315.56 326.08 315.56 326.08 326.08 315.56
326.08 315.56 326.08 2.48 2.24 2.48 2.40 2.48 2.40 2.48 2.48 2.40 2.48 2.40 2.48 13.0 13.0 13.0 13.0 13.0
13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0
341.0 330.0 341.0 330.0 341.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 1
1 2 1 33.2 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1073 A1073 1 344 600.00 10.00 0 1 0 1 2 0 0 2 2 -1 -1 -1 2028.95 1832.60 2028.95 1963.50 2028.95
1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1832.60 2028.95 1963.50 2028.95
1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 16325.77 14758.64 16325.77 15572.21 16226.69
15770.37 16325.77 16325.77 15671.29 16424.85 15869.45 16028.53 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 289.85 261.80 289.85 280.50 289.85 280.50 289.85
289.85 280.50 289.85 280.50 289.85 326.08 294.52 326.08 315.56 326.08 315.56 326.08 326.08 315.56
326.08 315.56 326.08 2.48 2.24 2.48 2.40 2.48 2.40 2.48 2.48 2.40 2.48 2.40 2.48 13.0 13.0 13.0 13.0 13.0
13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0
341.0 330.0 341.0 330.0 341.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 1
1 2 1 60.2 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1074 A1074 1 344 50.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 2250.60 2032.80 2250.60 2178.00 2250.60 2178.00
2250.60 2250.60 2178.00 2250.60 2178.00 2250.60 2250.60 2032.80 2250.60 2178.00 2250.60 2178.00
2250.60 2250.60 2178.00 2250.60 2178.00 2250.60 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00
```

16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 321.51 290.40 321.51 311.14 321.51 311.14 321.51 321.51 311.14 321.51 311.14 321.51 321.51 290.40 321.51 311.14 321.51 311.14 321.51 321.51 311.14 321.51 311.14 321.51 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 310.0 280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 3 0 2 0 7.3 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1077 A1077 1 344 200.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 2250.60 2032.80 2250.60 2178.00 2250.60 2178.00 2250.60 2250.60 2178.00 2250.60 2178.00 2250.60 2250.60 2032.80 2250.60 2178.00 2250.60 2178.00 2250.60 2250.60 2178.00 2250.60 2178.00 2250.60 1503.15 1359.71 1503.15 1418.75 1487.47 1450.11 1503.15 1503.15 1434.43 1518.83 1465.79 1456.11 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 385.82 348.48 385.82 373.37 385.82 373.37 385.82 385.82 373.37 385.82 373.37 385.82 385.82 348.48 385.82 373.37 385.82 373.37 385.82 385.82 373.37 385.82 373.37 385.82 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 310.0 280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 1 1 1 0 11.4 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1078 A1078 1 344 400.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 2153.10 1944.73 2153.10 2083.64 2153.10 2083.64 2153.10 2153.10 2083.64 2153.10 2083.64 2153.10 1474.70 1331.99 1474.70 1427.13 1474.70 1427.13 1474.70 1474.70 1427.13 1474.70 1427.13 1474.70 4020.94 3631.82 4020.94 3891.23 4020.94 3891.23 4020.94 4020.94 3891.23 4020.94 3891.23 4020.94 30.38 27.44 30.38 29.40 30.38 29.40 30.38 30.38 29.40 30.38 29.40 30.38 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 294.94 266.40 294.94 285.43 294.94 285.43 294.94 294.94 285.43 294.94 285.43 294.94 294.94 266.40 294.94 285.43 294.94 285.43 294.94 294.94 285.43 294.94 285.43 294.94 6.94 6.27 6.94 6.72 6.94 6.72 6.94 6.94 6.72 6.94 6.72 6.94 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 310.0 280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 1 1 1 0 11.4 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1079 A1079 1 344 150.00 0.00 0 0 0 0 1 0 0 1 1 15 15 15 221.08 199.69 221.08 213.95 221.08 213.95 221.08 221.08 213.95 221.08 213.95 221.08 141.35 127.67 141.35 136.79 141.35 136.79 141.35 141.35 136.79 141.35 136.79 141.35 1176.53 1057.67 1176.53 1227.47 1215.34 1149.85 1176.53 1176.53 1188.66 1137.72 1111.04 1292.96 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 16.50 16.50 16.50 16.50 16.50 16.50 16.50 16.50 16.50 16.50 16.50 16.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 32.95 29.76 32.95 31.89 32.95 31.89 32.95 32.95 31.89 32.95 31.89 32.95 40.70 36.76 40.70 39.39 40.70 39.39 40.70 40.70 39.39 40.70 39.39 40.70 21.91 19.79 21.91 21.20 21.91 21.20 21.91 21.91 21.20 21.91 21.20 21.91 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 2 2 2

```
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 217.0 196.0 217.0 210.0 217.0 210.0 217.0 217.0 210.0 217.0 210.0
217.0 1 1 0 0 5.2 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1080 A1080 1 344 100.00 0.00 0 0 0 0 1 0 0 1 1 15 15 15 419.99 379.35 419.99 406.45 419.99 406.45
419.99 419.99 406.45 419.99 406.45 419.99 140.00 126.45 140.00 135.48 140.00 135.48 140.00 140.00
135.48 140.00 135.48 140.00 694.32 627.13 694.32 671.92 694.32 671.92 694.32 694.32 671.92 694.32
671.92 694.32 162.87 147.10 162.87 157.61 162.87 157.61 162.87 162.87 157.61 162.87 157.61 162.87 25.0
25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 62.22 56.20 62.22 60.21 62.22 60.21 62.22 62.22 60.21 62.22 60.21 62.22 78.22
70.65 78.22 75.70 78.22 75.70 78.22 78.22 75.70 78.22 75.70 78.22 16.43 14.84 16.43 15.90 16.43 15.90
16.43 16.43 15.90 16.43 15.90 16.43 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0
11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 403.0 364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0 403.0 390.0 403.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 403.0
364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0 403.0 390.0 403.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 1 1 0 1 4.4 0 31.0 28.0 31.0
30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1081 A1081 1 344 150.00 0.00 0 0 0 0 1 0 0 1 1 15 15 15 190.24 171.78 190.24 185.06 190.66 184.22
190.24 190.24 184.64 189.82 183.81 191.49 121.63 109.82 121.63 118.31 121.89 117.78 121.63 121.63
118.05 121.36 117.52 122.43 656.73 593.04 656.73 637.90 657.76 635.84 656.73 656.73 636.87 655.70
634.81 659.82 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0
25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00
15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
28.35 25.60 28.35 27.58 28.41 27.46 28.35 28.35 27.52 28.29 27.39 28.54 40.43 36.54 40.43 38.74 40.26
39.08 40.43 40.43 38.91 40.60 39.25 39.92 33.52 30.27 33.52 32.61 33.59 32.46 33.52 33.52 32.53 33.45
32.39 33.74 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 2.0 2.0 1.0 1.0 1.0 1.0 2.0 1.0 1.0 2.0
2.0 2.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0
21.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0
4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 345.0
312.0 345.0 327.0 342.0 333.0 345.0 345.0 330.0 348.0 336.0 336.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0
4.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 345.0 312.0 345.0 327.0 342.0 333.0 345.0
345.0 330.0 348.0 336.0 336.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 239.0 216.0 239.0 229.0 238.0
231.0 239.0 239.0 230.0 240.0 232.0 236.0 1 1 0 0 4.2 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0
30.0 31.0
 1 1082 A1082 1 344 100.00 0.00 0 0 0 0 1 0 0 1 1 15 15 15 285.29 257.60 285.29 277.52 285.91 276.27
285.29 285.29 276.89 284.67 275.64 287.16 285.29 257.60 285.29 277.52 285.91 276.27 285.29 285.29
276.89 284.67 275.64 287.16 367.20 331.83 367.20 352.47 365.94 354.99 367.20 367.20 353.73 368.46
356.25 363.43 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0
25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00
15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
48.91 44.16 48.91 47.57 49.01 47.36 48.91 48.91 47.47 48.80 47.25 49.23 69.74 63.03 69.74 66.82 69.45
67.41 69.74 69.74 67.12 70.03 67.70 68.87 154.71 139.69 154.71 150.49 155.05 149.82 154.71 154.71 150.15
154.37 149.48 155.72 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 2.0 2.0 1.0 1.0 1.0 1.0 2.0
1.0 1.0 2.0 2.0 2.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0
21.0 21.0 21.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0
21.0 21.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0
21.0 345.0 312.0 345.0 327.0 342.0 333.0 345.0 345.0 330.0 348.0 336.0 336.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0
4.0 4.0 4.0 4.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 345.0 312.0 345.0 327.0 342.0
333.0 345.0 345.0 330.0 348.0 336.0 336.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 239.0 216.0 239.0
229.0 238.0 231.0 239.0 239.0 230.0 240.0 232.0 236.0 1 1 0 0 3.3 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0
30.0 31.0 30.0 31.0
 1 1083 A1083 1 344 300.00 0.00 0 0 0 0 1 0 0 1 1 15 15 15 110.97 100.88 110.97 95.84 105.93 105.93
110.97 110.97 100.88 116.02 110.97 95.84 206.09 187.35 206.09 177.99 196.72 196.72 206.09 206.09 187.35
215.46 206.09 177.99 2619.53 2375.46 2619.53 2367.66 2546.45 2513.81 2619.53 2619.53 2440.74 2692.60
2586.89 2400.30 1232.72 1117.86 1232.72 1114.19 1198.33 1182.97 1232.72 1232.72 1148.58 1267.11
1217.36 1129.55 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 23.78 21.62 23.78 20.54 22.70 22.70 23.78 23.78 21.62
24.86 23.78 20.54 43.80 39.82 43.80 37.83 41.81 41.81 43.80 43.80 39.82 45.80 43.80 37.83 23.32 21.20
```

23.32 20.14 22.26 22.26 23.32 23.32 21.20 24.38 23.32 20.14 48.0 48.0 48.0 48.0 48.0 48.0 48.0 48.0 48.0 48.0 48.0 48.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 374.0 340.0 374.0 323.0 357.0 357.0 374.0 374.0 340.0 391.0 374.0 323.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 374.0 340.0 374.0 323.0 357.0 357.0 374.0 374.0 340.0 391.0 374.0 323.0 1 1 1 2 2 2 1 1 2 1 1 2 1 1 1 2 2 2 1 1 2 1 1 2 154.0 140.0 154.0 133.0 147.0 147.0 154.0 154.0 140.0 161.0 154.0 133.0 1 1 0 0 21.9 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1084 A1084 1 344 100.00 0.00 0 0 0 1 1 0 0 1 1 15 15 15 321.35 288.00 321.35 350.93 338.79 316.05 321.35 321.35 333.49 303.91 298.60 373.67 321.35 288.00 321.35 350.93 338.79 316.05 321.35 321.35 333.49 303.91 298.60 373.67 500.25 449.98 500.25 517.11 514.66 488.30 500.25 500.25 502.70 485.85 473.89 543.47 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 35.71 32.00 35.71 38.99 37.64 35.12 35.71 35.71 37.05 33.77 33.18 41.52 47.60 42.79 47.60 49.61 49.15 46.51 47.60 47.60 48.06 46.05 44.96 52.25 160.60 143.94 160.60 175.39 169.32 157.95 160.60 160.60 166.67 151.89 149.24 186.76 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 545.0 492.0 545.0 532.0 547.0 528.0 545.0 545.0 530.0 543.0 526.0 551.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 545.0 492.0 545.0 532.0 547.0 528.0 545.0 545.0 530.0 543.0 526.0 551.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 307.0 276.0 307.0 320.0 317.0 300.0 307.0 307.0 310.0 297.0 290.0 337.0 1 1 0 0 3.0 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1085 A1085 1 344 150.00 0.00 0 0 0 0 1 0 0 1 1 15 15 15 93.98 85.44 93.98 81.17 89.71 89.71 93.98 93.98 85.44 98.26 93.98 81.17 93.98 85.44 93.98 81.17 89.71 89.71 93.98 93.98 85.44 98.26 93.98 81.17 257.56 233.70 257.56 230.19 249.24 246.83 257.56 257.56 238.51 265.88 255.15 232.60 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 18.80 17.09 18.80 16.23 17.94 17.94 18.80 18.80 17.09 19.65 18.80 16.23 34.63 31.48 34.63 29.90 33.05 33.05 34.63 34.63 31.48 36.20 34.63 29.90 23.32 21.20 23.32 20.14 22.26 22.26 23.32 23.32 21.20 24.38 23.32 20.14 48.0 48.0 48.0 48.0 48.0 48.0 48.0 48.0 48.0 48.0 48.0 48.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 374.0 340.0 374.0 323.0 357.0 357.0 374.0 374.0 340.0 391.0 374.0 323.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 374.0 340.0 374.0 323.0 357.0 357.0 374.0 374.0 340.0 391.0 374.0 323.0 1 1 1 2 2 2 1 1 2 1 1 2 1 1 1 2 2 2 1 1 2 1 1 2 154.0 140.0 154.0 133.0 147.0 147.0 154.0 154.0 140.0 161.0 154.0 133.0 1 1 0 0 2.8 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1086 A1086 1 344 100.00 0.00 0 1 0 1 1 0 0 1 1 15 15 15 443.80 399.60 443.80 451.76 453.53 432.31 443.80 443.80 442.04 434.08 422.59 472.98 443.80 399.60 443.80 451.76 453.53 432.31 443.80 443.80 442.04 434.08 422.59 472.98 1115.32 1007.39 1115.32 1079.34 1115.32 1079.34 1115.32 1115.32 1079.34 1115.32 1079.34 1115.32 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 49.31 44.40 49.31 50.20 50.39 48.03 49.31 49.31 49.12 48.23 46.95 52.55 64.44 58.15 64.44 63.26 64.83 62.47 64.44 64.44 62.87 64.05 62.08 65.62 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 5.0 5.0 3.0 3.0 3.0 3.0 5.0 3.0 3.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 328.0 296.0 328.0 322.0 330.0 318.0 328.0 328.0 320.0 326.0 316.0 334.0 1 1 0 0 3.9 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1087 A1087 1 344 150.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 2864.40 2587.20 2864.40 2772.00 2864.40 2772.00 2864.40 2864.40 2772.00 2864.40 2772.00 2864.40 2864.40 2587.20 2864.40 2772.00 2864.40 2772.00 2864.40 2864.40 2772.00 2864.40 2772.00 2864.40 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00

0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 10.50 10.50 10.50 10.50 10.50 10.50 10.50 10.50 10.50 10.50 10.50 10.50 10.50 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 409.20 369.60 409.20 396.00 409.20 396.00 409.20 409.20 396.00 409.20 396.00 409.20 409.20 369.60 409.20 396.00 409.20 396.00 409.20 409.20 396.00 409.20 396.00 409.20 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 465.0 420.0 465.0 450.0 465.0 450.0 465.0 450.0 465.0 465.0 450.0 465.0 450.0 465.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 465.0 420.0 465.0 450.0 465.0 450.0 465.0 465.0 450.0 465.0 450.0 465.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 403.0 364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0 403.0 390.0 403.0 3 0 2 0 7.1 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1088 A1088 1 344 100.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 3274.74 2956.80 3274.74 3187.30 3282.68 3171.41 3274.74 3274.74 3179.35 3266.79 3163.46 3298.58 3274.74 2956.80 3274.74 3187.30 3282.68 3171.41 3274.74 3274.74 3179.35 3266.79 3163.46 3298.58 1488.00 1344.00 1488.00 1440.00 1488.00 1440.00 1488.00 1488.00 1440.00 1488.00 1440.00 1488.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 467.82 422.40 467.82 455.33 468.95 453.06 467.82 467.82 454.19 466.68 451.92 471.23 467.82 422.40 467.82 455.33 468.95 453.06 467.82 467.82 454.19 466.68 451.92 471.23 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 412.0 372.0 412.0 401.0 413.0 399.0 412.0 412.0 400.0 411.0 398.0 415.0 3 1 0 0 9.9 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1089 A1089 1 344 200.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 1812.58 1647.80 1812.58 1565.41 1730.19 1730.19 1812.58 1812.58 1647.80 1894.97 1812.58 1565.41 1812.58 1647.80 1812.58 1565.41 1730.19 1730.19 1812.58 1812.58 1647.80 1894.97 1812.58 1565.41 1438.57 1306.70 1438.57 1261.74 1381.62 1375.63 1438.57 1438.57 1318.69 1495.51 1432.57 1267.74 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 310.73 282.48 310.73 268.36 296.60 296.60 310.73 310.73 282.48 324.85 310.73 268.36 310.73 282.48 310.73 268.36 296.60 296.60 310.73 310.73 282.48 324.85 310.73 268.36 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 60.0 60.0 60.0 60.0 60.0 60.0 60.0 60.0 60.0 60.0 60.0 60.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 12.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 1 1 1 0 12.9 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1090 A1090 1 344 500.00 9.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 4229.81 3820.47 4229.81 4093.36 4229.81 4093.36 4229.81 4229.81 4093.36 4229.81 4093.36 4229.81 1694.29 1530.33 1694.29 1639.64 1694.29 1639.64 1694.29 1694.29 1639.64 1694.29 1639.64 1694.29 6485.26 5857.65 6485.26 6276.05 6485.26 6276.05 6485.26 6485.26 6276.05 6485.26 6276.05 6485.26 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 15.50 15.50 15.50 15.50 15.50 15.50 15.50 15.50 15.50 15.50 15.50 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 423.15 382.20 423.15 409.50 423.15 409.50 423.15 423.15 409.50 423.15 409.50 423.15 564.20 509.60 564.20 546.00 564.20 546.00 564.20 564.20 546.00 564.20 546.00 564.20 0.93 0.84 0.93 0.90 0.93 0.90 0.93 0.93 0.90 0.93 0.90 0.93 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0

```
18.0 18.0 18.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 1 0 0
30.6 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1093 A1093 1 344 1000.00 0.00 0 0 0 0 2 0 0 2 3 -1 -1 -1 2004.93 1822.66 2004.93 1731.53 1913.79
1913.79 2004.93 2004.93 1822.66 2096.06 2004.93 1731.53 1614.07 1467.34 1614.07 1393.97 1540.71
1540.71 1614.07 1614.07 1467.34 1687.44 1614.07 1393.97 24735.00 22470.00 24735.00 21652.50 23737.50
23647.50 24735.00 24735.00 22650.00 25732.50 24645.00 21742.50 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 10.0
10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 2479.40 2254.00 2479.40 2141.30 2366.70 2366.70
2479.40 2479.40 2254.00 2592.10 2479.40 2141.30 2479.40 2254.00 2479.40 2141.30 2366.70 2366.70
2479.40 2479.40 2254.00 2592.10 2479.40 2141.30 25.77 23.43 25.77 22.26 24.60 24.60 25.77 25.77 23.43
26.94 25.77 22.26 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0
744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0
720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 308.0 280.0 308.0 266.0 294.0 294.0
308.0 308.0 280.0 322.0 308.0 266.0 1 0 0 1 81.5 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0
31.0
 1 1094 A1094 1 344 125.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 8947.67 8081.77 8947.67 8659.04 8947.67
8659.04 8947.67 8947.67 8659.04 8947.67 8659.04 8947.67 3584.08 3237.23 3584.08 3468.46 3584.08
3468.46 3584.08 3584.08 3468.46 3584.08 3468.46 3584.08 8990.00 8120.00 8990.00 8700.00 8990.00
8700.00 8990.00 8990.00 8700.00 8990.00 8700.00 8990.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 10.0 10.0 10.0
10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 895.12 808.50 895.12 866.25 895.12 866.25 895.12 895.12
866.25 895.12 866.25 895.12 1302.00 1176.00 1302.00 1260.00 1302.00 1260.00 1302.00 1302.00 1260.00
1302.00 1260.00 1302.00 186.00 168.00 186.00 180.00 186.00 180.00 186.00 186.00 180.00 186.00 180.00
186.00 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 6.0 6.0 6.0
6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 6.0 6.0 6.0 6.0
6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 4.0 4.0 4.0 4.0 4.0
4.0 4.0 4.0 4.0 4.0 4.0 4.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 527.0 476.0 527.0 510.0
527.0 510.0 527.0 527.0 510.0 527.0 510.0 527.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 527.0 476.0 527.0 510.0 527.0 510.0 527.0 527.0 510.0 527.0
510.0 527.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0
360.0 372.0 360.0 372.0 1 1 0 0 55.0 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1095 A1095 1 344 300.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 4991.10 4508.09 4991.10 4830.10 4991.10
4830.10 4991.10 4991.10 4830.10 4991.10 4830.10 4991.10 1996.30 1803.11 1996.30 1931.90 1996.30
1931.90 1996.30 1996.30 1931.90 1996.30 1931.90 1996.30 33623.04 30369.20 33623.04 32538.42 33623.04
32538.42 33623.04 33623.04 32538.42 33623.04 32538.42 33623.04 15822.61 14291.39 15822.61 15312.20
15822.61 15312.20 15822.61 15822.61 15312.20 15822.61 15312.20 15822.61 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 14.00 14.00 14.00
14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0
10.0 1247.75 1127.00 1247.75 1207.50 1247.75 1207.50 1247.75 1247.75 1207.50 1247.75 1207.50 1247.75
1247.75 1127.00 1247.75 1207.50 1247.75 1207.50 1247.75 1247.75 1207.50 1247.75 1207.50 1247.75 10.23
9.24 10.23 9.90 10.23 9.90 10.23 10.23 9.90 10.23 9.90 10.23 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0
360.0 372.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 310.0 280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 1 1 0 0 96.4 0
31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1096 A1096 1 344 150.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 37200.00 33600.00 37200.00 36000.00
37200.00 36000.00 37200.00 37200.00 36000.00 37200.00 36000.00 37200.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 5.0 5.0 5.0 5.0 5.0
5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.00 5.00 5.00 5.00 5.00 5.00 5.00 5.00 5.00 5.00 5.00 5.00 5.0 5.0 5.0 5.0 5.0 5.0
```

```
5.0 5.0 5.0 5.0 5.0 5.0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 0 2 0 59.1 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0
 1 1097 A1097 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 4477.44 4070.40 4477.44 3866.88 4273.92
4273.92 4477.44 4477.44 4070.40 4680.96 4477.44 3866.88 4850.56 4409.60 4850.56 4189.12 4630.08
4630.08 4850.56 4850.56 4409.60 5071.04 4850.56 4189.12 5110.00 4640.00 5110.00 4510.00 4920.00
4890.00 5110.00 5110.00 4700.00 5300.00 5080.00 4540.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 10.0 10.0 10.0
10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 1408.00 1280.00 1408.00 1216.00 1344.00 1344.00 1408.00
1408.00 1280.00 1472.00 1408.00 1216.00 1584.00 1440.00 1584.00 1368.00 1512.00 1512.00 1584.00
1584.00 1440.00 1656.00 1584.00 1368.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 61.0
61.0 61.0 61.0 61.0 61.0 61.0 61.0 61.0 61.0 61.0 61.0 6.0 6.0 7.0 6.0 6.0 7.0 6.0 6.0 13.0 6.0 6.0 6.0 9.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 242.0 220.0 242.0
209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 242.0 220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0
253.0 242.0 209.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 198.0 180.0 198.0 171.0 189.0 189.0 198.0
198.0 180.0 207.0 198.0 171.0 1 1 0 1 50.4 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1098 A1098 1 344 125.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 2838.46 2563.77 2838.46 2746.90 2838.46
2746.90 2838.46 2838.46 2746.90 2838.46 2746.90 2838.46 1136.98 1026.95 1136.98 1100.30 1136.98
1100.30 1136.98 1136.98 1100.30 1136.98 1100.30 1136.98 694.32 627.13 694.32 671.92 694.32 671.92
694.32 694.32 671.92 694.32 671.92 694.32 162.87 147.10 162.87 157.61 162.87 157.61 162.87 162.87
157.61 162.87 157.61 162.87 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00
14.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 283.96 256.48 283.96 274.80 283.96 274.80
283.96 283.96 274.80 283.96 274.80 283.96 356.98 322.43 356.98 345.46 356.98 345.46 356.98 356.98
345.46 356.98 345.46 356.98 248.00 224.00 248.00 240.00 248.00 240.00 248.00 248.00 240.00 248.00
240.00 248.00 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0
11.0 11.0 11.0 11.0 11.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 403.0 364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0 403.0 390.0 403.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 403.0 364.0 403.0 390.0 403.0
390.0 403.0 403.0 390.0 403.0 390.0 403.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 341.0 308.0 341.0
330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 1 1 0 1 13.4 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0
30.0 31.0 30.0 31.0
 1 1099 A1099 1 344 400.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 3272.29 2955.62 3272.29 3166.73 3272.29
3166.73 3272.29 3272.29 3166.73 3272.29 3166.73 3272.29 1310.75 1183.90 1310.75 1268.47 1310.75
1268.47 1310.75 1310.75 1268.47 1310.75 1268.47 1310.75 18555.04 16759.39 18555.04 17956.49 18555.04
17956.49 18555.04 18555.04 17956.49 18555.04 17956.49 18555.04 976.58 882.07 976.58 945.08 976.58
945.08 976.58 976.58 945.08 976.58 945.08 976.58 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0
27.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 14.00 14.00 14.00 14.00 14.00 14.00 14.00
14.00 14.00 14.00 14.00 14.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 392.83 354.82
392.83 380.16 392.83 380.16 392.83 392.83 380.16 392.83 380.16 392.83 523.78 473.09 523.78 506.88
523.78 506.88 523.78 523.78 506.88 523.78 506.88 523.78 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0
5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0
5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 434.0
392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0
5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0
434.0 420.0 434.0 420.0 434.0 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 372.0 336.0 372.0 360.0 372.0
360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 1 0 0 55.8 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
```

```
 1 1100 A1100 1 344 500.00 0.00 0 1 0 0 2 0 1 2 2 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 7440.00 6720.00 7440.00 7200.00
7440.00 7200.00 7440.00 7440.00 7200.00 7440.00 7200.00 7440.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0
744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0
720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 744.0 672.0 744.0 720.0 744.0 720.0
744.0 744.0 720.0 744.0 720.0 744.0 1 1 0 1 10.0 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0
31.0
 1 1101 A1101 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 15847.92 14407.20 15847.92 13686.84 15127.56
15127.56 15847.92 15847.92 14407.20 16568.28 15847.92 13686.84 4392.08 3992.80 4392.08 3793.16
4192.44 4192.44 4392.08 4392.08 3992.80 4591.72 4392.08 3793.16 1022.00 928.00 1022.00 902.00 984.00
978.00 1022.00 1022.00 940.00 1060.00 1016.00 908.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 10.0 10.0 10.0
10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 1760.00 1600.00 1760.00 1520.00 1680.00 1680.00 1760.00
1760.00 1600.00 1840.00 1760.00 1520.00 1760.00 1600.00 1760.00 1520.00 1680.00 1680.00 1760.00
1760.00 1600.00 1840.00 1760.00 1520.00 6.60 6.00 6.60 5.70 6.30 6.30 6.60 6.60 6.00 6.90 6.60 5.70 14.0
14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0
210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0
220.0 190.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 176.0 160.0 176.0 152.0 168.0 168.0 176.0 176.0
160.0 184.0 176.0 152.0 1 1 2 1 95.0 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1102 A1102 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 2471.92 2247.20 2471.92 2134.84 2359.56
2359.56 2471.92 2471.92 2247.20 2584.28 2471.92 2134.84 2192.08 1992.80 2192.08 1893.16 2092.44
2092.44 2192.08 2192.08 1992.80 2291.72 2192.08 1893.16 1488.00 1344.00 1488.00 1440.00 1488.00
1440.00 1488.00 1488.00 1440.00 1488.00 1440.00 1488.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 26.0 26.0 26.0 26.0 26.0 26.0 26.0 26.0 26.0 26.0 26.0 26.0 26.0 26.0 26.0 26.0 26.0 26.0 26.0 26.0
26.0 26.0 26.0 26.0 19.00 19.00 19.00 19.00 19.00 19.00 19.00 19.00 19.00 19.00 19.00 19.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 211.20 192.00 211.20 182.40 201.60 201.60 211.20 211.20
192.00 220.80 211.20 182.40 211.20 192.00 211.20 182.40 201.60 201.60 211.20 211.20 192.00 220.80
211.20 182.40 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 62.0 62.0 62.0 62.0 62.0 62.0 62.0
62.0 62.0 62.0 62.0 62.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 14.0 7.0 7.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0
200.0 230.0 220.0 190.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0
16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 176.0 160.0 176.0 152.0 168.0 168.0 176.0 176.0 160.0 184.0 176.0 152.0 1 1 0
0 16.0 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1103 A1103 1 344 300.00 0.00 1 1 0 0 3 1 0 3 1 -1 -1 -1 7337.36 6630.80 7337.36 7038.43 7310.19
7092.78 7337.36 7337.36 7065.61 7364.54 7119.96 7255.83 7337.36 6630.80 7337.36 7038.43 7310.19
7092.78 7337.36 7337.36 7065.61 7364.54 7119.96 7255.83 26145.90 23657.74 26145.90 24555.38 25819.70
25207.78 26145.90 26145.90 24881.58 26472.10 25533.98 25167.30 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 15.0 15.0 15.0 15.0 15.0
15.0 15.0 15.0 15.0 15.0 15.0 12.50 12.50 12.50 12.50 12.50 12.50 12.50 12.50 12.50 12.50 12.50 12.50 10.0
10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 3635.33 3285.26 3635.33 3487.22 3621.86 3514.15
3635.33 3635.33 3500.69 3648.79 3527.62 3594.94 4645.14 4200.82 4645.14 4402.79 4604.75 4483.57
4645.14 4645.14 4443.18 4685.53 4523.96 4523.96 1.50 1.36 1.50 1.44 1.49 1.45 1.50 1.50 1.44 1.51 1.46
1.48 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 6.0 5.0 5.0 5.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0
```

7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 407.0 368.0 407.0 387.0 404.0 393.0 407.0 407.0 390.0 410.0 396.0 398.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 407.0 368.0 407.0 387.0 404.0 393.0 407.0 407.0 390.0 410.0 396.0 398.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 345.0 312.0 345.0 327.0 342.0 333.0 345.0 345.0 330.0 348.0 336.0 336.0 3 2 2 1 96.2 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1104 A1104 1 344 400.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 1583.37 1439.42 1583.37 1367.45 1511.40 1511.40 1583.37 1583.37 1439.42 1655.34 1583.37 1367.45 634.23 576.58 634.23 547.75 605.40 605.40 634.23 634.23 576.58 663.06 634.23 547.75 6983.07 6342.96 6983.07 6124.74 6706.66 6677.56 6983.07 6983.07 6401.15 7259.49 6953.98 6153.83 215.97 196.17 215.97 189.42 207.42 206.52 215.97 215.97 197.97 224.52 215.07 190.32 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 1536.48 1396.80 1536.48 1326.96 1466.64 1466.64 1536.48 1536.48 1396.80 1606.32 1536.48 1326.96 2048.64 1862.40 2048.64 1769.28 1955.52 1955.52 2048.64 2048.64 1862.40 2141.76 2048.64 1769.28 7.26 6.60 7.26 6.27 6.93 6.93 7.26 7.26 6.60 7.59 7.26 6.27 58.0 58.0 58.0 58.0 58.0 58.0 58.0 58.0 58.0 58.0 58.0 58.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 10.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 1 1 0 1 30.5 -1 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1105 A1105 1 344 1000.00 0.00 0 0 0 0 2 0 0 2 3 -1 -1 -1 4235.00 3850.00 4235.00 3657.50 4042.50 4042.50 4235.00 4235.00 3850.00 4427.50 4235.00 3657.50 1925.00 1750.00 1925.00 1662.50 1837.50 1837.50 1925.00 1925.00 1750.00 2012.50 1925.00 1662.50 24735.00 22470.00 24735.00 21652.50 23737.50 23647.50 24735.00 24735.00 22650.00 25732.50 24645.00 21742.50 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 4966.50 4515.00 4966.50 4289.25 4740.75 4740.75 4966.50 4966.50 4515.00 5192.25 4966.50 4289.25 4966.50 4515.00 4966.50 4289.25 4740.75 4740.75 4966.50 4966.50 4515.00 5192.25 4966.50 4289.25 27.61 25.10 27.61 23.85 26.36 26.36 27.61 27.61 25.10 28.87 27.61 23.85 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 1 0 0 0 88.8 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1106 A1106 1 344 100.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 7724.75 7022.50 7724.75 6671.38 7373.62 7373.62 7724.75 7724.75 7022.50 8075.88 7724.75 6671.38 6850.25 6227.50 6850.25 5916.12 6538.88 6538.88 6850.25 6850.25 6227.50 7161.62 6850.25 5916.12 1658.50 1498.00 1658.50 1605.00 1658.50 1605.00 1658.50 1658.50 1605.00 1658.50 1605.00 1658.50 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 16.50 16.50 16.50 16.50 16.50 16.50 16.50 16.50 16.50 16.50 16.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 1100.00 1000.00 1100.00 950.00 1050.00 1050.00 1100.00 1100.00 1000.00 1150.00 1100.00 950.00 1100.00 1000.00 1100.00 950.00 1050.00 1050.00 1100.00 1100.00 1000.00 1150.00 1100.00 950.00 29.04 26.40 29.04 25.08 27.72 27.72 29.04 29.04 26.40 30.36 29.04 25.08 60.0 60.0 60.0 60.0 60.0 60.0 60.0 60.0 60.0 60.0 60.0 60.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 12.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 1 0 1 0 40.1 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1107 A1107 1 344 300.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 15847.92 14407.20 15847.92 13686.84 15127.56 15127.56 15847.92 15847.92 14407.20 16568.28 15847.92 13686.84 4392.08 3992.80 4392.08 3793.16 4192.44 4192.44 4392.08 4392.08 3992.80 4591.72 4392.08 3793.16 1022.00 928.00 1022.00 902.00 984.00 978.00 1022.00 1022.00 940.00 1060.00 1016.00 908.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 880.00 800.00 880.00 760.00 840.00 840.00 880.00 880.00 800.00 920.00 880.00 760.00 880.00 800.00 880.00 760.00 840.00 840.00 880.00 880.00 800.00 920.00 880.00 760.00 6.60 6.00 6.60 5.70 6.30 6.30 6.60 6.60 6.00 6.90 6.60 5.70 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 176.0 160.0 176.0 152.0 168.0 168.0 176.0 176.0 160.0 184.0 176.0 152.0 1 1 0 1 95.0 -1 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1108 A1108 1 344 1000.00 0.00 0 0 0 0 2 0 0 2 3 -1 -1 -1 3078.77 2798.88 3078.77 2658.94 2938.82 2938.82 3078.77 3078.77 2798.88 3218.71 3078.77 2658.94 1233.23 1121.12 1233.23 1065.06 1177.18 1177.18 1233.23 1233.23 1121.12 1289.29 1233.23 1065.06 49470.00 44940.00 49470.00 43305.00 47475.00 47295.00 49470.00 49470.00 45300.00 51465.00 49290.00 43485.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 2494.80 2268.00 2494.80 2154.60 2381.40 2381.40 2494.80 2494.80 2268.00 2608.20 2494.80 2154.60 2494.80 2268.00 2494.80 2154.60 2381.40 2381.40 2494.80 2494.80 2268.00 2608.20 2494.80 2154.60 12.74 11.59 12.74 11.01 12.17 12.17 12.74 12.74 11.59 13.32 12.74 11.01 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 1 0 0 1 160.0 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1109 A1109 1 344 125.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 3844.99 3472.90 3844.99 3720.96 3844.99 3720.96 3844.99 3844.99 3720.96 3844.99 3720.96 3844.99 1540.15 1391.10 1540.15 1490.47 1540.15 1490.47 1540.15 1540.15 1490.47 1540.15 1490.47 1540.15 1016.16 917.82 1016.16 983.38 1016.16 983.38 1016.16 1016.16 983.38 1016.16 983.38 1016.16 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 461.58 416.91 461.58 446.69 461.58 446.69 461.58 461.58 446.69 461.58 446.69 461.58 696.73 629.30 696.73 674.25 696.73 674.25 696.73 696.73 674.25 696.73 674.25 696.73 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 248.0 224.0 248.0 240.0 248.0 240.0 248.0 248.0 240.0 248.0 240.0 248.0 1 1 0 0 25.7 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1110 A1110 1 344 150.00 0.00 0 0 0 0 2 0 0 2 3 -1 -1 -1 6527.18 5895.52 6527.18 6316.63 6527.18 6316.63 6527.18 6527.18 6316.63 6527.18 6316.63 6527.18 3355.00 3030.32 3355.00 3246.77 3355.00 3246.77 3355.00 3355.00 3246.77 3355.00 3246.77 3355.00 6655.55 6011.46 6655.55 6440.85 6655.55 6440.85 6655.55 6655.55 6440.85 6655.55 6440.85 6655.55 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 1228.21 1109.35 1228.21 1188.59 1228.21 1188.59 1228.21 1228.21 1188.59 1228.21 1188.59 1228.21 1228.21 1109.35 1228.21 1188.59 1228.21 1188.59 1228.21 1228.21 1188.59 1228.21 1188.59 1228.21 58.20 52.56 58.20 56.32 58.20 56.32 58.20 58.20 56.32 58.20

```
56.32 58.20 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0
720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0
744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 403.0 364.0 403.0 390.0 403.0 390.0 403.0
403.0 390.0 403.0 390.0 403.0 1 1 0 1 29.0 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1111 A1111 1 344 350.00 0.00 1 0 0 0 2 0 0 2 3 -1 -1 -1 3657.83 3325.30 3657.83 3159.04 3491.57
3491.57 3657.83 3657.83 3325.30 3824.10 3657.83 3159.04 1747.57 1588.70 1747.57 1509.26 1668.13
1668.13 1747.57 1747.57 1588.70 1827.00 1747.57 1509.26 8421.72 7649.73 8421.72 7386.55 8088.36
8053.27 8421.72 8421.72 7719.91 8755.08 8386.63 7421.64 63.64 57.81 63.64 55.82 61.12 60.86 63.64 63.64
58.34 66.16 63.38 56.08 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 21.0 21.0 21.0 21.0 21.0
21.0 21.0 21.0 21.0 21.0 21.0 21.0 15.50 15.50 15.50 15.50 15.50 15.50 15.50 15.50 15.50 15.50 15.50 15.50
10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 386.10 351.00 386.10 333.45 368.55 368.55
386.10 386.10 351.00 403.65 386.10 333.45 514.80 468.00 514.80 444.60 491.40 491.40 514.80 514.80
468.00 538.20 514.80 444.60 4.12 3.75 4.12 3.56 3.93 3.93 4.12 4.12 3.75 4.31 4.12 3.56 10.0 10.0 10.0 10.0
10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0
5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0
308.0 308.0 280.0 322.0 308.0 266.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0
264.0 228.0 1 1 0 1 45.6 -1 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1112 A1112 1 344 100.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 2267.65 2048.20 2267.65 2194.50 2267.65
2194.50 2267.65 2267.65 2194.50 2267.65 2194.50 2267.65 2267.65 2048.20 2267.65 2194.50 2267.65
2194.50 2267.65 2267.65 2194.50 2267.65 2194.50 2267.65 1488.00 1344.00 1488.00 1440.00 1488.00
1440.00 1488.00 1488.00 1440.00 1488.00 1440.00 1488.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 323.95 292.60 323.95 313.50 323.95 313.50 323.95 323.95
313.50 323.95 313.50 323.95 431.93 390.13 431.93 418.00 431.93 418.00 431.93 431.93 418.00 431.93
418.00 431.93 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0
744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 3 1 0 0 10.1 0 31.0 28.0
31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1113 A1113 1 344 200.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 37200.00 33600.00 37200.00 36000.00
37200.00 36000.00 37200.00 37200.00 36000.00 37200.00 36000.00 37200.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 2 0 2 0 59.9 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1114 A1114 1 344 200.00 9.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 3203.35 2893.35 3203.35 3100.02 3203.35
3100.02 3203.35 3203.35 3100.02 3203.35 3100.02 3203.35 2048.05 1849.85 2048.05 1981.98 2048.05
1981.98 2048.05 2048.05 1981.98 2048.05 1981.98 2048.05 1807.40 1632.49 1807.40 1749.10 1807.40
1749.10 1807.40 1807.40 1749.10 1807.40 1749.10 1807.40 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
```

20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 375.10 338.80 375.10 363.00 375.10 363.00 375.10 375.10 363.00 375.10 363.00 375.10 375.10 338.80 375.10 363.00 375.10 363.00 375.10 375.10 363.00 375.10 363.00 375.10 1.86 1.68 1.86 1.80 1.86 1.80 1.86 1.86 1.80 1.86 1.80 1.86 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 310.0 280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 1 1 0 0 15.8 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1115 A1115 1 344 50.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 3341.80 3018.40 3341.80 3234.00 3341.80 3234.00 3341.80 3341.80 3234.00 3341.80 3234.00 3341.80 3341.80 3018.40 3341.80 3234.00 3341.80 3234.00 3341.80 3341.80 3234.00 3341.80 3234.00 3341.80 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 477.40 431.20 477.40 462.00 477.40 462.00 477.40 477.40 462.00 477.40 462.00 477.40 477.40 431.20 477.40 462.00 477.40 462.00 477.40 477.40 462.00 477.40 462.00 477.40 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 465.0 420.0 465.0 450.0 465.0 450.0 465.0 465.0 450.0 465.0 450.0 465.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 465.0 420.0 465.0 450.0 465.0 450.0 465.0 465.0 450.0 465.0 450.0 465.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 403.0 364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0 403.0 390.0 403.0 3 0 2 0 8.3 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1118 A1118 1 344 200.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 2321.90 2097.20 2321.90 2247.00 2321.90 2247.00 2321.90 2321.90 2247.00 2321.90 2247.00 2321.90 2321.90 2097.20 2321.90 2247.00 2321.90 2247.00 2321.90 2321.90 2247.00 2321.90 2247.00 2321.90 3720.00 3360.00 3720.00 3600.00 3720.00 3600.00 3720.00 3720.00 3600.00 3720.00 3600.00 3720.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 398.04 359.52 398.04 385.20 398.04 385.20 398.04 398.04 385.20 398.04 385.20 398.04 530.72 479.36 530.72 513.60 530.72 513.60 530.72 530.72 513.60 530.72 513.60 530.72 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 1 1 0 13.3 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1119 A1119 1 344 200.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 6353.76 5738.88 6353.76 6148.80 6353.76 6148.80 6353.76 6353.76 6148.80 6353.76 6148.80 6353.76 4062.24 3669.12 4062.24 3931.20 4062.24 3931.20 4062.24 4062.24 3931.20 4062.24 3931.20 4062.24 3720.00 3360.00 3720.00 3600.00 3720.00 3600.00 3720.00 3720.00 3600.00 3720.00 3600.00 3720.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 744.00 672.00 744.00 720.00 744.00 720.00 744.00 744.00 720.00 744.00 720.00 744.00 744.00 672.00 744.00 720.00 744.00 720.00 744.00 744.00 720.00 744.00 720.00 744.00 15.50 14.00 15.50 15.00 15.50 15.00 15.50 15.50 15.00 15.50 15.00 15.50 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 403.0 364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0 403.0 390.0 403.0 1 1 0

1 20.8 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
1 1120 A1120 1 344 200.00 0.00 0 1 0 0 2 0 0 2 1 -1 -1 -1 8004.64 7230.00 8004.64 7746.43 8004.64 7746.43 8004.64 8004.64 7746.43 8004.64 7746.43 8004.64 8004.64 7230.00 8004.64 7746.43 8004.64 7746.43 8004.64 8004.64 7746.43 8004.64 7746.43 8004.64 3720.00 3360.00 3720.00 3600.00 3720.00 3600.00 3720.00 3720.00 3600.00 3720.00 3600.00 3720.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 889.40 803.33 889.40 860.71 889.40 860.71 889.40 889.40 860.71 889.40 860.71 889.40 889.40 803.33 889.40 860.71 889.40 860.71 889.40 889.40 860.71 889.40 860.71 889.40 32.01 28.91 32.01 30.97 32.01 30.97 32.01 32.01 30.97 32.01 30.97 32.01 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 3 1 2 1 23.4 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
1 1121 A1121 1 344 500.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 3007.62 2716.56 3007.62 2910.60 3007.62 2910.60 3007.62 3007.62 2910.60 3007.62 2910.60 3007.62 5585.58 5045.04 5585.58 5405.40 5585.58 5405.40 5585.58 5585.58 5405.40 5585.58 5405.40 5585.58 20236.80 18278.40 20236.80 19584.00 20236.80 19584.00 20236.80 20236.80 19584.00 20236.80 19584.00 20236.80 9523.20 8601.60 9523.20 9216.00 9523.20 9216.00 9523.20 9523.20 9216.00 9523.20 9216.00 9523.20 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 1193.50 1078.00 1193.50 1155.00 1193.50 1155.00 1193.50 1193.50 1155.00 1193.50 1155.00 1193.50 1193.50 1078.00 1193.50 1155.00 1193.50 1155.00 1193.50 1193.50 1155.00 1193.50 1155.00 1193.50 12.45 11.25 12.45 12.05 12.45 12.05 12.45 12.45 12.05 12.45 12.05 12.45 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 403.0 364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0 403.0 390.0 403.0 1 1 0 0 34.7 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
1 1122 A1122 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 6240.30 5636.40 6240.30 6039.00 6240.30 6039.00 6240.30 6240.30 6039.00 6240.30 6039.00 6240.30 3989.70 3603.60 3989.70 3861.00 3989.70 3861.00 3989.70 3989.70 3861.00 3989.70 3861.00 3989.70 14880.00 13440.00 14880.00 14400.00 14880.00 14400.00 14880.00 14880.00 14400.00 14880.00 14400.00 14880.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 930.00 840.00 930.00 900.00 930.00 900.00 930.00 930.00 900.00 930.00 900.00 930.00 930.00 840.00 930.00 900.00 930.00 900.00 930.00 930.00 900.00 930.00 900.00 930.00 229.63 207.41 229.63 222.22 229.63 222.22 229.63 229.63 222.22 229.63 222.22 229.63 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 1 1 0 0 34.4 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
1 1123 A1123 1 344 400.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 2497.99 2256.25 2497.99 2417.41 2497.99 2417.41 2497.99 2497.99 2417.41 2497.99 2417.41 2497.99 1710.92 1545.35 1710.92 1655.73 1710.92 1655.73 1710.92 1710.92 1655.73 1710.92 1655.73 1710.92 9377.93 8470.39 9377.93 9075.42 9377.93 9075.42 9377.93 9377.93 9075.42 9377.93 9075.42 9377.93 70.87 64.01 70.87 68.58 70.87 68.58 70.87 70.87 68.58 70.87 68.58 70.87 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 342.19 309.07 342.19 331.15 342.19 331.15 342.19 342.19 331.15 342.19 331.15 342.19 456.25 412.10 456.25 441.53 456.25 441.53 456.25 456.25

```
441.53 456.25 441.53 456.25 4.98 4.50 4.98 4.82 4.98 4.82 4.98 4.98 4.82 4.98 4.82 4.98 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0
720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 1 0
1 21.6 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1124 A1124 1 344 200.00 0.00 0 1 0 0 2 0 0 2 1 -1 -1 -1 5783.91 5066.90 6357.52 5162.50 4923.50
4875.69 5497.11 10372.80 4875.69 5210.30 3728.47 6070.72 5783.91 5066.90 6357.52 5162.50 4923.50
4875.69 5497.11 10372.80 4875.69 5210.30 3728.47 6070.72 14880.00 13440.00 14880.00 14400.00
14880.00 14400.00 14880.00 14880.00 14400.00 14880.00 14400.00 14880.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00
15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 826.27 723.84 908.22 737.50 703.36 696.53
785.30 1481.83 696.53 744.33 532.64 867.25 1154.05 1031.13 1195.02 1085.76 1092.59 1065.28 1133.56
1481.83 1065.28 1113.08 983.33 1174.54 22.69 19.88 24.94 20.25 19.31 19.13 21.56 40.69 19.13 20.44 14.63
23.81 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 7.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 20.0 16.0 16.0 16.0 16.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0
744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0
720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 338.0 302.0 350.0 318.0 320.0 312.0 332.0 434.0
312.0 326.0 288.0 344.0 1 1 2 0 43.9 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

 1 1125 A1125 1 344 50.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 2625.70 2371.60 2625.70 2541.00 2625.70 2541.00
2625.70 2625.70 2541.00 2625.70 2541.00 2625.70 2625.70 2371.60 2625.70 2541.00 2625.70 2541.00
2625.70 2625.70 2541.00 2625.70 2541.00 2625.70 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00
16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 375.10
338.80 375.10 363.00 375.10 363.00 375.10 375.10 363.00 375.10 363.00 375.10 375.10 338.80 375.10
363.00 375.10 363.00 375.10 375.10 363.00 375.10 363.00 375.10 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0
6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0
6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0
372.0 372.0 360.0 372.0 360.0 372.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 310.0 280.0 310.0 300.0
310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 3 0 2 0 8.5 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0
31.0 30.0 31.0
 1 1126 A1126 1 344 100.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 2625.70 2371.60 2625.70 2541.00 2625.70
2541.00 2625.70 2625.70 2541.00 2625.70 2541.00 2625.70 2625.70 2371.60 2625.70 2541.00 2625.70
2541.00 2625.70 2625.70 2541.00 2625.70 2541.00 2625.70 722.30 652.40 722.30 699.00 722.30 699.00
722.30 722.30 699.00 722.30 699.00 722.30 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 375.10 338.80 375.10 363.00 375.10 363.00 375.10 375.10 363.00 375.10
363.00 375.10 375.10 338.80 375.10 363.00 375.10 363.00 375.10 375.10 363.00 375.10 363.00 375.10 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0
6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0
336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 310.0 280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 3 1 0 0 10.7 0 31.0 28.0 31.0
30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
```

1 1127 A1127 1 344 200.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 2625.70 2371.60 2625.70 2541.00 2625.70 2541.00 2625.70 2625.70 2541.00 2625.70 2541.00 2625.70 2625.70 2371.60 2625.70 2541.00 2625.70 2541.00 2625.70 2625.70 2541.00 2625.70 2541.00 2625.70 2625.70 2541.00 2625.70 2541.00 2625.70 1807.40 1632.49 1807.40 1749.10 1807.40 1749.10 1807.40 1807.40 1749.10 1807.40 1749.10 1807.40 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 450.12 406.56 450.12 435.60 450.12 435.60 450.12 450.12 435.60 450.12 435.60 450.12 450.12 406.56 450.12 435.60 450.12 435.60 450.12 450.12 435.60 450.12 435.60 450.12 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 310.0 280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 1 1 1 0 14.0 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1128 A1128 1 344 200.00 9.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 2863.60 2586.48 2863.60 2771.23 2863.60 2771.23 2863.60 2863.60 2771.23 2863.60 2771.23 2863.60 1830.83 1653.65 1830.83 1771.77 1830.83 1771.77 1830.83 1830.83 1771.77 1830.83 1771.77 1830.83 3720.00 3360.00 3720.00 3600.00 3720.00 3600.00 3720.00 3720.00 3600.00 3720.00 3600.00 3720.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 335.32 302.87 335.32 324.50 335.32 324.50 335.32 335.32 324.50 335.32 324.50 335.32 447.09 403.82 447.09 432.67 447.09 432.67 447.09 447.09 432.67 447.09 432.67 447.09 0.81 0.73 0.81 0.79 0.81 0.79 0.81 0.81 0.79 0.81 0.79 0.81 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 1 0 0 15.3 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1129 A1129 1 344 100.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 3667.30 3312.40 3667.30 3549.00 3667.30 3549.00 3667.30 3667.30 3549.00 3667.30 3549.00 3667.30 3667.30 3312.40 3667.30 3549.00 3667.30 3549.00 3667.30 3667.30 3549.00 3667.30 3549.00 3667.30 2100.25 1897.00 2100.25 2032.50 2100.25 2032.50 2100.25 2100.25 2032.50 2100.25 2032.50 2100.25 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 523.90 473.20 523.90 507.00 523.90 507.00 523.90 523.90 507.00 523.90 507.00 523.90 523.90 473.20 523.90 507.00 523.90 507.00 523.90 523.90 507.00 523.90 507.00 523.90 3720.00 3360.00 3720.00 3600.00 3720.00 3600.00 3720.00 3720.00 3600.00 3720.00 3600.00 3720.00 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 465.0 420.0 465.0 450.0 465.0 450.0 465.0 465.0 450.0 465.0 450.0 465.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 465.0 420.0 465.0 450.0 465.0 450.0 465.0 465.0 450.0 465.0 450.0 465.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 403.0 364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0 403.0 390.0 403.0 1 0 1 0 14.1 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1130 A1130 1 344 500.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 1139.57 1031.04 1139.57 1071.74 1126.01 1098.87 1139.57 1139.57 1085.31 1153.14 1112.44 1098.87 2116.35 1914.79 2116.35 1990.38 2091.16 2040.77 2116.35 2116.35 2015.57 2141.55 2065.96 2040.77 9349.69 8440.69 9349.69 9122.39 9382.13 9057.50 9349.69 9349.69 9089.94 9317.24 9025.06 9447.01 4399.85 3972.09 4399.85 4292.89 4415.12 4262.35 4399.85 4399.85 4277.62 4384.59 4247.09 4445.65 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 452.21 409.14 452.21 425.29 446.83 436.06 452.21 452.21 430.68 457.60 441.44 436.06 861.36 775.22 861.36 882.89 882.89 839.82 861.36 861.36 861.36 839.82 818.29 925.96 9.24 8.36 9.24 8.69 9.13 8.91 9.24 9.24 8.80 9.35 9.02 8.91 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 9.0 9.0 7.0 7.0 7.0 7.0 7.0 7.0

```
7.0 9.0 9.0 7.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 14.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 14.0 14.0 14.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 14.0 14.0 222.0 200.0 222.0 224.0 226.0 216.0 222.0 222.0 220.0 218.0 212.0 234.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 222.0
200.0 222.0 224.0 226.0 216.0 222.0 222.0 220.0 218.0 212.0 234.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 160.0 144.0 160.0 164.0 164.0 156.0 160.0 160.0 160.0 156.0 152.0 172.0 1 1 0 0 47.4 0 31.0 28.0 31.0
30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1131 A1131 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 1897.20 1713.60 1897.20 1836.00 1897.20
1836.00 1897.20 1897.20 1836.00 1897.20 1836.00 1897.20 3682.80 3326.40 3682.80 3564.00 3682.80
3564.00 3682.80 3682.80 3564.00 3682.80 3564.00 3682.80 781.20 705.60 781.20 756.00 781.20 756.00
781.20 781.20 756.00 781.20 756.00 781.20 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0
25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0
16.0 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 186.00 168.00 186.00 180.00 186.00 180.00 186.00 186.00 180.00 186.00
180.00 186.00 186.00 168.00 186.00 180.00 186.00 180.00 186.00 186.00 180.00 186.00 180.00 186.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0
10.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0
392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 3 2 2 1 7.1 -1 31.0 28.0 31.0
30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1132 A1132 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 2327.16 2105.53 2327.16 2188.64 2299.46
2244.05 2327.16 2327.16 2216.35 2354.87 2271.76 2244.05 1487.86 1346.16 1487.86 1399.30 1470.15
1434.72 1487.86 1487.86 1417.01 1505.57 1452.43 1434.72 6873.39 6205.14 6873.39 6706.29 6897.24
6658.59 6873.39 6873.39 6682.44 6849.54 6634.74 6944.94 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 346.82 313.79 346.82 326.18 342.69 334.43 346.82 346.82
330.31 350.95 338.56 334.43 660.61 594.55 660.61 677.13 677.13 644.09 660.61 660.61 660.61 644.09
627.58 710.16 224.00 202.67 224.00 210.67 221.33 216.00 224.00 224.00 213.33 226.67 218.67 216.00 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 9.0 9.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 7.0 11.0 11.0
11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 222.0 200.0 222.0 224.0 226.0 216.0 222.0 222.0 220.0 218.0 212.0 234.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 222.0 200.0 222.0 224.0 226.0
216.0 222.0 222.0 220.0 218.0 212.0 234.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 160.0 144.0 160.0
164.0 164.0 156.0 160.0 160.0 160.0 156.0 152.0 172.0 1 1 0 0 52.6 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0
30.0 31.0 30.0 31.0
 1 1133 A1133 1 344 200.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 37200.00 33600.00 37200.00 36000.00
37200.00 36000.00 37200.00 37200.00 36000.00 37200.00 36000.00 37200.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 2 0 2 0 59.9 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1134 A1134 1 344 100.00 0.00 0 0 0 0 2 0 0 2 3 -1 -1 -1 3687.45 3330.60 3687.45 3568.50 3687.45
3568.50 3687.45 3687.45 3568.50 3687.45 3568.50 3687.45 2357.55 2129.40 2357.55 2281.50 2357.55
2281.50 2357.55 2357.55 2281.50 2357.55 2281.50 2357.55 857.18 774.23 857.18 829.53 857.18 829.53
857.18 857.18 829.53 857.18 829.53 857.18 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0
```

```
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 604.50 546.00 604.50 585.00 604.50 585.00 604.50 604.50 585.00 604.50
585.00 604.50 759.94 686.40 759.94 735.43 759.94 735.43 759.94 759.94 735.43 759.94 735.43 759.94 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0
720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 341.0
308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 1 1 0 1 17.2 0 31.0 28.0 31.0 30.0 31.0
30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1135 A1135 1 344 80.00 0.00 0 0 0 0 2 0 0 2 3 -1 -1 -1 1512.80 1366.40 1512.80 1464.00 1512.80 1464.00
1512.80 1512.80 1464.00 1512.80 1464.00 1512.80 967.20 873.60 967.20 936.00 967.20 936.00 967.20
967.20 936.00 967.20 936.00 967.20 694.32 627.13 694.32 671.92 694.32 671.92 694.32 694.32 671.92
694.32 671.92 694.32 162.87 147.10 162.87 157.61 162.87 157.61 162.87 162.87 157.61 162.87 157.61
162.87 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 248.00 224.00 248.00 240.00 248.00 240.00 248.00 248.00
240.00 248.00 240.00 248.00 311.77 281.60 311.77 301.71 311.77 301.71 311.77 311.77 301.71 311.77
301.71 311.77 206.67 186.67 206.67 200.00 206.67 200.00 206.67 206.67 200.00 206.67 200.00 206.67 11.0
11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0
11.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 403.0
364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0 403.0 390.0 403.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 403.0 364.0 403.0 390.0 403.0 390.0 403.0
403.0 390.0 403.0 390.0 403.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 341.0 308.0 341.0 330.0 341.0
330.0 341.0 341.0 330.0 341.0 330.0 341.0 1 1 0 1 8.5 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0
30.0 31.0
 1 1136 A1136 1 344 150.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 1636.80 1478.40 1636.80 1584.00 1636.80
1584.00 1636.80 1636.80 1584.00 1636.80 1584.00 1636.80 1636.80 1478.40 1636.80 1584.00 1636.80
1584.00 1636.80 1636.80 1584.00 1636.80 1584.00 1636.80 386.57 349.16 386.57 374.10 386.57 374.10
386.57 386.57 374.10 386.57 374.10 386.57 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 27.0
27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0
24.0 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 327.36 295.68 327.36 316.80 327.36 316.80 327.36 327.36 316.80 327.36
316.80 327.36 494.13 446.31 494.13 478.19 494.13 478.19 494.13 494.13 478.19 494.13 478.19 494.13 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 2.0 2.0 2.0
2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 2.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 21.0 21.0 21.0 21.0 21.0 21.0
21.0 21.0 21.0 21.0 21.0 21.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0
21.0 21.0 21.0 21.0 21.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0
21.0 21.0 21.0 21.0 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 4.0 4.0 4.0 4.0
4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 372.0 336.0 372.0
360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 248.0
224.0 248.0 240.0 248.0 240.0 248.0 248.0 240.0 248.0 240.0 248.0 1 1 0 0 12.0 0 31.0 28.0 31.0 30.0 31.0
30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1137 A1137 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 6450.48 5826.24 6450.48 6242.40 6450.48
6242.40 6450.48 6450.48 6242.40 6450.48 6242.40 6450.48 12521.52 11309.76 12521.52 12117.60 12521.52
12117.60 12521.52 12521.52 12117.60 12521.52 12117.60 12521.52 5859.00 5292.00 5859.00 5670.00
5859.00 5670.00 5859.00 5859.00 5670.00 5859.00 5670.00 5859.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 1897.20 1713.60 1897.20 1836.00 1897.20 1836.00
1897.20 1897.20 1836.00 1897.20 1836.00 1897.20 1897.20 1713.60 1897.20 1836.00 1897.20 1836.00
1897.20 1897.20 1836.00 1897.20 1836.00 1897.20 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0 392.0
```

```
434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0
420.0 434.0 420.0 434.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0
372.0 372.0 360.0 372.0 360.0 372.0 1 1 0 1 32.3 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0
31.0
 1 1138 A1138 1 344 400.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 2599.87 2348.27 2599.87 2516.00 2599.87
2516.00 2599.87 2599.87 2516.00 2599.87 2516.00 2599.87 1780.70 1608.37 1780.70 1723.26 1780.70
1723.26 1780.70 1780.70 1723.26 1780.70 1723.26 1780.70 6036.25 5452.10 6036.25 5841.53 6036.25
5841.53 6036.25 6036.25 5841.53 6036.25 5841.53 6036.25 45.61 41.20 45.61 44.14 45.61 44.14 45.61 45.61
44.14 45.61 44.14 45.61 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 356.14 321.68 356.14 344.66 356.14 344.66
356.14 356.14 344.66 356.14 344.66 356.14 474.86 428.90 474.86 459.54 474.86 459.54 474.86 474.86
459.54 474.86 459.54 474.86 6.09 5.50 6.09 5.90 6.09 5.90 6.09 6.09 5.90 6.09 5.90 6.09 10.0 10.0 10.0 10.0
10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0
5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 434.0 392.0 434.0 420.0 434.0 420.0
434.0 434.0 420.0 434.0 420.0 434.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0
360.0 372.0 1 1 0 1 22.8 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1139 A1139 1 344 100.00 0.00 0 1 0 1 3 0 0 2 1 -1 -1 -1 1705.96 1540.00 1705.96 1666.34 1712.69
1652.88 1705.96 1705.96 1659.61 1699.23 1646.16 1726.15 1705.96 1540.00 1705.96 1666.34 1712.69
1652.88 1705.96 1705.96 1659.61 1699.23 1646.16 1726.15 544.46 492.19 544.46 519.53 541.24 525.97
544.46 544.46 522.75 547.68 529.19 534.80 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 243.71 220.00 243.71 238.05 244.67 236.13 243.71 243.71 237.09 242.75
235.17 246.59 335.34 303.30 335.34 317.18 332.14 323.59 335.34 335.34 320.39 338.54 326.80 325.73 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 5.0 5.0 5.0 5.0 5.0 5.0 6.0 5.0 7.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 376.0 340.0 376.0 357.0 373.0 363.0 376.0 376.0 360.0 379.0 366.0 367.0
5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 376.0
340.0 376.0 357.0 373.0 363.0 376.0 376.0 360.0 379.0 366.0 367.0 1 1 1 2 1 1 1 1 2 1 1 2 1 1 1 2 1 1 1 1 2 1
1 2 314.0 284.0 314.0 297.0 311.0 303.0 314.0 314.0 300.0 317.0 306.0 305.0 3 1 0 0 9.2 0 31.0 28.0 31.0
30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1140 A1140 1 344 200.00 0.00 0 0 0 0 3 0 0 2 1 -1 -1 -1 926.72 842.47 926.72 800.35 884.59 884.59
926.72 926.72 842.47 968.84 926.72 800.35 926.72 842.47 926.72 800.35 884.59 884.59 926.72 926.72
842.47 968.84 926.72 800.35 1573.00 1422.00 1573.00 1500.50 1563.50 1519.50 1573.00 1573.00 1510.00
1582.50 1529.00 1544.50 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0
25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00
16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 158.87 144.42 158.87 137.20 151.64 151.64 158.87 158.87 144.42 166.09 158.87 137.20
211.82 192.56 211.82 182.94 202.19 202.19 211.82 211.82 192.56 221.45 211.82 182.94 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 66.0 66.0 66.0 66.0 66.0 66.0 66.0 66.0 66.0 66.0 66.0 66.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 9.0 18.0 9.0 9.0 9.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0
14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0
14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 132.0 120.0 132.0 114.0 126.0 126.0 132.0 132.0 120.0
138.0 132.0 114.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 132.0 120.0 132.0 114.0 126.0 126.0 132.0 132.0 120.0 138.0 132.0 114.0 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 88.0 80.0 88.0 76.0 84.0 84.0 88.0 88.0 80.0 92.0 88.0 76.0 1 1 1 0 19.0 0 22.0 20.0
22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1141 A1141 1 344 200.00 9.00 0 0 0 0 3 0 0 2 2 -1 -1 -1 1860.01 1690.92 1860.01 1606.37 1775.47
1775.47 1860.01 1860.01 1690.92 1944.56 1860.01 1606.37 1189.19 1081.08 1189.19 1027.03 1135.13
1135.13 1189.19 1189.19 1081.08 1243.24 1189.19 1027.03 1125.74 1022.31 1125.74 991.58 1083.02
1077.03 1125.74 1125.74 1034.31 1168.47 1119.75 997.58 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
```

```
20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 217.80 198.00 217.80 188.10 207.90 207.90 217.80 217.80
198.00 227.70 217.80 188.10 217.80 198.00 217.80 188.10 207.90 207.90 217.80 217.80 198.00 227.70
217.80 188.10 2.64 2.40 2.64 2.28 2.52 2.52 2.64 2.64 2.40 2.76 2.64 2.28 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0
230.0 220.0 190.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0
16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 176.0 160.0 176.0 152.0 168.0 168.0 176.0 176.0 160.0 184.0 176.0 152.0 1 1 0 0
16.2 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1142 A1142 1 344 200.00 0.00 1 0 0 0 3 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 21412.45 19350.98 21412.45 20531.72
21329.49 20697.64 21412.45 21412.45 20614.68 21495.40 20780.60 21163.57 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 2 0 2 0 63.9 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1143 A1143 1 344 50.00 0.00 1 0 0 0 3 0 0 2 1 -1 -1 -1 1907.73 1724.80 1907.73 1816.27 1894.67 1842.40
1907.73 1907.73 1829.33 1920.80 1855.47 1868.53 1907.73 1724.80 1907.73 1816.27 1894.67 1842.40
1907.73 1907.73 1829.33 1920.80 1855.47 1868.53 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00
16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 272.53
246.40 272.53 259.47 270.67 263.20 272.53 272.53 261.33 274.40 265.07 266.93 272.53 246.40 272.53
259.47 270.67 263.20 272.53 272.53 261.33 274.40 265.07 266.93 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0
7.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
354.0 320.0 354.0 338.0 352.0 342.0 354.0 354.0 340.0 356.0 344.0 348.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0
6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 354.0 320.0 354.0 338.0 352.0 342.0
354.0 354.0 340.0 356.0 344.0 348.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 292.0 264.0 292.0 278.0
290.0 282.0 292.0 292.0 280.0 294.0 284.0 286.0 3 0 2 0 6.5 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0
31.0 30.0 31.0
 1 1144 A1144 1 344 400.00 0.00 1 0 0 0 3 0 0 2 2 -1 -1 -1 1963.66 1774.57 1963.66 1883.66 1956.39
1898.21 1963.66 1963.66 1890.93 1970.94 1905.48 1941.84 1344.95 1215.44 1344.95 1290.16 1339.97
1300.12 1344.95 1344.95 1295.14 1349.93 1305.10 1330.01 3965.45 3584.79 3965.45 3782.69 3941.51
3830.58 3965.45 3965.45 3806.63 3989.40 3854.53 3893.61 29.97 27.09 29.97 28.58 29.78 28.95 29.97 29.97
28.77 30.15 29.13 29.42 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 268.99 243.09 268.99 258.03 268.00 260.03
268.99 268.99 259.03 269.99 261.02 266.00 334.75 302.87 334.75 314.82 330.76 322.79 334.75 334.75
318.81 338.73 326.78 322.79 6.63 5.99 6.63 6.36 6.61 6.41 6.63 6.63 6.39 6.66 6.44 6.56 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0
5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 398.0 360.0 398.0 376.0 394.0 384.0
398.0 398.0 380.0 402.0 388.0 386.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 398.0 360.0 398.0 376.0 394.0 384.0 398.0 398.0 380.0 402.0 388.0 386.0 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 336.0 304.0 336.0 316.0 332.0 324.0 336.0 336.0 320.0 340.0
328.0 324.0 1 1 0 1 19.9 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1145 A1145 1 344 500.00 0.00 1 0 0 0 3 0 1 3 1 -1 -1 -1 1670.90 1509.20 1670.90 1617.00 1670.90
```

```
1617.00 1670.90 1670.90 1617.00 1670.90 1617.00 1670.90 1670.90 1509.20 1670.90 1617.00 1670.90
1617.00 1670.90 1670.90 1617.00 1670.90 1617.00 1670.90 372000.00 336000.00 372000.00 360000.00
372000.00 360000.00 372000.00 372000.00 360000.00 372000.00 360000.00 372000.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00
16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 238.70 215.60 238.70 231.00
238.70 231.00 238.70 238.70 231.00 238.70 231.00 238.70 238.70 215.60 238.70 231.00 238.70 231.00
238.70 238.70 231.00 238.70 231.00 238.70 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0
720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0
744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 248.0 224.0 248.0 240.0 248.0 240.0 248.0 248.0 240.0
248.0 240.0 248.0 1 0 0 0 506.7 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1146 A1146 1 344 100.00 0.00 1 0 0 0 2 0 1 2 2 -1 -1 -1 2038.50 1841.22 2038.50 1972.74 2038.50
1972.74 2038.50 2038.50 1972.74 2038.50 1972.74 2038.50 1303.30 1177.18 1303.30 1261.26 1303.30
1261.26 1303.30 1303.30 1261.26 1303.30 1261.26 1303.30 37200.00 33600.00 37200.00 36000.00 37200.00
36000.00 37200.00 37200.00 36000.00 37200.00 36000.00 37200.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 238.70 215.60 238.70 231.00 238.70 231.00 238.70
238.70 231.00 238.70 231.00 238.70 238.70 215.60 238.70 231.00 238.70 231.00 238.70 238.70 231.00
238.70 231.00 238.70 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0
744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 248.0 224.0 248.0 240.0 248.0 240.0 248.0 248.0 240.0 248.0 240.0 248.0 1 0 0 0
58.2 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1147 A1147 1 344 50.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 1524.60 1386.00 1524.60 1316.70 1455.30 1455.30
1524.60 1524.60 1386.00 1593.90 1524.60 1316.70 1524.60 1386.00 1524.60 1316.70 1455.30 1455.30
1524.60 1524.60 1386.00 1593.90 1524.60 1316.70 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00
16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 217.80
198.00 217.80 188.10 207.90 207.90 217.80 217.80 198.00 227.70 217.80 188.10 217.80 198.00 217.80
188.10 207.90 207.90 217.80 217.80 198.00 227.70 217.80 188.10 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0
16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0
16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0
220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0
220.0 220.0 200.0 230.0 220.0 190.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 176.0 160.0 176.0 152.0
168.0 168.0 176.0 176.0 160.0 184.0 176.0 152.0 3 0 2 0 8.7 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0
23.0 22.0 19.0
 1 1148 A1148 1 344 100.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 1524.60 1386.00 1524.60 1316.70 1455.30
1455.30 1524.60 1524.60 1386.00 1593.90 1524.60 1316.70 1524.60 1386.00 1524.60 1316.70 1455.30
1455.30 1524.60 1524.60 1386.00 1593.90 1524.60 1316.70 451.10 409.66 451.10 397.34 433.98 431.58
451.10 451.10 414.46 468.22 448.70 399.75 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 217.80 198.00 217.80 188.10 207.90 207.90 217.80 217.80 198.00 227.70
217.80 188.10 217.80 198.00 217.80 188.10 207.90 207.90 217.80 217.80 198.00 227.70 217.80 188.10 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0
```

16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 176.0 160.0 176.0 152.0 168.0 168.0 176.0 176.0 160.0 184.0 176.0 152.0 3 1 0 0 10.9 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1149 A1149 1 344 200.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 1524.60 1386.00 1524.60 1316.70 1455.30 1455.30 1524.60 1524.60 1386.00 1593.90 1524.60 1316.70 1524.60 1386.00 1524.60 1316.70 1455.30 1455.30 1524.60 1524.60 1386.00 1593.90 1524.60 1316.70 1125.74 1022.31 1125.74 991.58 1083.02 1077.03 1125.74 1125.74 1034.31 1168.47 1119.75 997.58 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 261.36 237.60 261.36 225.72 249.48 249.48 261.36 261.36 237.60 273.24 261.36 225.72 261.36 237.60 261.36 225.72 249.48 249.48 261.36 261.36 237.60 273.24 261.36 225.72 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 176.0 160.0 176.0 152.0 168.0 168.0 176.0 176.0 160.0 184.0 176.0 152.0 1 1 1 0 14.3 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1150 A1150 1 344 200.00 9.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 1601.68 1456.07 1601.68 1383.27 1528.87 1528.87 1601.68 1601.68 1456.07 1674.48 1601.68 1383.27 1024.02 930.93 1024.02 884.38 977.48 977.48 1024.02 1024.02 930.93 1070.57 1024.02 884.38 1231.50 1118.45 1231.50 1082.94 1183.98 1177.98 1231.50 1231.50 1130.46 1279.02 1225.49 1088.94 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 187.55 170.50 187.55 161.98 179.02 179.02 187.55 187.55 170.50 196.08 187.55 161.98 250.07 227.33 250.07 215.97 238.70 238.70 250.07 250.07 227.33 261.43 250.07 215.97 0.66 0.60 0.66 0.57 0.63 0.63 0.66 0.66 0.60 0.69 0.66 0.57 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 1 1 0 0 14.3 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1151 A1151 1 344 500.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 731.81 665.28 731.81 632.02 698.54 698.54 731.81 731.81 665.28 765.07 731.81 632.02 1359.07 1235.52 1359.07 1173.74 1297.30 1297.30 1359.07 1359.07 1235.52 1420.85 1359.07 1173.74 6696.64 6081.92 6696.64 5888.80 6438.24 6405.60 6696.64 6696.64 6147.20 6955.04 6664.00 5921.44 3151.36 2862.08 3151.36 2771.20 3029.76 3014.40 3151.36 3151.36 2892.80 3272.96 3136.00 2786.56 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 290.40 264.00 290.40 250.80 277.20 277.20 290.40 290.40 264.00 303.60 290.40 250.80 387.20 352.00 387.20 334.40 369.60 369.60 387.20 387.20 352.00 404.80 387.20 334.40 9.68 8.80 9.68 8.36 9.24 9.24 9.68 9.68 8.80 10.12 9.68 8.36 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 132.0 120.0 132.0 114.0 126.0 126.0 132.0 132.0 120.0 138.0 132.0 114.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 132.0 120.0 132.0 114.0 126.0 126.0 132.0 132.0 120.0 138.0 132.0 114.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 88.0 80.0 88.0 76.0 84.0 84.0 88.0 88.0 80.0 92.0 88.0 76.0 1 1 0 0 44.7 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1152 A1152 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 2254.56 2049.60 2254.56 1947.12 2152.08 2152.08 2254.56 2254.56 2049.60 2357.04 2254.56 1947.12 1441.44 1310.40 1441.44 1244.88 1375.92 1375.92 1441.44 1441.44 1310.40 1506.96 1441.44 1244.88 324.13 294.23 324.13 287.65 312.77 310.38

```
324.13 324.13 299.02 335.50 321.74 290.05 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 264.00 240.00 264.00 228.00 252.00 252.00 264.00 264.00 240.00 276.00
264.00 228.00 264.00 240.00 264.00 228.00 252.00 252.00 264.00 264.00 240.00 276.00 264.00 228.00 16.50
15.00 16.50 14.25 15.75 15.75 16.50 16.50 15.00 17.25 16.50 14.25 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0
230.0 220.0 190.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0
16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 176.0 160.0 176.0 152.0 168.0 168.0 176.0 176.0 160.0 184.0 176.0 152.0 1 1 2 1
14.3 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1153 A1153 1 344 800.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 1889.54 1717.76 1889.54 1631.87 1803.65
1803.65 1889.54 1889.54 1717.76 1975.42 1889.54 1631.87 1208.06 1098.24 1208.06 1043.33 1153.15
1153.15 1208.06 1208.06 1098.24 1262.98 1208.06 1043.33 2590.87 2353.22 2590.87 2275.13 2489.50
2477.86 2590.87 2590.87 2376.50 2692.23 2579.23 2286.77 80.13 72.78 80.13 70.36 76.99 76.63 80.13 80.13
73.50 83.26 79.77 70.72 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 232.32 211.20 232.32 200.64 221.76 221.76
232.32 232.32 211.20 242.88 232.32 200.64 309.76 281.60 309.76 267.52 295.68 295.68 309.76 309.76
281.60 323.84 309.76 267.52 7.26 6.60 7.26 6.27 6.93 6.93 7.26 7.26 6.60 7.59 7.26 6.27 10.0 10.0 10.0 10.0
10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0
5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0
308.0 308.0 280.0 322.0 308.0 266.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0
264.0 228.0 1 1 0 1 20.2 -1 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1154 A1154 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 1417.15 1288.32 1417.15 1223.90 1352.74
1352.74 1417.15 1417.15 1288.32 1481.57 1417.15 1223.90 906.05 823.68 906.05 782.50 864.86 864.86
906.05 906.05 823.68 947.23 906.05 782.50 4924.00 4472.00 4924.00 4330.00 4734.00 4710.00 4924.00
4924.00 4520.00 5114.00 4900.00 4354.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0
25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 211.20 192.00 211.20 182.40 201.60 201.60 211.20 211.20 192.00 220.80
211.20 182.40 281.60 256.00 281.60 243.20 268.80 268.80 281.60 281.60 256.00 294.40 281.60 243.20
144.00 130.91 144.00 124.36 137.45 137.45 144.00 144.00 130.91 150.55 144.00 124.36 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 11.0 11.0 11.0 11.0 11.0
11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 11.0 11.0 11.0
11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 132.0
120.0 132.0 114.0 126.0 126.0 132.0 132.0 120.0 138.0 132.0 114.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 132.0 120.0 132.0 114.0 126.0 126.0 132.0
132.0 120.0 138.0 132.0 114.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 88.0 80.0 88.0 76.0 84.0 84.0
88.0 88.0 80.0 92.0 88.0 76.0 1 1 0 0 46.2 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1155 A1155 1 344 200.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 24082.40 21751.85 24082.40 23305.55
24082.40 23305.55 24082.40 24082.40 23305.55 24082.40 23305.55 24082.40 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
```

0.0 0.0 2 0 2 0 59.1 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
1 1156 A1156 1 344 500.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 598.75 544.32 598.75 517.10 571.54 571.54 598.75 598.75 544.32 625.97 598.75 517.10 1618.85 1471.68 1618.85 1398.10 1545.26 1545.26 1618.85 1618.85 1471.68 1692.43 1618.85 1398.10 1336.72 1214.11 1336.72 1173.82 1284.42 1278.42 1336.72 1336.72 1226.12 1389.02 1330.71 1179.83 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 123.20 112.00 123.20 106.40 117.60 117.60 123.20 123.20 112.00 128.80 123.20 106.40 164.27 149.33 164.27 141.87 156.80 156.80 164.27 164.27 149.33 171.73 164.27 141.87 4.30 3.91 4.30 3.71 4.11 4.11 4.30 4.30 3.91 4.50 4.30 3.71 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 1 1 2 1 8.5 -1 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
1 1157 A1157 1 344 200.00 13.20 0 1 0 1 2 0 0 2 2 -1 -1 -1 3058.15 2765.06 3058.15 2908.69 3035.96 2953.06 3058.15 3058.15 2930.87 3080.33 2975.24 2991.59 3058.15 2765.06 3058.15 2908.69 3035.96 2953.06 3058.15 3058.15 2930.87 3080.33 2975.24 2991.59 1154.36 1042.65 1154.36 1117.12 1154.36 1117.12 1154.36 1154.36 1117.12 1154.36 1117.12 1154.36 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 392.07 354.50 392.07 372.91 389.23 378.60 392.07 392.07 375.75 394.91 381.44 383.54 543.42 491.02 543.42 522.46 541.92 525.46 543.42 543.42 523.96 544.92 526.95 538.93 1.26 1.14 1.26 1.20 1.25 1.21 1.26 1.26 1.20 1.27 1.22 1.23 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 425.0 384.0 425.0 409.0 424.0 411.0 425.0 425.0 410.0 426.0 412.0 422.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 425.0 384.0 425.0 409.0 424.0 411.0 425.0 425.0 410.0 426.0 412.0 422.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 363.0 328.0 363.0 349.0 362.0 351.0 363.0 363.0 350.0 364.0 352.0 360.0 1 1 2 1 15.2 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
1 1158 A1158 1 344 400.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 1426.14 1296.49 1426.14 1231.66 1361.31 1361.31 1426.14 1426.14 1296.49 1490.96 1426.14 1231.66 976.79 887.99 976.79 843.59 932.39 932.39 976.79 976.79 887.99 1021.19 976.79 843.59 3367.58 3058.68 3367.58 2957.19 3235.82 3220.69 3367.58 3367.58 3088.94 3499.33 3352.45 2972.32 25.45 23.11 25.45 22.35 24.45 24.34 25.45 25.45 23.34 26.44 25.33 22.46 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 195.36 177.60 195.36 168.72 186.48 186.48 195.36 195.36 177.60 204.24 195.36 168.72 260.48 236.80 260.48 224.96 248.64 248.64 260.48 260.48 236.80 272.32 260.48 224.96 6.48 5.90 6.48 5.60 6.19 6.19 6.48 6.48 5.90 6.78 6.48 5.60 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 1 1 0 1 21.0 -1 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
1 1159 A1159 1 344 50.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 1694.00 1540.00 1694.00 1463.00 1617.00 1617.00 1694.00 1694.00 1540.00 1771.00 1694.00 1463.00 1694.00 1540.00 1694.00 1463.00 1617.00 1617.00 1694.00 1694.00 1540.00 1771.00 1694.00 1463.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 242.00 220.00 242.00 209.00 231.00 231.00 242.00 242.00 220.00 253.00 242.00 209.00 242.00 220.00 242.00 209.00 231.00 231.00 242.00 242.00 220.00 253.00 242.00 209.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0

6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 3 0 2 0 7.7 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1160 A1160 1 344 100.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 1694.00 1540.00 1694.00 1463.00 1617.00 1617.00 1694.00 1694.00 1540.00 1771.00 1694.00 1463.00 1694.00 1540.00 1694.00 1463.00 1617.00 1617.00 1694.00 1694.00 1540.00 1771.00 1694.00 1463.00 492.40 447.20 492.40 433.00 473.40 471.00 492.40 492.40 452.00 511.40 490.00 435.40 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 242.00 220.00 242.00 209.00 231.00 231.00 242.00 242.00 220.00 253.00 242.00 209.00 242.00 220.00 242.00 209.00 231.00 231.00 242.00 242.00 220.00 253.00 242.00 209.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 3 1 0 0 9.7 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1161 A1161 1 344 200.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 1694.00 1540.00 1694.00 1463.00 1617.00 1617.00 1694.00 1694.00 1540.00 1771.00 1694.00 1463.00 1694.00 1540.00 1694.00 1463.00 1617.00 1617.00 1694.00 1694.00 1540.00 1771.00 1694.00 1463.00 1231.00 1118.00 1231.00 1082.50 1183.50 1177.50 1231.00 1231.00 1130.00 1278.50 1225.00 1088.50 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 290.40 264.00 290.40 250.80 277.20 277.20 290.40 290.40 264.00 303.60 290.40 250.80 290.40 264.00 290.40 250.80 277.20 277.20 290.40 290.40 264.00 303.60 290.40 250.80 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 1 1 1 0 12.7 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1162 A1162 1 344 200.00 9.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 1446.68 1315.16 1446.68 1249.40 1380.92 1380.92 1446.68 1446.68 1315.16 1512.43 1446.68 1249.40 924.92 840.84 924.92 798.80 882.88 882.88 924.92 924.92 840.84 966.97 924.92 798.80 1125.74 1022.31 1125.74 991.58 1083.02 1077.03 1125.74 1125.74 1034.31 1168.47 1119.75 997.58 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 169.40 154.00 169.40 146.30 161.70 161.70 169.40 169.40 154.00 177.10 169.40 146.30 262.45 238.59 262.45 226.66 250.52 250.52 262.45 262.45 238.59 274.38 262.45 226.66 1.32 1.20 1.32 1.14 1.26 1.26 1.32 1.32 1.20 1.38 1.32 1.14 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 286.0 260.0 286.0 247.0 273.0 273.0 286.0 286.0 260.0 299.0 286.0 247.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 286.0 260.0 286.0 247.0 273.0 273.0 286.0 286.0 260.0 299.0 286.0 247.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 242.0 220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 1 1 0 0 14.9 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1163 A1163 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 841.50 765.00 841.50 726.75 803.25 803.25 841.50 841.50 765.00 879.75 841.50 726.75 1633.50 1485.00 1633.50 1410.75 1559.25 1559.25 1633.50

```
1633.50 1485.00 1707.75 1633.50 1410.75 491.42 446.31 491.42 432.14 472.46 470.07 491.42 491.42 451.10
510.39 489.03 434.54 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0
25.0 25.0 25.0 25.0 25.0 25.0 25.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 14.00 14.00
14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 82.50 75.00 82.50 71.25 78.75 78.75 82.50 82.50 75.00 86.25 82.50 71.25 92.81 84.38 92.81
80.16 88.59 88.59 92.81 92.81 84.38 97.03 92.81 80.16 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 242.0 220.0
242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 242.0 220.0 242.0 209.0 231.0 231.0 242.0 242.0
220.0 253.0 242.0 209.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 198.0 180.0 198.0 171.0 189.0 189.0
198.0 198.0 180.0 207.0 198.0 171.0 3 2 2 1 6.5 -1 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1164 A1164 1 344 500.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 594.59 540.54 594.59 513.51 567.57 567.57
594.59 594.59 540.54 621.62 594.59 513.51 1104.25 1003.86 1104.25 953.67 1054.05 1054.05 1104.25
1104.25 1003.86 1154.44 1104.25 953.67 4991.20 4531.52 4991.20 4415.92 4810.32 4777.68 4991.20
4991.20 4596.80 5172.08 4958.56 4448.56 2348.80 2132.48 2348.80 2078.08 2263.68 2248.32 2348.80
2348.80 2163.20 2433.92 2333.44 2093.44 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50
14.50 14.50 14.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 235.95 214.50 235.95 203.78
225.23 225.23 235.95 235.95 214.50 246.68 235.95 203.78 314.60 286.00 314.60 271.70 300.30 300.30
314.60 314.60 286.00 328.90 314.60 271.70 14.52 13.20 14.52 12.54 13.86 13.86 14.52 14.52 13.20 15.18
14.52 12.54 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 9.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 14.0 14.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 14.0 132.0 120.0 132.0 114.0 126.0 126.0 132.0 132.0 120.0 138.0 132.0 114.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 132.0 120.0
132.0 114.0 126.0 126.0 132.0 132.0 120.0 138.0 132.0 114.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
88.0 80.0 88.0 76.0 84.0 84.0 88.0 88.0 80.0 92.0 88.0 76.0 1 1 0 0 33.0 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0
22.0 20.0 23.0 22.0 19.0
 1 1165 A1165 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 1328.58 1207.80 1328.58 1147.41 1268.19
1268.19 1328.58 1328.58 1207.80 1388.97 1328.58 1147.41 849.42 772.20 849.42 733.59 810.81 810.81
849.42 849.42 772.20 888.03 849.42 733.59 3668.96 3331.05 3668.96 3246.08 3536.00 3512.00 3668.96
3668.96 3379.04 3801.92 3644.97 3270.07 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0
25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 198.00 180.00 198.00 171.00 189.00 189.00 198.00 198.00 180.00 207.00
198.00 171.00 264.00 240.00 264.00 228.00 252.00 252.00 264.00 264.00 240.00 276.00 264.00 228.00
264.00 240.00 264.00 228.00 252.00 252.00 264.00 264.00 240.00 276.00 264.00 228.00 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 11.0 11.0 11.0 11.0 11.0
11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 11.0 11.0 11.0
11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 132.0
120.0 132.0 114.0 126.0 126.0 132.0 132.0 120.0 138.0 132.0 114.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 132.0 120.0 132.0 114.0 126.0 126.0 132.0
132.0 120.0 138.0 132.0 114.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 88.0 80.0 88.0 76.0 84.0 84.0
88.0 88.0 80.0 92.0 88.0 76.0 1 1 0 0 37.8 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1166 A1166 1 344 200.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25575.00 23100.00 25575.00 24750.00
25575.00 24750.00 25575.00 25575.00 24750.00 25575.00 24750.00 25575.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
```

0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 0 2 0 59.9 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

1 1167 A1167 1 344 100.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 1401.40 1274.00 1401.40 1210.30 1337.70 1337.70 1401.40 1401.40 1274.00 1465.10 1401.40 1210.30 1401.40 1274.00 1401.40 1210.30 1337.70 1337.70 1401.40 1401.40 1274.00 1465.10 1401.40 1210.30 1231.49 1118.44 1231.49 1082.93 1183.97 1177.97 1231.49 1231.49 1130.45 1279.01 1225.49 1088.93 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 200.20 182.00 200.20 172.90 191.10 191.10 200.20 200.20 182.00 209.30 200.20 172.90 225.23 204.75 225.23 194.51 214.99 214.99 225.23 225.23 204.75 235.46 225.23 194.51 2640.00 2400.00 2640.00 2280.00 2520.00 2520.00 2640.00 2640.00 2400.00 2760.00 2640.00 2280.00 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 242.0 220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 242.0 220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 198.0 180.0 198.0 171.0 189.0 189.0 198.0 198.0 180.0 207.0 198.0 171.0 1 0 1 0 12.2 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1168 A1168 1 344 500.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 1632.71 1474.70 1632.71 1580.04 1632.71 1580.04 1632.71 1632.71 1580.04 1632.71 1580.04 1632.71 2663.89 2406.10 2663.89 2577.96 2663.89 2577.96 2663.89 2663.89 2577.96 2663.89 2577.96 2663.89 1804.12 1629.52 1804.12 1745.92 1804.12 1745.92 1804.12 1804.12 1745.92 1804.12 1745.92 1804.12 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 238.70 215.60 238.70 231.00 238.70 231.00 238.70 238.70 231.00 238.70 231.00 238.70 238.70 215.60 238.70 231.00 238.70 231.00 238.70 238.70 231.00 238.70 231.00 238.70 26.04 23.52 26.04 25.20 26.04 25.20 26.04 26.04 25.20 26.04 25.20 26.04 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 1 2 1 9.0 -1 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

1 1169 A1169 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 2254.56 2049.60 2254.56 1947.12 2152.08 2152.08 2254.56 2254.56 2049.60 2357.04 2254.56 1947.12 1441.44 1310.40 1441.44 1244.88 1375.92 1375.92 1441.44 1441.44 1310.40 1506.96 1441.44 1244.88 408.80 371.20 408.80 360.80 393.60 391.20 408.80 408.80 376.00 424.00 406.40 363.20 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 264.00 240.00 264.00 228.00 252.00 252.00 264.00 264.00 240.00 276.00 264.00 228.00 409.01 371.83 409.01 353.24 390.42 390.42 409.01 409.01 371.83 427.61 409.01 353.24 6.60 6.00 6.60 5.70 6.30 6.30 6.60 6.60 6.00 6.90 6.60 5.70 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 286.0 260.0 286.0 247.0 273.0 273.0 286.0 286.0 260.0 299.0 286.0 247.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 286.0 260.0 286.0 247.0 273.0 273.0 286.0 286.0 260.0 299.0 286.0 247.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 242.0 220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 1 1 2 1 16.4 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1170 A1170 1 344 400.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 1256.40 1142.18 1256.40 1085.07 1199.29 1199.29 1256.40 1256.40 1142.18 1313.51 1256.40 1085.07 860.53 782.30 860.53 743.19 821.42 821.42 860.53 860.53 782.30 899.65 860.53 743.19 2248.05 2041.69 2248.05 1976.86 2161.31 2150.35 2248.05 2248.05 2063.61 2334.80 2237.10 1987.82 16.99 15.43 16.99 14.94 16.33 16.25 16.99 16.99 15.59 17.64 16.91 15.02 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 165.38 150.35 165.38 142.83 157.87 157.87 165.38 165.38

150.35 172.90 165.38 142.83 256.23 232.94 256.23 221.29 244.58 244.58 256.23 256.23 232.94 267.88 256.23 221.29 5.34 4.86 5.34 4.61 5.10 5.10 5.34 5.34 4.86 5.58 5.34 4.61 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 286.0 260.0 286.0 247.0 273.0 273.0 286.0 286.0 260.0 299.0 286.0 247.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 286.0 260.0 286.0 247.0 273.0 273.0 286.0 286.0 260.0 299.0 286.0 247.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 242.0 220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 1 1 0 1 17.2 -1 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1171 A1171 1 344 50.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 1863.40 1694.00 1863.40 1609.30 1778.70 1778.70 1863.40 1863.40 1694.00 1948.10 1863.40 1609.30 1863.40 1694.00 1863.40 1609.30 1778.70 1778.70 1863.40 1863.40 1694.00 1948.10 1863.40 1609.30 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 266.20 242.00 266.20 229.90 254.10 254.10 266.20 266.20 242.00 278.30 266.20 229.90 266.20 242.00 266.20 229.90 254.10 254.10 266.20 266.20 242.00 278.30 266.20 229.90 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 3 0 2 0 8.5 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1172 A1172 1 344 100.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 1863.40 1694.00 1863.40 1609.30 1778.70 1778.70 1863.40 1863.40 1694.00 1948.10 1863.40 1609.30 1863.40 1694.00 1863.40 1609.30 1778.70 1778.70 1863.40 1863.40 1694.00 1948.10 1863.40 1609.30 534.20 485.20 534.20 469.10 513.30 510.90 534.20 534.20 490.00 555.10 531.80 471.50 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 266.20 242.00 266.20 229.90 254.10 254.10 266.20 266.20 242.00 278.30 266.20 229.90 266.20 242.00 266.20 229.90 254.10 254.10 266.20 266.20 242.00 278.30 266.20 229.90 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 3 1 0 0 10.7 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1173 A1173 1 344 200.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 1863.40 1694.00 1863.40 1609.30 1778.70 1778.70 1863.40 1863.40 1694.00 1948.10 1863.40 1609.30 1863.40 1694.00 1863.40 1609.30 1778.70 1778.70 1863.40 1863.40 1694.00 1948.10 1863.40 1609.30 1336.72 1214.11 1336.72 1173.82 1284.42 1278.42 1336.72 1336.72 1226.12 1389.02 1330.71 1179.83 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 319.44 290.40 319.44 275.88 304.92 304.92 319.44 319.44 290.40 333.96 319.44 275.88 319.44 290.40 319.44 275.88 304.92 304.92 319.44 319.44 290.40 333.96 319.44 275.88 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 1 1 1 0 14.0 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

```
 1 1174 A1174 1 344 200.00 9.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 1860.01 1690.92 1860.01 1606.37 1775.47
1775.47 1860.01 1860.01 1690.92 1944.56 1860.01 1606.37 1189.19 1081.08 1189.19 1027.03 1135.13
1135.13 1189.19 1189.19 1081.08 1243.24 1189.19 1027.03 1336.72 1214.11 1336.72 1173.82 1284.42
1278.42 1336.72 1336.72 1226.12 1389.02 1330.71 1179.83 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 217.80 198.00 217.80 188.10 207.90 207.90 217.80 217.80
198.00 227.70 217.80 188.10 290.40 264.00 290.40 250.80 277.20 277.20 290.40 290.40 264.00 303.60
290.40 250.80 0.66 0.60 0.66 0.57 0.63 0.63 0.66 0.66 0.60 0.69 0.66 0.57 10.0 10.0 10.0 10.0 10.0 10.0 10.0
10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0
322.0 308.0 266.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 1 1 0 0
14.9 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1175 A1175 1 344 200.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 19200.00 17400.00 19200.00 17550.00
18750.00 18450.00 19200.00 19200.00 18000.00 19650.00 18900.00 17850.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 2 0 2 0 67.4 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1176 A1176 1 344 445.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 1281.60 1165.09 1281.60 1106.83 1223.34
1223.34 1281.60 1281.60 1165.09 1339.85 1281.60 1106.83 877.79 797.99 877.79 758.09 837.89 837.89
877.79 877.79 797.99 917.69 877.79 758.09 2638.86 2396.81 2638.86 2317.28 2535.62 2523.76 2638.86
2638.86 2420.52 2742.11 2627.01 2329.13 19.94 18.11 19.94 17.51 19.16 19.07 19.94 19.94 18.29 20.72
19.85 17.60 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 162.36 147.60 162.36 140.22 154.98 154.98 162.36 162.36
147.60 169.74 162.36 140.22 216.48 196.80 216.48 186.96 206.64 206.64 216.48 216.48 196.80 226.32
216.48 186.96 4.32 3.93 4.32 3.73 4.13 4.13 4.32 4.32 3.93 4.52 4.32 3.73 10.0 10.0 10.0 10.0 10.0 10.0 10.0
10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0
322.0 308.0 266.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 1 1 0 1
17.3 -1 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1177 A1177 1 344 50.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 1524.60 1386.00 1524.60 1316.70 1455.30 1455.30
1524.60 1524.60 1386.00 1593.90 1524.60 1316.70 1524.60 1386.00 1524.60 1316.70 1455.30 1455.30
1524.60 1524.60 1386.00 1593.90 1524.60 1316.70 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00
16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 217.80
198.00 217.80 188.10 207.90 207.90 217.80 217.80 198.00 227.70 217.80 188.10 245.02 222.75 245.02
211.61 233.89 233.89 245.02 245.02 222.75 256.16 245.02 211.61 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0
6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
242.0 220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
```

7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 242.0 220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 198.0 180.0 198.0 171.0 189.0 189.0 198.0 198.0 180.0 207.0 198.0 171.0 3 0 2 0 8.7 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1178 A1178 1 344 100.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 1558.48 1416.80 1558.48 1345.96 1487.64 1487.64 1558.48 1558.48 1416.80 1629.32 1558.48 1345.96 1558.48 1416.80 1558.48 1345.96 1487.64 1487.64 1558.48 1558.48 1416.80 1629.32 1558.48 1345.96 534.20 485.20 534.20 469.10 513.30 510.90 534.20 534.20 490.00 555.10 531.80 471.50 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 222.64 202.40 222.64 192.28 212.52 212.52 222.64 222.64 202.40 232.76 222.64 192.28 296.85 269.87 296.85 256.37 283.36 283.36 296.85 296.85 269.87 310.35 296.85 256.37 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 3 1 0 0 10.1 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1179 A1179 1 344 200.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 1558.48 1416.80 1558.48 1345.96 1487.64 1487.64 1558.48 1558.48 1416.80 1629.32 1558.48 1345.96 1558.48 1416.80 1558.48 1345.96 1487.64 1487.64 1558.48 1558.48 1416.80 1629.32 1558.48 1345.96 1336.72 1214.11 1336.72 1173.82 1284.42 1278.42 1336.72 1336.72 1226.12 1389.02 1330.71 1179.83 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 267.17 242.88 267.17 230.74 255.02 255.02 267.17 267.17 242.88 279.31 267.17 230.74 356.22 323.84 356.22 307.65 340.03 340.03 356.22 356.22 323.84 372.42 356.22 307.65 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 1 1 1 0 13.4 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1180 A1180 1 344 200.00 9.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 1756.68 1596.98 1756.68 1517.13 1676.83 1676.83 1756.68 1756.68 1596.98 1836.53 1756.68 1517.13 1123.12 1021.02 1123.12 969.97 1072.07 1072.07 1123.12 1123.12 1021.02 1174.17 1123.12 969.97 1336.72 1214.11 1336.72 1173.82 1284.42 1278.42 1336.72 1336.72 1226.12 1389.02 1330.71 1179.83 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 205.70 187.00 205.70 177.65 196.35 196.35 205.70 205.70 187.00 215.05 205.70 177.65 274.27 249.33 274.27 236.87 261.80 261.80 274.27 274.27 249.33 286.73 274.27 236.87 0.66 0.60 0.66 0.57 0.63 0.63 0.66 0.66 0.60 0.69 0.66 0.57 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 308.0 280.0 308.0 266.0 294.0 294.0 308.0 308.0 280.0 322.0 308.0 266.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 1 1 0 0 14.4 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1181 A1181 1 344 500.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 762.30 693.00 762.30 658.35 727.65 727.65 762.30 762.30 693.00 796.95 762.30 658.35 1415.70 1287.00 1415.70 1222.65 1351.35 1351.35 1415.70 1415.70 1287.00 1480.05 1415.70 1222.65 6696.64 6081.92 6696.64 5888.80 6438.24 6405.60 6696.64 6696.64 6147.20 6955.04 6664.00 5921.44 3151.36 2862.08 3151.36 2771.20 3029.76 3014.40 3151.36 3151.36 2892.80 3272.96 3136.00 2786.56 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50

```
14.50 14.50 14.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 302.50 275.00 302.50 261.25
288.75 288.75 302.50 302.50 275.00 316.25 302.50 261.25 403.33 366.67 403.33 348.33 385.00 385.00
403.33 403.33 366.67 421.67 403.33 348.33 7.26 6.60 7.26 6.27 6.93 6.93 7.26 7.26 6.60 7.59 7.26 6.27 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 11.0 11.0
11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 132.0 120.0 132.0 114.0 126.0 126.0 132.0 132.0 120.0 138.0 132.0 114.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 132.0 120.0 132.0 114.0 126.0
126.0 132.0 132.0 120.0 138.0 132.0 114.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 88.0 80.0 88.0 76.0
84.0 84.0 88.0 88.0 80.0 92.0 88.0 76.0 1 1 0 0 45.2 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0
19.0
 1 1182 A1182 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 2442.44 2220.40 2442.44 2109.38 2331.42
2331.42 2442.44 2442.44 2220.40 2553.46 2442.44 2109.38 1561.56 1419.60 1561.56 1348.62 1490.58
1490.58 1561.56 1561.56 1419.60 1632.54 1561.56 1348.62 408.80 371.20 408.80 360.80 393.60 391.20
408.80 408.80 376.00 424.00 406.40 363.20 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 286.00 260.00 286.00 247.00 273.00 273.00 286.00 286.00 260.00 299.00
286.00 247.00 321.75 292.50 321.75 277.88 307.12 307.12 321.75 321.75 292.50 336.38 321.75 277.88 6.60
6.00 6.60 5.70 6.30 6.30 6.60 6.60 6.00 6.90 6.60 5.70 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0
13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 242.0 220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 242.0
220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 198.0 180.0 198.0 171.0 189.0 189.0 198.0 198.0 180.0 207.0 198.0 171.0 1 1 2 1 15.9 0 22.0 20.0 22.0
19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1183 A1183 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 1141.59 1037.81 1141.59 985.92 1089.70 1089.70
1141.59 1141.59 1037.81 1193.49 1141.59 985.92 729.87 663.52 729.87 630.34 696.70 696.70 729.87 729.87
663.52 763.05 729.87 630.34 4507.01 4092.91 4507.01 3969.89 4335.97 4311.96 4507.01 4507.01 4140.93
4678.05 4483.00 3993.89 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0
25.0 25.0 25.0 25.0 25.0 25.0 25.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 17.50 17.50
17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 170.13 154.67 170.13 146.93 162.40 162.40 170.13 170.13 154.67 177.87 170.13 146.93
226.84 206.22 226.84 195.91 216.53 216.53 226.84 226.84 206.22 237.16 226.84 195.91 176.00 160.00
176.00 152.00 168.00 168.00 176.00 176.00 160.00 184.00 176.00 152.00 18.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0
11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 11.0 11.0 11.0 11.0 11.0 11.0
11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 9.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 132.0 120.0 132.0 114.0
126.0 126.0 132.0 132.0 120.0 138.0 132.0 114.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 132.0 120.0 132.0 114.0 126.0 126.0 132.0 132.0 120.0 138.0
132.0 114.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 88.0 80.0 88.0 76.0 84.0 84.0 88.0 88.0 80.0 92.0
88.0 76.0 1 1 0 0 39.7 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1184 A1184 1 344 200.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 18375.00 16650.00 18375.00 16837.50
17962.50 17662.50 18375.00 18375.00 17250.00 18787.50 18075.00 17137.50 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 2 0 2 0 63.7 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1185 A1185 1 344 100.00 0.00 0 0 0 0 2 0 0 2 3 -1 -1 -1 1919.06 1744.60 1919.06 1657.37 1831.83
```

1831.83 1919.06 1919.06 1744.60 2006.29 1919.06 1657.37 1226.94 1115.40 1226.94 1059.63 1171.17 1171.17 1226.94 1226.94 1115.40 1282.71 1226.94 1059.63 1336.72 1214.11 1336.72 1173.82 1284.42 1278.42 1336.72 1336.72 1226.12 1389.02 1330.71 1179.83 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 314.60 286.00 314.60 271.70 300.30 300.30 314.60 314.60 286.00 328.90 314.60 271.70 314.60 286.00 314.60 271.70 300.30 300.30 314.60 314.60 286.00 328.90 314.60 271.70 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 264.0 240.0 264.0 228.0 252.0 252.0 264.0 264.0 240.0 276.0 264.0 228.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 1 1 0 1 14.2 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1186 A1186 1 344 400.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 1381.57 1255.97 1381.57 1193.17 1318.77 1318.77 1381.57 1381.57 1255.97 1444.37 1381.57 1193.17 946.27 860.24 946.27 817.23 903.25 903.25 946.27 946.27 860.24 989.28 946.27 817.23 3103.29 2818.42 3103.29 2728.93 2983.54 2968.42 3103.29 3103.29 2848.67 3223.03 3088.16 2744.05 23.45 21.30 23.45 20.62 22.55 22.43 23.45 23.45 21.53 24.36 23.34 20.74 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 189.26 172.05 189.26 163.45 180.65 180.65 189.26 189.26 172.05 197.86 189.26 163.45 293.21 266.56 293.21 253.23 279.88 279.88 293.21 293.21 266.56 306.54 293.21 253.23 4.38 3.99 4.38 3.79 4.19 4.19 4.38 4.38 3.99 4.58 4.38 3.79 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 286.0 260.0 286.0 247.0 273.0 273.0 286.0 286.0 260.0 299.0 286.0 247.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 286.0 260.0 286.0 247.0 273.0 273.0 286.0 286.0 260.0 299.0 286.0 247.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 242.0 220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 1 1 0 1 21.4 -1 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1187 A1187 1 344 100.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 3277.73 2960.53 3277.73 3172.00 3277.73 3172.00 3277.73 3277.73 3172.00 3277.73 3172.00 3277.73 3277.73 2960.53 3277.73 3172.00 3277.73 3172.00 3277.73 3277.73 3172.00 3277.73 3172.00 3277.73 2246.85 2029.42 2246.85 2174.37 2246.85 2174.37 2246.85 2246.85 2174.37 2246.85 2174.37 2246.85 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 468.25 422.93 468.25 453.14 468.25 453.14 468.25 468.25 453.14 468.25 453.14 468.25 468.25 422.93 468.25 453.14 468.25 453.14 468.25 468.25 453.14 468.25 453.14 468.25 4960.00 4480.00 4960.00 4800.00 4960.00 4800.00 4960.00 4960.00 4800.00 4960.00 4800.00 4960.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 0 1 0 14.6 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1188 A1188 1 344 100.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 2625.70 2371.60 2625.70 2541.00 2625.70 2541.00 2625.70 2625.70 2541.00 2625.70 2541.00 2625.70 2625.70 2371.60 2625.70 2541.00 2625.70 2541.00 2625.70 2625.70 2541.00 2625.70 2541.00 2625.70 899.00 812.00 899.00 870.00 899.00 870.00 899.00 899.00 870.00 899.00 870.00 899.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 375.10 338.80 375.10 363.00 375.10 363.00 375.10 375.10 363.00 375.10 363.00 375.10 375.10 338.80 375.10 363.00 375.10 363.00 375.10 375.10 363.00 375.10 363.00 375.10 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0

20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 3 1 0 0 9.1 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1189 A1189 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 1600.86 1448.40 1600.86 1505.57 1581.81 1543.69 1600.86 1600.86 1524.63 1619.92 1562.75 1543.69 3107.56 2811.60 3107.56 2922.58 3070.56 2996.57 3107.56 3107.56 2959.58 3144.55 3033.57 2996.57 822.07 742.51 822.07 795.55 822.07 795.55 822.07 822.07 795.55 822.07 795.55 822.07 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 156.95 142.00 156.95 147.61 155.08 151.34 156.95 156.95 149.47 158.82 153.21 151.34 161.15 145.74 161.15 152.74 159.75 155.55 161.15 161.15 154.14 162.55 156.95 156.95 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 3.0 3.0 6.0 3.0 3.0 6.0 3.0 3.0 6.0 3.0 3.0 3.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 407.0 368.0 407.0 387.0 404.0 393.0 407.0 407.0 390.0 410.0 396.0 398.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 407.0 368.0 407.0 387.0 404.0 393.0 407.0 407.0 390.0 410.0 396.0 398.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 345.0 312.0 345.0 327.0 342.0 333.0 345.0 345.0 330.0 348.0 336.0 336.0 3 2 2 1 6.9 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1190 A1190 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 1230.45 1113.27 1230.45 1157.21 1215.81 1186.51 1230.45 1230.45 1171.86 1245.10 1201.16 1186.51 786.68 711.76 786.68 739.86 777.32 758.59 786.68 786.68 749.22 796.05 767.95 758.59 4865.62 4394.75 4865.62 4708.66 4865.62 4708.66 4865.62 4865.62 4708.66 4865.62 4708.66 4865.62 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 183.38 165.91 183.38 172.46 181.19 176.83 183.38 183.38 174.64 185.56 179.01 176.83 349.29 314.36 349.29 358.02 358.02 340.56 349.29 349.29 349.29 340.56 331.82 375.48 224.00 202.67 224.00 210.67 221.33 216.00 224.00 224.00 213.33 226.67 218.67 216.00 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 9.0 9.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 7.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 222.0 200.0 222.0 224.0 226.0 216.0 222.0 222.0 220.0 218.0 212.0 234.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 222.0 200.0 222.0 224.0 226.0 216.0 222.0 222.0 220.0 218.0 212.0 234.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 160.0 144.0 160.0 164.0 164.0 156.0 160.0 160.0 160.0 156.0 152.0 172.0 1 1 0 0 32.3 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1191 A1191 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 5017.04 4531.52 5017.04 4855.20 5017.04 4855.20 5017.04 5017.04 4855.20 5017.04 4855.20 5017.04 9738.96 8796.48 9738.96 9424.80 9738.96 9424.80 9738.96 9738.96 9424.80 9738.96 9424.80 9738.96 781.20 705.60 781.20 756.00 781.20 756.00 781.20 781.20 756.00 781.20 756.00 781.20 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 1475.60 1332.80 1475.60 1428.00 1475.60 1428.00 1475.60 1475.60 1428.00 1475.60 1428.00 1475.60 1475.60 1332.80 1475.60 1428.00 1475.60 1428.00 1475.60 1475.60 1428.00 1475.60 1428.00 1475.60 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 1 0 1 15.5 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1192 A1192 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 5332.05 4803.18 5332.05 5388.40 5431.76 5189.00 5332.05 5332.05 5288.70 5232.34 5089.29 5631.16 10350.45 9323.82 10350.45 10459.84 10544.00

```
10072.76 10350.45 10350.45 10266.30 10156.91 9879.21 10931.08 7185.21 6489.87 7185.21 6953.43
7185.21 6953.43 7185.21 7185.21 6953.43 7185.21 6953.43 7185.21 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 1568.25 1412.70 1568.25 1584.82 1597.57 1526.18
1568.25 1568.25 1555.50 1538.92 1496.85 1656.22 1989.00 1795.20 1989.00 1948.20 1999.20 1927.80
1989.00 1989.00 1938.00 1978.80 1917.60 2019.60 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 6.0 6.0 6.0 6.0
6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 6.0 6.0 6.0 6.0 6.0
6.0 6.0 6.0 6.0 6.0 6.0 6.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 4.0 4.0 4.0 4.0 4.0 4.0
4.0 4.0 4.0 4.0 4.0 4.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 518.0 468.0 518.0 499.0
517.0 501.0 518.0 518.0 500.0 519.0 502.0 515.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 518.0 468.0 518.0 499.0 517.0 501.0 518.0 518.0 500.0 519.0
502.0 515.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 390.0 352.0 390.0 382.0 392.0 378.0 390.0 390.0
380.0 388.0 376.0 396.0 1 1 0 1 33.4 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1193 A1193 1 344 500.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 690.65 624.87 690.65 649.54 682.43 665.98
690.65 690.65 657.76 698.87 674.21 665.98 1282.64 1160.48 1282.64 1206.29 1267.37 1236.83 1282.64
1282.64 1221.56 1297.91 1252.10 1236.83 8221.20 7425.60 8221.20 7956.00 8221.20 7956.00 8221.20
8221.20 7956.00 8221.20 7956.00 8221.20 3868.80 3494.40 3868.80 3744.00 3868.80 3744.00 3868.80
3868.80 3744.00 3868.80 3744.00 3868.80 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50
14.50 14.50 14.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 274.07 247.97 274.07 257.75
270.81 264.28 274.07 274.07 261.02 277.33 267.54 264.28 522.03 469.83 522.03 535.08 535.08 508.98
522.03 522.03 522.03 508.98 495.93 561.19 12.32 11.15 12.32 11.59 12.17 11.88 12.32 12.32 11.73 12.47
12.03 11.88 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 9.0 9.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0
9.0 7.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 14.0 14.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 14.0 222.0 200.0 222.0 224.0 226.0 216.0 222.0 222.0 220.0 218.0 212.0 234.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 222.0 200.0
222.0 224.0 226.0 216.0 222.0 222.0 220.0 218.0 212.0 234.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
160.0 144.0 160.0 164.0 164.0 156.0 160.0 160.0 160.0 156.0 152.0 172.0 1 1 0 0 38.7 0 31.0 28.0 31.0 30.0
31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1194 A1194 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 2630.32 2391.20 2630.32 2271.64 2510.76
2510.76 2630.32 2630.32 2391.20 2749.88 2630.32 2271.64 1681.68 1528.80 1681.68 1452.36 1605.24
1605.24 1681.68 1681.68 1528.80 1758.12 1681.68 1452.36 408.80 371.20 408.80 360.80 393.60 391.20
408.80 408.80 376.00 424.00 406.40 363.20 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 308.00 280.00 308.00 266.00 294.00 294.00 308.00 308.00 280.00 322.00
308.00 266.00 346.50 315.00 346.50 299.25 330.75 330.75 346.50 346.50 315.00 362.25 346.50 299.25 6.60
6.00 6.60 5.70 6.30 6.30 6.60 6.60 6.00 6.90 6.60 5.70 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0
13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 242.0 220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 242.0
220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 198.0 180.0 198.0 171.0 189.0 189.0 198.0 198.0 180.0 207.0 198.0 171.0 1 1 2 1 16.9 0 22.0 20.0 22.0
19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1195 A1195 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 3655.93 3302.13 3655.93 3538.00 3655.93
3538.00 3655.93 3655.93 3538.00 3655.93 3538.00 3655.93 2337.40 2111.20 2337.40 2262.00 2337.40
2262.00 2337.40 2337.40 2262.00 2337.40 2262.00 2337.40 923.35 833.99 923.35 893.57 923.35 893.57
923.35 923.35 893.57 923.35 893.57 923.35 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0
25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 599.33 541.33 599.33 580.00 599.33 580.00 599.33 599.33 580.00 599.33
580.00 599.33 740.35 668.71 740.35 716.47 740.35 716.47 740.35 740.35 716.47 740.35 716.47 740.35 6.20
5.60 6.20 6.00 6.20 6.00 6.20 6.20 6.00 6.20 6.00 6.20 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0
15.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0
22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0
```

```
16.0 16.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 279.0 252.0 279.0 270.0 279.0
270.0 279.0 279.0 270.0 279.0 270.0 279.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 22.0
22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0
279.0 270.0 279.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 217.0 196.0 217.0 210.0 217.0 210.0 217.0
217.0 210.0 217.0 210.0 217.0 1 0 2 1 22.6 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1196 A1196 1 344 200.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 9765.00 8820.00 9765.00 9450.00 9765.00
9450.00 9765.00 9765.00 9450.00 9765.00 9450.00 9765.00 4185.00 3780.00 4185.00 4050.00 4185.00
4050.00 4185.00 4185.00 4050.00 4185.00 4050.00 4185.00 545.60 492.80 545.60 528.00 545.60 528.00
545.60 545.60 528.00 545.60 528.00 545.60 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.00
12.00 12.00 12.00 12.00 12.00 12.00 12.00 12.00 12.00 12.00 12.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 558.00 504.00 558.00 540.00 558.00 540.00 558.00 558.00 540.00 558.00 540.00 558.00
627.75 567.00 627.75 607.50 627.75 607.50 627.75 627.75 607.50 627.75 607.50 627.75 3.72 3.36 3.72 3.60
3.72 3.60 3.72 3.72 3.60 3.72 3.60 3.72 720.0 672.0 744.0 336.0 480.0 720.0 744.0 624.0 720.0 744.0 720.0
576.0 720.0 672.0 744.0 48.0 72.0 720.0 744.0 96.0 720.0 744.0 720.0 120.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0
270.0 279.0 270.0 279.0 3 0 2 1 41.4 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1197 A1197 1 344 300.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 2148.30 1940.40 2148.30 2079.00 2148.30
2079.00 2148.30 2148.30 2079.00 2148.30 2079.00 2148.30 2148.30 1940.40 2148.30 2079.00 2148.30
2079.00 2148.30 2148.30 2079.00 2148.30 2079.00 2148.30 11558.47 10439.91 11558.47 11185.62 11558.47
11185.62 11558.47 11558.47 11185.62 11558.47 11185.62 11558.47 608.34 549.47 608.34 588.72 608.34
588.72 608.34 608.34 588.72 608.34 588.72 608.34 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0
27.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00
15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 286.44 258.72
286.44 277.20 286.44 277.20 286.44 286.44 277.20 286.44 277.20 286.44 286.44 258.72 286.44 277.20
286.44 277.20 286.44 286.44 277.20 286.44 277.20 286.44 12400.00 11200.00 12400.00 12000.00 12400.00
12000.00 12400.00 12400.00 12000.00 12400.00 12000.00 12400.00 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 310.0 280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0
310.0 300.0 310.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0
16.0 16.0 16.0 310.0 280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 248.0 224.0 248.0 240.0 248.0 240.0 248.0 248.0 240.0 248.0 240.0 248.0 1 1 0 0
50.5 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1198 A1198 1 344 750.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 737.80 666.40 737.80 714.00 737.80 714.00
737.80 737.80 714.00 737.80 714.00 737.80 1432.20 1293.60 1432.20 1386.00 1432.20 1386.00 1432.20
1432.20 1386.00 1432.20 1386.00 1432.20 455.56 411.47 455.56 440.87 455.56 440.87 455.56 455.56 440.87
455.56 440.87 455.56 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0
25.0 25.0 25.0 25.0 25.0 25.0 25.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00
17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 86.80 78.40 86.80 84.00 86.80 84.00 86.80 86.80 84.00 86.80 84.00 86.80 107.22 96.85 107.22
103.76 107.22 103.76 107.22 107.22 103.76 107.22 103.76 107.22 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 22.0 22.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0
270.0 279.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 217.0 196.0 217.0 210.0 217.0 210.0 217.0 217.0 210.0 217.0 210.0 217.0 1
0 0 0 6.5 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1199 A1199 1 344 200.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 29064.65 26251.94 29064.65 28127.08
29064.65 28127.08 29064.65 29064.65 28127.08 29064.65 28127.08 29064.65 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0
```

```
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 2 0 2 0 67.9 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1200 A1200 1 344 200.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 3813.00 3444.00 3813.00 3690.00 3813.00
3690.00 3813.00 3813.00 3690.00 3813.00 3690.00 3813.00 3813.00 3444.00 3813.00 3690.00 3813.00
3690.00 3813.00 3813.00 3690.00 3813.00 3690.00 3813.00 1734.37 1571.71 1734.37 1586.37 1694.18
1666.75 1734.37 1734.37 1626.56 1774.56 1706.95 1613.80 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 423.67 382.67 423.67 410.00 423.67 410.00 423.67 423.67
410.00 423.67 410.00 423.67 523.35 472.71 523.35 506.47 523.35 506.47 523.35 523.35 506.47 523.35
506.47 523.35 3.72 3.36 3.72 3.60 3.72 3.60 3.72 3.72 3.60 3.72 3.60 3.72 15.0 15.0 15.0 15.0 15.0 15.0 15.0
15.0 15.0 15.0 15.0 15.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0
16.0 16.0 16.0 16.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 16.0 16.0 16.0 16.0 16.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 279.0
252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 279.0 252.0 279.0 270.0 279.0
270.0 279.0 279.0 270.0 279.0 270.0 279.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 217.0 196.0 217.0
210.0 217.0 210.0 217.0 217.0 210.0 217.0 210.0 217.0 3 1 0 0 25.7 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0
30.0 31.0 30.0 31.0
 1 1201 A1201 1 344 200.00 9.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 2659.75 2402.36 2659.75 2573.96 2659.75
2573.96 2659.75 2659.75 2573.96 2659.75 2573.96 2659.75 1700.50 1535.93 1700.50 1645.64 1700.50
1645.64 1700.50 1700.50 1645.64 1700.50 1645.64 1700.50 2101.87 1898.46 2101.87 2034.07 2101.87
2034.07 2101.87 2101.87 2034.07 2101.87 2034.07 2101.87 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 311.45 281.31 311.45 301.40 311.45 301.40 311.45 311.45
301.40 311.45 301.40 311.45 350.38 316.47 350.38 339.07 350.38 339.07 350.38 350.38 339.07 350.38
339.07 350.38 1.24 1.12 1.24 1.20 1.24 1.20 1.24 1.24 1.20 1.24 1.20 1.24 13.0 13.0 13.0 13.0 13.0 13.0 13.0
13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0
341.0 330.0 341.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 1 1 0 0
16.1 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1202 A1202 1 344 600.00 10.00 0 1 0 1 2 0 0 2 2 -1 -1 -1 1575.42 1422.96 1575.42 1524.60 1575.42
1524.60 1575.42 1575.42 1524.60 1575.42 1524.60 1575.42 1575.42 1422.96 1575.42 1524.60 1575.42
1524.60 1575.42 1575.42 1524.60 1575.42 1524.60 1575.42 1658.50 1498.00 1658.50 1605.00 1658.50
1605.00 1658.50 1658.50 1605.00 1658.50 1605.00 1658.50 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 225.06 203.28 225.06 217.80 225.06 217.80 225.06 225.06
217.80 225.06 217.80 225.06 348.68 314.94 348.68 337.44 348.68 337.44 348.68 348.68 337.44 348.68
337.44 348.68 1.24 1.12 1.24 1.20 1.24 1.20 1.24 1.24 1.20 1.24 1.20 1.24 11.0 11.0 11.0 11.0 11.0 11.0 11.0
11.0 11.0 11.0 11.0 11.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 403.0 364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0
403.0 390.0 403.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 403.0 364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0 403.0 390.0 403.0 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 1 1 2 1
```

12.2 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
1 1203 A1203 1 344 50.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 3103.10 2802.80 3103.10 3003.00 3103.10 3003.00 3103.10 3103.10 3003.00 3103.10 3003.00 3103.10 3103.10 2802.80 3103.10 3003.00 3103.10 3003.00 3103.10 3103.10 3003.00 3103.10 3003.00 3103.10 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 443.30 400.40 443.30 429.00 443.30 429.00 443.30 443.30 429.00 443.30 429.00 443.30 443.30 400.40 443.30 429.00 443.30 429.00 443.30 443.30 429.00 443.30 429.00 443.30 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 465.0 420.0 465.0 450.0 465.0 450.0 465.0 465.0 450.0 465.0 450.0 465.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 465.0 420.0 465.0 450.0 465.0 450.0 465.0 465.0 450.0 465.0 450.0 465.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 403.0 364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0 403.0 390.0 403.0 3 1 2 0 7.7 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
1 1204 A1204 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 3379.51 3042.45 3397.25 3237.59 3335.16 3237.59 3361.77 3574.65 3228.72 3361.77 3193.24 3379.51 6560.23 5905.92 6594.66 6284.73 6474.13 6284.73 6525.79 6939.03 6267.51 6525.79 6198.64 6560.23 1488.00 1344.00 1488.00 1440.00 1488.00 1440.00 1488.00 1488.00 1440.00 1488.00 1440.00 1488.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 931.85 838.91 936.74 892.72 919.62 892.72 926.96 985.66 890.27 926.96 880.49 931.85 931.85 838.91 936.74 892.72 919.62 892.72 926.96 985.66 890.27 926.96 880.49 931.85 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 11.0 11.0 10.0 11.0 10.0 11.0 10.0 9.0 11.0 11.0 10.0 10.0 3.0 3.0 4.0 3.0 3.0 3.0 4.0 4.0 3.0 3.0 3.0 4.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 7.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 19.0 20.0 20.0 20.0 20.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 7.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 19.0 20.0 20.0 20.0 20.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 5.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 19.0 20.0 20.0 20.0 20.0 443.0 399.0 445.0 425.0 438.0 425.0 441.0 465.0 424.0 441.0 420.0 443.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 5.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 19.0 20.0 20.0 20.0 20.0 443.0 399.0 445.0 425.0 438.0 425.0 441.0 465.0 424.0 441.0 420.0 443.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 381.0 343.0 383.0 365.0 376.0 365.0 379.0 403.0 364.0 379.0 360.0 381.0 1 1 2 0 10.9 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
1 1205 A1205 1 344 200.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 2625.70 2371.60 2625.70 2541.00 2625.70 2541.00 2625.70 2625.70 2541.00 2625.70 2541.00 2625.70 2625.70 2371.60 2625.70 2541.00 2625.70 2541.00 2625.70 2625.70 2541.00 2625.70 2541.00 2625.70 2247.50 2030.00 2247.50 2175.00 2247.50 2175.00 2247.50 2247.50 2175.00 2247.50 2175.00 2247.50 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 450.12 406.56 450.12 435.60 450.12 435.60 450.12 450.12 435.60 450.12 435.60 450.12 450.12 406.56 450.12 435.60 450.12 435.60 450.12 450.12 435.60 450.12 435.60 450.12 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 1 1 0 12.1 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
1 1206 A1206 1 344 500.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 573.80 521.64 573.80 495.56 547.72 547.72 573.80 573.80 521.64 599.89 573.80 495.56 1551.40 1410.36 1551.40 1339.84 1480.88 1480.88 1551.40 1551.40 1410.36 1621.91 1551.40 1339.84 1231.00 1118.00 1231.00 1082.50 1183.50 1177.50 1231.00 1231.00 1130.00 1278.50 1225.00 1088.50 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 118.07 107.33 118.07 101.97 112.70 112.70 118.07 118.07 107.33 123.43 118.07 101.97 132.83 120.75 132.83 114.71 126.79 126.79 132.83 132.83 120.75 138.86 132.83 114.71 6.16 5.60 6.16 5.32 5.88 5.88 6.16 6.16 5.60 6.44 6.16 5.32 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0

17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 242.0 220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 242.0 220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 198.0 180.0 198.0 171.0 189.0 189.0 198.0 198.0 180.0 207.0 198.0 171.0 1 1 2 1 8.3 -1 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

1 1207 A1207 1 344 400.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 1871.40 1690.30 1871.40 1811.03 1871.40 1811.03 1871.40 1871.40 1811.03 1871.40 1811.03 1871.40 1281.76 1157.72 1281.76 1240.41 1281.76 1240.41 1281.76 1281.76 1240.41 1281.76 1240.41 1281.76 6036.25 5452.10 6036.25 5841.53 6036.25 5841.53 6036.25 6036.25 5841.53 6036.25 5841.53 6036.25 45.61 41.20 45.61 44.14 45.61 44.14 45.61 45.61 44.14 45.61 44.14 45.61 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 256.35 231.55 256.35 248.08 256.35 248.08 256.35 256.35 248.08 256.35 248.08 256.35 397.17 358.73 397.17 384.36 397.17 384.36 397.17 397.17 384.36 397.17 384.36 397.17 6.18 5.58 6.18 5.98 6.18 5.98 6.18 6.18 5.98 6.18 5.98 6.18 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 403.0 364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0 403.0 390.0 403.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 403.0 364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0 403.0 390.0 403.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 1 1 0 1 22.0 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1208 A1208 1 344 100.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 2015.05 1826.01 1993.27 1966.55 2047.71 1977.44 2025.93 1840.86 1977.44 2036.82 2020.98 2004.16 2015.05 1826.01 1993.27 1966.55 2047.71 1977.44 2025.93 1840.86 1977.44 2036.82 2020.98 2004.16 1713.17 1557.17 1657.97 1679.57 1777.57 1716.37 1731.57 1326.76 1707.17 1768.37 1817.57 1676.37 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 287.86 260.86 284.75 280.94 292.53 282.49 289.42 262.98 282.49 290.97 288.71 286.31 397.30 361.95 385.99 393.06 414.26 398.71 402.95 306.81 398.71 408.61 421.33 391.64 1720.56 1559.15 1701.97 1679.15 1748.45 1688.45 1729.86 1571.83 1688.45 1739.15 1725.63 1711.27 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 6.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 15.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 15.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 15.0 17.0 17.0 17.0 17.0 343.0 312.0 335.0 338.0 355.0 342.0 347.0 279.0 342.0 351.0 358.0 339.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 15.0 17.0 17.0 17.0 17.0 343.0 312.0 335.0 338.0 355.0 342.0 347.0 279.0 342.0 351.0 358.0 339.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 281.0 256.0 273.0 278.0 293.0 282.0 285.0 217.0 282.0 289.0 298.0 277.0 1 0 1 0 14.7 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1209 A1209 1 344 100.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 1108.80 1039.50 970.20 1178.10 1316.70 1247.40 1178.10 0.00 1247.40 1247.40 1524.60 1039.50 1108.80 1039.50 970.20 1178.10 1316.70 1247.40 1178.10 0.00 1247.40 1247.40 1524.60 1039.50 500.00 466.20 446.80 524.20 579.80 550.80 526.60 74.40 550.80 553.20 657.20 473.40 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 158.40 148.50 138.60 168.30 188.10 178.20 168.30 0.00 178.20 178.20 217.80 148.50 211.20 198.00 184.80 224.40 250.80 237.60 224.40 0.00 237.60 237.60 290.40 198.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 0.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 224.0 210.0 196.0 238.0 266.0 252.0 238.0 0.0 252.0 252.0 308.0 210.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 224.0 210.0 196.0 238.0 266.0 252.0 238.0 0.0 252.0 252.0 308.0 210.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 192.0 180.0 168.0 204.0 228.0 216.0 204.0 0.0 216.0 216.0 264.0 180.0 3 1 0 0 9.7 0 16.0 15.0 14.0 17.0 19.0 18.0 17.0 0.0 18.0 18.0 22.0 15.0

1 1210 A1210 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 1258.42 1145.49 1226.16 1242.29 1306.83 1258.42 1274.56 1000.29 1258.42 1290.69 1322.96 1242.29 2442.83 2223.60 2380.19 2411.51 2536.78

2442.83 2474.14 1941.73 2442.83 2505.46 2568.10 2411.51 685.27 622.87 663.19 671.83 711.03 686.55 692.63 530.70 682.87 707.35 727.03 670.55 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 123.38 112.30 120.21 121.79 128.12 123.38 124.96 98.07 123.38 126.54 129.70 121.79 156.59 142.88 151.32 155.54 164.50 158.17 159.23 114.41 158.17 161.86 168.72 153.96 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 18.0 18.0 18.0 18.0 359.0 327.0 349.0 355.0 374.0 360.0 364.0 279.0 360.0 369.0 380.0 354.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 18.0 18.0 18.0 18.0 359.0 327.0 349.0 355.0 374.0 360.0 364.0 279.0 360.0 369.0 380.0 354.0 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 1 2 297.0 271.0 287.0 295.0 312.0 300.0 302.0 217.0 300.0 307.0 320.0 292.0 3 2 2 1 7.3 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1211 A1211 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 644.16 603.90 563.64 684.42 764.94 724.68 684.42 0.00 724.68 724.68 885.72 603.90 411.84 386.10 360.36 437.58 489.06 463.32 437.58 0.00 463.32 463.32 566.28 386.10 4392.36 4092.34 3936.33 4596.38 5076.42 4824.40 4620.38 744.06 4824.40 4848.40 5736.48 4164.35 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 96.00 90.00 84.00 102.00 114.00 108.00 102.00 0.00 108.00 108.00 132.00 90.00 128.00 120.00 112.00 136.00 152.00 144.00 136.00 0.00 144.00 144.00 176.00 120.00 96.00 90.00 84.00 102.00 114.00 108.00 102.00 0.00 108.00 108.00 132.00 90.00 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 0.0 9.0 9.0 9.0 9.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 0.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 0.0 14.0 14.0 14.0 14.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 0.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 0.0 14.0 14.0 14.0 14.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 0.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 0.0 14.0 14.0 14.0 14.0 96.0 90.0 84.0 102.0 114.0 108.0 102.0 0.0 108.0 108.0 132.0 90.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 0.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 0.0 14.0 14.0 14.0 14.0 96.0 90.0 84.0 102.0 114.0 108.0 102.0 0.0 108.0 108.0 132.0 90.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 64.0 60.0 56.0 68.0 76.0 72.0 68.0 0.0 72.0 72.0 88.0 60.0 1 1 0 0 32.0 0 16.0 15.0 14.0 17.0 19.0 18.0 17.0 0.0 18.0 18.0 22.0 15.0

1 1212 A1212 1 344 500.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 548.86 514.55 480.25 583.16 651.77 617.46 583.16 0.00 617.46 617.46 754.68 514.55 1019.30 955.60 891.89 1083.01 1210.42 1146.72 1083.01 0.00 1146.72 1146.72 1401.54 955.60 5146.24 4789.92 4629.44 5372.00 5921.44 5630.40 5404.64 1011.84 5630.40 5663.04 6664.00 4887.84 2421.76 2254.08 2178.56 2528.00 2786.56 2649.60 2543.36 476.16 2649.60 2664.96 3136.00 2300.16 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 217.80 204.19 190.58 231.41 258.64 245.02 231.41 0.00 245.02 245.02 299.47 204.19 290.40 272.25 254.10 308.55 344.85 326.70 308.55 0.00 326.70 326.70 399.30 272.25 5.28 4.95 4.62 5.61 6.27 5.94 5.61 0.00 5.94 5.94 7.26 4.95 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 0.0 9.0 9.0 9.0 9.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 0.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 0.0 14.0 14.0 14.0 14.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 0.0 11.0 11.0 11.0 11.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 0.0 14.0 14.0 14.0 14.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 0.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 0.0 14.0 14.0 14.0 14.0 96.0 90.0 84.0 102.0 114.0 108.0 102.0 0.0 108.0 108.0 132.0 90.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 0.0 9.0 9.0 9.0 9.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 0.0 14.0 14.0 14.0 14.0 96.0 90.0 84.0 102.0 114.0 108.0 102.0 0.0 108.0 108.0 132.0 90.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 64.0 60.0 56.0 68.0 76.0 72.0 68.0 0.0 72.0 72.0 88.0 60.0 1 1 0 0 45.1 0 16.0 15.0 14.0 17.0 19.0 18.0 17.0 0.0 18.0 18.0 22.0 15.0

1 1213 A1213 1 344 300.00 0.00 1 1 0 0 2 0 0 2 2 -1 -1 -1 2117.92 1985.55 1853.18 2250.29 2515.03 2382.66 2250.29 0.00 2382.66 2382.66 2912.14 1985.55 1354.08 1269.45 1184.82 1438.71 1607.97 1523.34 1438.71 0.00 1523.34 1523.34 1861.86 1269.45 500.00 466.20 446.80 524.20 579.80 550.80 526.60 74.40 550.80 553.20 657.20 473.40 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 248.00 232.50 217.00 263.50 294.50 279.00 263.50 0.00 279.00 279.00 341.00 232.50 384.23 360.21 336.20 408.24 456.27 432.25 408.24 0.00 432.25 432.25 528.31 360.21 2.40 2.25 2.10 2.55 2.85 2.70 2.55 0.00 2.70 2.70 3.30 2.25 11.0 11.0 11.0 11.0 11.0 11.0 11.0 0.0 11.0 11.0 11.0 11.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0

```
17.0 17.0 17.0 17.0 208.0 195.0 182.0 221.0 247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0 5.0 5.0 5.0 5.0 5.0
5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 208.0 195.0 182.0 221.0
247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 176.0 165.0
154.0 187.0 209.0 198.0 187.0 0.0 198.0 198.0 242.0 165.0 1 1 2 1 20.6 0 16.0 15.0 14.0 17.0 19.0 18.0 17.0
0.0 18.0 18.0 22.0 15.0
 1 1214 A1214 1 344 300.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 1815.36 1701.90 1588.44 1928.82 2155.74
2042.28 1928.82 0.00 2042.28 2042.28 2496.12 1701.90 1160.64 1088.10 1015.56 1233.18 1378.26 1305.72
1233.18 0.00 1305.72 1305.72 1595.88 1088.10 7500.00 6993.00 6702.00 7863.00 8697.00 8262.00 7899.00
1116.00 8262.00 8298.00 9858.00 7101.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 24.0
24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 248.00 232.50 217.00 263.50 294.50 279.00 263.50 0.00 279.00 279.00 341.00
232.50 384.23 360.21 336.20 408.24 456.27 432.25 408.24 0.00 432.25 432.25 528.31 360.21 9.60 9.00 8.40
10.20 11.40 10.80 10.20 0.00 10.80 10.80 13.20 9.00 11.0 11.0 11.0 11.0 11.0 11.0 11.0 0.0 11.0 11.0 11.0
11.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 0.0 17.0 17.0 17.0 17.0 208.0 195.0 182.0 221.0 247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0 5.0
5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 208.0
195.0 182.0 221.0 247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 176.0 165.0 154.0 187.0 209.0 198.0 187.0 0.0 198.0 198.0 242.0 165.0 1 1 0 1 46.0 -1 16.0 15.0 14.0 17.0
19.0 18.0 17.0 0.0 18.0 18.0 22.0 15.0
 1 1215 A1215 1 344 800.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 2326.78 2181.36 2035.94 2472.21 2763.06
2617.63 2472.21 0.00 2617.63 2617.63 3199.33 2181.36 1487.62 1394.64 1301.66 1580.59 1766.54 1673.57
1580.59 0.00 1673.57 1673.57 2045.47 1394.64 1982.68 1846.39 1779.95 2072.40 2286.77 2173.77 2084.04
360.84 2173.77 2185.41 2579.23 1881.31 61.32 57.10 55.05 64.09 70.72 67.23 64.45 11.16 67.23 67.59 79.77
58.18 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 286.08 268.20 250.32 303.96 339.72 321.84 303.96 0.00 321.84
321.84 393.36 268.20 443.22 415.52 387.82 470.92 526.33 498.63 470.92 0.00 498.63 498.63 609.43 415.52
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 11.0 11.0 11.0 11.0 11.0 11.0 11.0 0.0 11.0 11.0
11.0 11.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 208.0 195.0 182.0 221.0 247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0
5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 208.0
195.0 182.0 221.0 247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 176.0 165.0 154.0 187.0 209.0 198.0 187.0 0.0 198.0 198.0 242.0 165.0 1 1 0 1 31.1 -1 16.0 15.0 14.0 17.0
19.0 18.0 17.0 0.0 18.0 18.0 22.0 15.0
 1 1216 A1216 1 344 200.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 21915.77 19875.80 21459.77 21387.78
22447.78 21691.79 22067.77 18723.73 21615.79 22371.78 22527.80 21611.77 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 2 0 2 0 60.4 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1217 A1217 1 344 200.00 9.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 1262.55 1183.64 1104.73 1341.46 1499.28
1420.37 1341.46 0.00 1420.37 1420.37 1736.01 1183.64 807.21 756.76 706.31 857.66 958.56 908.11 857.66
0.00 908.11 908.11 1109.91 756.76 869.42 808.71 783.97 906.14 997.58 948.86 912.14 185.88 948.86 954.86
1119.75 826.70 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00
16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
147.84 138.60 129.36 157.08 175.56 166.32 157.08 0.00 166.32 166.32 203.28 138.60 147.84 138.60 129.36
157.08 175.56 166.32 157.08 0.00 166.32 166.32 203.28 138.60 1.92 1.80 1.68 2.04 2.28 2.16 2.04 0.00 2.16
```

2.16 2.64 1.80 14.0 14.0 14.0 14.0 14.0 14.0 14.0 0.0 14.0 14.0 14.0 14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 0.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 0.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 0.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 0.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 0.0 16.0 16.0 16.0 16.0 160.0 150.0 140.0 170.0 190.0 180.0 170.0 0.0 180.0 180.0 220.0 150.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 0.0 16.0 16.0 16.0 16.0 160.0 150.0 140.0 170.0 190.0 180.0 170.0 0.0 180.0 180.0 220.0 150.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 128.0 120.0 112.0 136.0 152.0 144.0 136.0 0.0 144.0 144.0 176.0 120.0 1 1 0 0 15.5 0 16.0 15.0 14.0 17.0 19.0 18.0 17.0 0.0 18.0 18.0 22.0 15.0

1 1218 A1218 1 344 50.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 1355.20 1270.50 1185.80 1439.90 1609.30 1524.60 1439.90 0.00 1524.60 1524.60 1863.40 1270.50 1355.20 1270.50 1185.80 1439.90 1609.30 1524.60 1439.90 0.00 1524.60 1524.60 1863.40 1270.50 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 193.60 181.50 169.40 205.70 229.90 217.80 205.70 0.00 217.80 217.80 266.20 181.50 193.60 181.50 169.40 205.70 229.90 217.80 205.70 0.00 217.80 217.80 266.20 181.50 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 0.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 0.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 0.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 192.0 180.0 168.0 204.0 228.0 216.0 204.0 0.0 216.0 216.0 264.0 180.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 192.0 180.0 168.0 204.0 228.0 216.0 204.0 0.0 216.0 216.0 264.0 180.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 160.0 150.0 140.0 170.0 190.0 180.0 170.0 0.0 180.0 180.0 220.0 150.0 3 0 2 0 8.5 0 16.0 15.0 14.0 17.0 19.0 18.0 17.0 0.0 18.0 18.0 22.0 15.0

1 1219 A1219 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 2236.38 2046.05 2141.21 2236.38 2379.12 2283.96 2283.96 1475.06 2283.96 2331.54 2474.29 2188.79 4341.20 3971.74 4156.47 4341.20 4618.30 4433.56 4433.56 2863.34 4433.56 4525.93 4803.03 4248.83 1488.00 1344.00 1488.00 1440.00 1488.00 1440.00 1488.00 1488.00 1440.00 1488.00 1440.00 1488.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 616.65 564.17 590.41 616.65 656.01 629.77 629.77 406.72 629.77 642.89 682.25 603.53 649.45 592.59 627.58 645.07 682.25 656.01 660.38 474.51 656.01 671.31 699.74 638.51 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 3.0 3.0 7.0 3.0 7.0 7.0 7.0 7.0 7.0 3.0 3.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 15.0 20.0 20.0 20.0 20.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 15.0 20.0 20.0 20.0 20.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 15.0 20.0 20.0 20.0 20.0 359.0 327.0 349.0 355.0 374.0 360.0 364.0 279.0 360.0 369.0 380.0 354.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 15.0 20.0 20.0 20.0 20.0 359.0 327.0 349.0 355.0 374.0 360.0 364.0 279.0 360.0 369.0 380.0 354.0 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 1 2 297.0 271.0 287.0 295.0 312.0 300.0 302.0 217.0 300.0 307.0 320.0 292.0 1 1 2 0 9.9 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1220 A1220 1 344 200.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 1108.80 1039.50 970.20 1178.10 1316.70 1247.40 1178.10 0.00 1247.40 1247.40 1524.60 1039.50 1108.80 1039.50 970.20 1178.10 1316.70 1247.40 1178.10 0.00 1247.40 1247.40 1524.60 1039.50 1250.00 1165.50 1117.00 1310.50 1449.50 1377.00 1316.50 186.00 1377.00 1383.00 1643.00 1183.50 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 190.08 178.20 166.32 201.96 225.72 213.84 201.96 0.00 213.84 213.84 261.36 178.20 253.44 237.60 221.76 269.28 300.96 285.12 269.28 0.00 285.12 285.12 348.48 237.60 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 0.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 224.0 210.0 196.0 238.0 266.0 252.0 238.0 0.0 252.0 252.0 308.0 210.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 224.0 210.0 196.0 238.0 266.0 252.0 238.0 0.0 252.0 252.0 308.0 210.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 192.0 180.0 168.0 204.0 228.0 216.0 204.0 0.0 216.0 216.0 264.0 180.0 1 1 1 0 12.7 0 16.0 15.0 14.0 17.0 19.0 18.0 17.0 0.0 18.0 18.0 22.0 15.0

1 1221 A1221 1 344 500.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 623.08 584.14 545.19 662.02 739.91 700.96 662.02 0.00 700.96 700.96 856.73 584.14 1016.60 953.06 889.53 1080.14 1207.21 1143.68 1080.14 0.00 1143.68 1143.68 1397.83 953.06 946.38 880.86 851.34 987.90 1088.94 1035.42 993.90 186.08 1035.42

```
1041.42 1225.49 898.86 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 27.0 27.0 27.0 27.0 27.0
27.0 27.0 27.0 27.0 27.0 27.0 27.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00
15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 91.09 85.40 79.71 96.79 108.17 102.48 96.79 0.00 102.48 102.48 125.25 85.40 121.46 113.87
106.28 129.05 144.23 136.64 129.05 0.00 136.64 136.64 167.00 113.87 4.48 4.20 3.92 4.76 5.32 5.04 4.76
0.00 5.04 5.04 6.16 4.20 10.0 10.0 10.0 10.0 10.0 10.0 10.0 0.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0
0.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0
18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0
18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0
18.0 224.0 210.0 196.0 238.0 266.0 252.0 238.0 0.0 252.0 252.0 308.0 210.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0
5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 224.0 210.0 196.0 238.0 266.0 252.0
238.0 0.0 252.0 252.0 308.0 210.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 192.0 180.0 168.0 204.0
228.0 216.0 204.0 0.0 216.0 216.0 264.0 180.0 1 1 2 1 10.5 -1 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1222 A1222 1 344 280.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 4392.08 4117.58 3843.07 4666.58 5215.60
4941.09 4666.58 0.00 4941.09 4941.09 6039.11 4117.58 4392.08 4117.58 3843.07 4666.58 5215.60 4941.09
4666.58 0.00 4941.09 4941.09 6039.11 4117.58 1175.00 1095.57 1049.98 1231.87 1362.53 1294.38 1237.51
174.84 1294.38 1300.02 1544.42 1112.49 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 345.09 323.52 301.96 366.66 409.80 388.23 366.66 0.00 388.23 388.23 474.50
323.52 534.65 501.23 467.82 568.07 634.90 601.48 568.07 0.00 601.48 601.48 735.14 501.23 19.17 17.97
16.77 20.37 22.76 21.57 20.37 0.00 21.57 21.57 26.36 17.97 11.0 11.0 11.0 11.0 11.0 11.0 11.0 0.0 11.0 11.0
11.0 11.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 208.0 195.0 182.0 221.0 247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0
5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 208.0
195.0 182.0 221.0 247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 176.0 165.0 154.0 187.0 209.0 198.0 187.0 0.0 198.0 198.0 242.0 165.0 1 1 0 1 43.4 -1 16.0 15.0 14.0 17.0
19.0 18.0 17.0 0.0 18.0 18.0 22.0 15.0
 1 1223 A1223 1 344 400.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 854.16 800.78 747.39 907.55 1014.32 960.93
907.55 0.00 960.93 960.93 1174.48 800.78 585.03 548.47 511.90 621.60 694.73 658.16 621.60 0.00 658.16
658.16 804.42 548.47 2373.27 2210.13 2130.60 2480.67 2737.27 2602.00 2494.60 431.93 2602.00 2615.93
3087.34 2251.93 17.93 16.70 16.10 18.75 20.68 19.66 18.85 3.26 19.66 19.77 23.33 17.02 24.0 24.0 24.0 24.0
24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 16.50
16.50 16.50 16.50 16.50 16.50 16.50 16.50 16.50 16.50 16.50 16.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 117.01 109.70 102.38 124.32 138.95 131.63 124.32 0.00 131.63 131.63 160.89 109.70
181.28 169.95 158.62 192.61 215.27 203.94 192.61 0.00 203.94 203.94 249.26 169.95 4.26 3.99 3.73 4.53
5.06 4.79 4.53 0.00 4.79 4.79 5.86 3.99 11.0 11.0 11.0 11.0 11.0 11.0 11.0 0.0 11.0 11.0 11.0 11.0 5.0 5.0 5.0
5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 0.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
0.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0
17.0 17.0 17.0 17.0 208.0 195.0 182.0 221.0 247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0 5.0 5.0 5.0 5.0 5.0
5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 208.0 195.0 182.0 221.0
247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 176.0 165.0
154.0 187.0 209.0 198.0 187.0 0.0 198.0 198.0 242.0 165.0 1 1 0 1 20.2 -1 16.0 15.0 14.0 17.0 19.0 18.0 17.0
0.0 18.0 18.0 22.0 15.0
 1 1224 A1224 1 344 100.00 0.00 0 0 0 0 2 0 0 2 3 -1 -1 -1 3141.16 2837.18 3141.16 3039.83 3141.16
3039.83 3141.16 3141.16 3039.83 3141.16 3039.83 3141.16 2008.28 1813.93 2008.28 1943.50 2008.28
1943.50 2008.28 2008.28 1943.50 2008.28 1943.50 2008.28 857.18 774.23 857.18 829.53 857.18 829.53
857.18 857.18 829.53 857.18 829.53 857.18 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 514.94 465.11 514.94 498.33 514.94 498.33 514.94 514.94 498.33 514.94
498.33 514.94 647.36 584.71 647.36 626.48 647.36 626.48 647.36 647.36 626.48 647.36 626.48 647.36
142.49 128.70 142.49 137.89 142.49 137.89 142.49 142.49 137.89 142.49 137.89 142.49 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0
720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
```

23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 1 1 0 1 15.2 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
1 1225 A1225 1 344 50.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 2113.07 1909.60 2113.07 2026.75 2105.14 2042.60 2113.07 2113.07 2034.68 2120.99 2050.53 2089.29 2113.07 1909.60 2113.07 2026.75 2105.14 2042.60 2113.07 2113.07 2034.68 2120.99 2050.53 2089.29 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 301.87 272.80 301.87 289.54 300.73 291.80 301.87 301.87 290.67 303.00 292.93 298.47 346.03 312.06 346.03 343.52 349.81 335.97 346.03 346.03 339.74 342.26 332.19 357.36 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 7.0 7.0 6.0 5.0 6.0 6.0 5.0 6.0 5.0 7.0 7.0 5.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 337.0 304.0 337.0 333.0 340.0 327.0 337.0 337.0 330.0 334.0 324.0 346.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 337.0 304.0 337.0 333.0 340.0 327.0 337.0 337.0 330.0 334.0 324.0 346.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 275.0 248.0 275.0 273.0 278.0 267.0 275.0 275.0 270.0 272.0 264.0 284.0 3 0 2 0 8.8 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
1 1226 A1226 1 344 100.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 1908.58 1724.80 1908.58 1830.61 1901.42 1844.93 1908.58 1908.58 1837.77 1915.74 1852.09 1887.10 1908.58 1724.80 1908.58 1830.61 1901.42 1844.93 1908.58 1908.58 1837.77 1915.74 1852.09 1887.10 530.21 479.12 530.21 509.19 528.50 512.61 530.21 530.21 510.90 531.92 514.32 525.08 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 272.65 246.40 272.65 261.52 271.63 263.56 272.65 272.65 262.54 273.68 264.58 269.59 312.55 281.86 312.55 310.27 315.96 303.45 312.55 312.55 306.86 309.14 300.04 322.77 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 7.0 7.0 6.0 5.0 6.0 6.0 5.0 6.0 5.0 7.0 7.0 5.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 337.0 304.0 337.0 333.0 340.0 327.0 337.0 337.0 330.0 334.0 324.0 346.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 337.0 304.0 337.0 333.0 340.0 327.0 337.0 337.0 330.0 334.0 324.0 346.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 275.0 248.0 275.0 273.0 278.0 267.0 275.0 275.0 270.0 272.0 264.0 284.0 3 1 0 0 10.0 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
1 1227 A1227 1 344 200.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 1908.58 1724.80 1908.58 1830.61 1901.42 1844.93 1908.58 1908.58 1837.77 1915.74 1852.09 1887.10 1908.58 1724.80 1908.58 1830.61 1901.42 1844.93 1908.58 1908.58 1837.77 1915.74 1852.09 1887.10 1325.53 1197.80 1325.53 1272.98 1321.25 1281.53 1325.53 1325.53 1277.25 1329.80 1285.80 1312.70 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 327.18 295.68 327.18 313.82 325.96 316.27 327.18 327.18 315.05 328.41 317.50 323.50 375.06 338.23 375.06 372.33 379.15 364.14 375.06 375.06 368.24 370.96 360.05 387.33 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 7.0 7.0 6.0 5.0 6.0 6.0 5.0 6.0 5.0 7.0 7.0 5.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 337.0 304.0 337.0 333.0 340.0 327.0 337.0 337.0 330.0 334.0 324.0 346.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 337.0 304.0 337.0 333.0 340.0 327.0 337.0 337.0 330.0 334.0 324.0 346.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 275.0 248.0 275.0 273.0 278.0 267.0 275.0 275.0 270.0 272.0 264.0 284.0 1 1 1 0 13.0 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
1 1228 A1228 1 344 200.00 9.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 1818.68 1653.34 1818.68 1570.68 1736.01 1736.01 1818.68 1818.68 1653.34 1901.35 1818.68 1570.68 1162.76 1057.06 1162.76 1004.20 1109.91 1109.91 1162.76 1162.76 1057.06 1215.61 1162.76 1004.20 1231.50 1118.45 1231.50 1082.94 1183.98 1177.98 1231.50 1231.50 1130.46 1279.02 1225.49 1088.94 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 212.96 193.60 212.96 183.92 203.28 203.28 212.96 212.96

```
193.60 222.64 212.96 183.92 212.96 193.60 212.96 183.92 203.28 203.28 212.96 212.96 193.60 222.64
212.96 183.92 2.64 2.40 2.64 2.28 2.52 2.52 2.64 2.64 2.40 2.76 2.64 2.28 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 14.0 14.0 14.0 14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0
230.0 220.0 190.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0
16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 176.0 160.0 176.0 152.0 168.0 168.0 176.0 176.0 160.0 184.0 176.0 152.0 1 1 0 0
16.5 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1229 A1229 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 1264.80 1142.40 1264.80 1224.00 1264.80
1224.00 1264.80 1264.80 1224.00 1264.80 1224.00 1264.80 2455.20 2217.60 2455.20 2376.00 2455.20
2376.00 2455.20 2455.20 2376.00 2455.20 2376.00 2455.20 545.60 492.80 545.60 528.00 545.60 528.00
545.60 545.60 528.00 545.60 528.00 545.60 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0
25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0
16.0 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 124.00 112.00 124.00 120.00 124.00 120.00 124.00 124.00 120.00 124.00
120.00 124.00 124.00 112.00 124.00 120.00 124.00 120.00 124.00 124.00 120.00 124.00 120.00 124.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0
14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 310.0 280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 310.0
280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 248.0 224.0 248.0 240.0 248.0 240.0 248.0 248.0 240.0 248.0 240.0 248.0 3 2 2 1 7.1 -1 31.0 28.0 31.0
30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1230 A1230 1 344 100.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 2256.80 2038.40 2256.80 2184.00 2256.80
2184.00 2256.80 2256.80 2184.00 2256.80 2184.00 2256.80 2256.80 2038.40 2256.80 2184.00 2256.80
2184.00 2256.80 2256.80 2184.00 2256.80 2184.00 2256.80 1364.00 1232.00 1364.00 1320.00 1364.00
1320.00 1364.00 1364.00 1320.00 1364.00 1320.00 1364.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 322.40 291.20 322.40 312.00 322.40 312.00 322.40 322.40
312.00 322.40 312.00 322.40 322.40 291.20 322.40 312.00 322.40 312.00 322.40 322.40 312.00 322.40
312.00 322.40 3720.00 3360.00 3720.00 3600.00 3720.00 3600.00 3720.00 3720.00 3600.00 3720.00 3600.00
3720.00 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 310.0
280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 310.0 280.0 310.0 300.0 310.0 300.0 310.0
310.0 300.0 310.0 300.0 310.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 248.0 224.0 248.0 240.0 248.0
240.0 248.0 248.0 240.0 248.0 240.0 248.0 1 0 1 0 14.1 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0
30.0 31.0
 1 1231 A1231 1 344 300.00 0.00 1 1 0 0 2 0 0 2 2 -1 -1 -1 2442.44 2220.40 2442.44 2109.38 2331.42
2331.42 2442.44 2442.44 2220.40 2553.46 2442.44 2109.38 1561.56 1419.60 1561.56 1348.62 1490.58
1490.58 1561.56 1561.56 1419.60 1632.54 1561.56 1348.62 451.10 409.66 451.10 397.34 433.98 431.58
451.10 451.10 414.46 468.22 448.70 399.75 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 286.00 260.00 286.00 247.00 273.00 273.00 286.00 286.00 260.00 299.00
286.00 247.00 443.10 402.82 443.10 382.68 422.96 422.96 443.10 443.10 402.82 463.24 443.10 382.68 6.60
6.00 6.60 5.70 6.30 6.30 6.60 6.60 6.00 6.90 6.60 5.70 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0
11.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 286.0 260.0 286.0 247.0 273.0 273.0 286.0 286.0 260.0 299.0 286.0 247.0
5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 286.0
260.0 286.0 247.0 273.0 273.0 286.0 286.0 260.0 299.0 286.0 247.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 242.0 220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 1 1 2 1 17.9 0 22.0 20.0 22.0
```

```
19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1232 A1232 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 2412.92 2179.41 2412.92 2335.08 2412.92 2335.08 2412.92 2412.92 2335.08 2412.92 2335.08 2412.92 1542.68 1393.39 1542.68 1492.92 1542.68 1492.92 1542.68 1542.68 1492.92 1542.68 1492.92 1542.68 3984.47 3598.88 3984.47 3855.94 3984.47 3855.94 3984.47 3984.47 3855.94 3984.47 3855.94 3984.47 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 359.60 324.80 359.60 348.00 359.60 348.00 359.60 359.60 348.00 359.60 348.00 359.60 523.05 472.44 523.05 506.18 523.05 506.18 523.05 523.05 506.18 523.05 506.18 523.05 248.00 224.00 248.00 240.00 248.00 240.00 248.00 248.00 240.00 248.00 240.00 248.00 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 527.0 476.0 527.0 510.0 527.0 510.0 527.0 527.0 510.0 527.0 510.0 527.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 527.0 476.0 527.0 510.0 527.0 510.0 527.0 527.0 510.0 527.0 510.0 527.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 1 0 0 23.1 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1233 A1233 1 344 150.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 3199.20 2889.60 3199.20 3096.00 3199.20 3096.00 3199.20 3199.20 3096.00 3199.20 3096.00 3199.20 3199.20 2889.60 3199.20 3096.00 3199.20 3096.00 3199.20 3199.20 3096.00 3199.20 3096.00 3199.20 899.00 812.00 899.00 870.00 899.00 870.00 899.00 899.00 870.00 899.00 870.00 899.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 639.84 577.92 639.84 619.20 639.84 619.20 639.84 639.84 619.20 639.84 619.20 639.84 930.68 840.61 930.68 900.65 930.68 900.65 930.68 930.68 900.65 930.68 900.65 930.68 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 527.0 476.0 527.0 510.0 527.0 510.0 527.0 527.0 510.0 527.0 510.0 527.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 527.0 476.0 527.0 510.0 527.0 510.0 527.0 527.0 510.0 527.0 510.0 527.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 1 0 0 14.5 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1234 A1234 1 344 200.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 37200.00 33600.00 37200.00 36000.00 37200.00 36000.00 37200.00 37200.00 36000.00 37200.00 36000.00 37200.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 0 2 0 59.9 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1235 A1235 1 344 500.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 580.04 523.91 580.04 561.33 580.04 561.33 580.04 580.04 561.33 580.04 561.33 580.04 1568.26 1416.49 1568.26 1517.67 1568.26 1517.67 1568.26 1568.26 1517.67 1568.26 1517.67 1568.26 1068.80 965.37 1068.80 1034.32 1068.80 1034.32 1068.80 1068.80 1034.32 1068.80 1034.32 1068.80 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 119.35 107.80 119.35 115.50 119.35 115.50 119.35 119.35 115.50 119.35 115.50 119.35 134.27 121.28 134.27 129.94 134.27 129.94 134.27 134.27 129.94 134.27 129.94 134.27 13.02 11.76 13.02 12.60 13.02 12.60 13.02 13.02 12.60 13.02 12.60 13.02 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0
```

17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 1 1 2 1 6.1 -1 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

1 1237 A1237 1 344 300.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 2148.30 1940.40 2148.30 2079.00 2148.30 2079.00 2148.30 2148.30 2079.00 2148.30 2079.00 2148.30 2148.30 1940.40 2148.30 2079.00 2148.30 2079.00 2148.30 2148.30 2079.00 2148.30 2079.00 2148.30 15002.33 13543.85 15002.33 14636.34 15053.83 14533.34 15002.33 15002.33 14584.84 14950.83 14481.84 15156.83 789.60 712.83 789.60 770.33 792.31 764.91 789.60 789.60 767.62 786.89 762.20 797.73 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 286.44 258.72 286.44 277.20 286.44 277.20 286.44 286.44 277.20 286.44 277.20 286.44 381.92 344.96 381.92 369.60 381.92 369.60 381.92 381.92 369.60 381.92 369.60 381.92 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 1 0 0 61.6 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

1 1238 A1238 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 4300.32 3884.16 4300.32 4161.60 4300.32 4161.60 4300.32 4300.32 4161.60 4300.32 4161.60 4300.32 8347.68 7539.84 8347.68 8078.40 8347.68 8078.40 8347.68 8347.68 8078.40 8347.68 8078.40 8347.68 4092.00 3696.00 4092.00 3960.00 4092.00 3960.00 4092.00 4092.00 3960.00 4092.00 3960.00 4092.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 1264.80 1142.40 1264.80 1224.00 1264.80 1224.00 1264.80 1264.80 1224.00 1264.80 1224.00 1264.80 1264.80 1142.40 1264.80 1224.00 1264.80 1224.00 1264.80 1264.80 1224.00 1264.80 1224.00 1264.80 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 310.0 280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 310.0 280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 248.0 224.0 248.0 240.0 248.0 240.0 248.0 248.0 240.0 248.0 240.0 248.0 1 1 0 1 32.3 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1239 A1239 1 344 300.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 2002.00 1820.00 2002.00 1729.00 1911.00 1911.00 2002.00 2002.00 1820.00 2093.00 2002.00 1729.00 2002.00 1820.00 2002.00 1729.00 1911.00 1911.00 2002.00 2002.00 1820.00 2093.00 2002.00 1729.00 1125.74 1022.31 1125.74 991.58 1083.02 1077.03 1125.74 1125.74 1034.31 1168.47 1119.75 997.58 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 143.00 130.00 143.00 123.50 136.50 136.50 143.00 143.00 130.00 149.50 143.00 123.50 221.55 201.41 221.55 191.34 211.48 211.48 221.55 221.55 201.41 231.62 221.55 191.34 6.60 6.00 6.60 5.70 6.30 6.30 6.60 6.60 6.00 6.90 6.60 5.70 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 286.0 260.0 286.0 247.0 273.0 273.0 286.0 286.0 260.0 299.0 286.0 247.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 286.0 260.0 286.0 247.0 273.0 273.0 286.0 286.0 260.0 299.0 286.0 247.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 242.0 220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 1 1 0 1 18.4 -1 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0

1 1240 A1240 1 344 400.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 1958.78 1769.22 1958.78 1895.59 1958.78 1895.59 1958.78 1958.78 1895.59 1958.78 1895.59 1958.78 1341.61 1211.77 1341.61 1298.33 1341.61 1298.33 1341.61 1341.61 1298.33 1341.61 1298.33 1341.61 3511.58 3177.43 3511.58 3297.43 3467.54 3385.52 3511.58 3511.58 3341.48 3555.62 3429.56 3379.45 26.54 24.01 26.54 24.92 26.20 25.58 26.54 26.54 25.25 26.87 25.92 25.54 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0

22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 259.87 234.72 259.87 251.49 259.87 251.49
259.87 259.87 251.49 259.87 251.49 259.87 346.50 312.97 346.50 335.32 346.50 335.32 346.50 346.50
335.32 346.50 335.32 346.50 25.53 23.06 25.53 24.71 25.53 24.71 25.53 25.53 24.71 25.53 24.71 25.53 10.0
10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0
5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 434.0 392.0 434.0 420.0
434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0
420.0 434.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0
360.0 372.0 360.0 372.0 1 1 0 1 16.3 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
1 1241 A1241 1 344 500.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 1790.25 1617.00 1790.25 1732.50 1790.25
1732.50 1790.25 1790.25 1732.50 1790.25 1732.50 1790.25 3324.75 3003.00 3324.75 3217.50 3324.75
3217.50 3324.75 3324.75 3217.50 3324.75 3217.50 3324.75 14838.50 13402.52 14838.50 14359.84 14838.50
14359.84 14838.50 14838.50 14359.84 14838.50 14359.84 14838.50 6982.82 6307.07 6982.82 6757.57
6982.82 6757.57 6982.82 6982.82 6757.57 6982.82 6757.57 6982.82 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0
21.0 21.0 21.0 21.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 14.50 14.50 14.50 14.50 14.50
14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 710.42
641.67 710.42 687.50 710.42 687.50 710.42 710.42 687.50 710.42 687.50 710.42 1033.33 933.33 1033.33
1000.00 1033.33 1000.00 1033.33 1033.33 1000.00 1033.33 1000.00 1033.33 6.20 5.60 6.20 6.00 6.20 6.00
6.20 6.20 6.00 6.20 6.00 6.20 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0
3.0 3.0 3.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 527.0
476.0 527.0 510.0 527.0 510.0 527.0 527.0 510.0 527.0 510.0 527.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0
4.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 527.0 476.0 527.0 510.0 527.0 510.0 527.0
527.0 510.0 527.0 510.0 527.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0
360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 1 0 0 35.7 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0
30.0 31.0
1 1242 A1242 1 344 100.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 2015.05 1826.01 1993.27 1966.55 2047.71
1977.44 2025.93 1840.86 1977.44 2036.82 2020.98 2004.16 2015.05 1826.01 1993.27 1966.55 2047.71
1977.44 2025.93 1840.86 1977.44 2036.82 2020.98 2004.16 1713.17 1557.17 1657.97 1679.57 1777.57
1716.37 1731.57 1326.76 1707.17 1768.37 1817.57 1676.37 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0
15.0 15.0 15.0 15.0 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 287.86 260.86 284.75 280.94 292.53 282.49 289.42 262.98
282.49 290.97 288.71 286.31 397.30 361.95 385.99 393.06 414.26 398.71 402.95 306.81 398.71 408.61
421.33 391.64 1720.56 1559.15 1701.97 1679.15 1748.45 1688.45 1729.86 1571.83 1688.45 1739.15 1725.63
1711.27 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 6.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 5.0 5.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 15.0 17.0 17.0 17.0 17.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 15.0 17.0 17.0 17.0 17.0 5.0
5.0 5.0 5.0 5.0 5.0 5.0 7.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 15.0 17.0 17.0 17.0 17.0 343.0
312.0 335.0 338.0 355.0 342.0 347.0 279.0 342.0 351.0 358.0 339.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 5.0 5.0 5.0
5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 15.0 17.0 17.0 17.0 17.0 343.0 312.0 335.0 338.0 355.0 342.0 347.0
279.0 342.0 351.0 358.0 339.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 281.0 256.0 273.0 278.0 293.0
282.0 285.0 217.0 282.0 289.0 298.0 277.0 1 0 1 0 14.7 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0
30.0 31.0
1 1243 A1243 1 344 100.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 1108.80 1039.50 970.20 1178.10 1316.70 1247.40
1178.10 0.00 1247.40 1247.40 1524.60 1039.50 1108.80 1039.50 970.20 1178.10 1316.70 1247.40 1178.10
0.00 1247.40 1247.40 1524.60 1039.50 500.00 466.20 446.80 524.20 579.80 550.80 526.60 74.40 550.80
553.20 657.20 473.40 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 13.50 13.50
13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 158.40 148.50 138.60 168.30 188.10 178.20 168.30 0.00 178.20 178.20 217.80 148.50 211.20
198.00 184.80 224.40 250.80 237.60 224.40 0.00 237.60 237.60 290.40 198.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 0.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0
5.0 0.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0
18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0
18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0
18.0 224.0 210.0 196.0 238.0 266.0 252.0 238.0 0.0 252.0 252.0 308.0 210.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0

5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 224.0 210.0 196.0 238.0 266.0 252.0 238.0 0.0 252.0 252.0 308.0 210.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 192.0 180.0 168.0 204.0 228.0 216.0 204.0 0.0 216.0 216.0 264.0 180.0 3 1 0 0 9.7 0 16.0 15.0 14.0 17.0 19.0 18.0 17.0 0.0 18.0 18.0 22.0 15.0

1 1244 A1244 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 1258.42 1145.49 1226.16 1242.29 1306.83 1258.42 1274.56 1000.29 1258.42 1290.69 1322.96 1242.29 2442.83 2223.60 2380.19 2411.51 2536.78 2442.83 2474.14 1941.73 2442.83 2505.46 2568.10 2411.51 685.27 622.87 663.19 671.83 711.03 686.55 692.63 530.70 682.87 707.35 727.03 670.55 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 123.38 112.30 120.21 121.79 128.12 123.38 124.96 98.07 123.38 126.54 129.70 121.79 156.59 142.88 151.32 155.54 164.50 158.17 159.23 114.41 158.17 161.86 168.72 153.96 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 18.0 18.0 18.0 18.0 359.0 327.0 349.0 355.0 374.0 360.0 364.0 279.0 360.0 369.0 380.0 354.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 18.0 18.0 18.0 18.0 359.0 327.0 349.0 355.0 374.0 360.0 364.0 279.0 360.0 369.0 380.0 354.0 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 1 2 297.0 271.0 287.0 295.0 312.0 300.0 302.0 217.0 300.0 307.0 320.0 292.0 3 2 2 1 7.3 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1245 A1245 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 1783.63 1693.80 1603.96 1988.89 2168.56 2021.01 1931.18 0.00 2078.72 1963.30 2438.06 1751.51 1140.35 1082.92 1025.48 1271.59 1386.46 1292.12 1234.69 0.00 1329.02 1255.22 1558.76 1119.82 6596.00 6258.00 6064.00 7370.00 7926.00 7370.00 7128.00 744.00 7636.00 7128.00 8700.00 6596.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 265.82 252.43 239.04 296.41 323.18 301.19 287.81 0.00 309.80 292.59 363.35 261.03 389.48 370.00 350.53 434.91 473.86 441.41 421.93 0.00 454.39 428.42 532.28 382.98 234.28 222.48 210.68 261.24 284.84 265.46 253.66 0.00 273.04 257.88 320.24 230.06 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 0.0 3.0 3.0 3.0 3.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 6.0 6.0 6.0 6.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 0.0 20.0 20.0 20.0 20.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 6.0 6.0 6.0 6.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 0.0 20.0 20.0 20.0 20.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 0.0 4.0 4.0 4.0 4.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 0.0 20.0 20.0 20.0 20.0 344.0 327.0 310.0 385.0 419.0 390.0 373.0 0.0 402.0 378.0 470.0 339.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 0.0 4.0 4.0 4.0 4.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 0.0 20.0 20.0 20.0 20.0 344.0 327.0 310.0 385.0 419.0 390.0 373.0 0.0 402.0 378.0 470.0 339.0 2 2 2 1 1 1 1 2 1 1 1 2 2 2 2 1 1 1 1 2 1 1 1 2 240.0 228.0 216.0 268.0 292.0 272.0 260.0 0.0 280.0 264.0 328.0 236.0 1 1 0 0 30.9 0 22.0 21.0 20.0 25.0 27.0 25.0 24.0 0.0 26.0 24.0 30.0 22.0

1 1246 A1246 1 344 500.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 973.13 924.12 875.10 1085.12 1183.14 1102.64 1053.63 0.00 1134.13 1071.15 1330.18 955.60 1807.24 1716.21 1625.19 2015.21 2197.26 2047.76 1956.74 0.00 2106.24 1989.29 2470.33 1774.69 8970.56 8510.88 8247.04 10023.20 10779.36 10023.20 9694.08 1011.84 10384.96 9694.08 11832.00 8970.56 4221.44 4005.12 3880.96 4716.80 5072.64 4716.80 4561.92 476.16 4887.04 4561.92 5568.00 4221.44 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 386.16 366.71 347.26 430.60 469.50 437.56 418.11 0.00 450.05 425.06 527.85 379.21 565.81 537.51 509.22 631.82 688.40 641.25 612.96 0.00 660.11 622.39 773.27 556.38 8.41 7.98 7.56 9.37 10.22 9.52 9.10 0.00 9.80 9.25 11.49 8.25 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 0.0 3.0 3.0 3.0 3.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 6.0 6.0 6.0 6.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 0.0 20.0 20.0 20.0 20.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 6.0 6.0 6.0 6.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 0.0 20.0 20.0 20.0 20.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 0.0 4.0 4.0 4.0 4.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 0.0 20.0 20.0 20.0 20.0 344.0 327.0 310.0 385.0 419.0 390.0 373.0 0.0 402.0 378.0 470.0 339.0 4.0 4.0 4.0 4.0 4.0 4.0 4.0 0.0 4.0 4.0 4.0 4.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 0.0 20.0 20.0 20.0 20.0 344.0 327.0 310.0 385.0 419.0 390.0 373.0 0.0 402.0 378.0 470.0 339.0 2 2 2 1 1 1 1 2 1 1 1 2 2 2 2 1 1 1 1 2 1 1 1 2 240.0 228.0 216.0 268.0 292.0 272.0 260.0 0.0 280.0 264.0 328.0 236.0 1 1 0 0 33.1 0 22.0 21.0 20.0 25.0 27.0 25.0 24.0 0.0 26.0 24.0 30.0 22.0

1 1247 A1247 1 344 300.00 0.00 1 1 0 0 2 0 0 2 2 -1 -1 -1 2117.92 1985.55 1853.18 2250.29 2515.03 2382.66 2250.29 0.00 2382.66 2382.66 2912.14 1985.55 1354.08 1269.45 1184.82 1438.71 1607.97 1523.34 1438.71 0.00 1523.34 1523.34 1861.86 1269.45 500.00 466.20 446.80 524.20 579.80 550.80 526.60 74.40 550.80 553.20 657.20 473.40 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0

```
12.0 12.0 12.0 12.0 248.00 232.50 217.00 263.50 294.50 279.00 263.50 0.00 279.00 279.00 341.00 232.50
384.23 360.21 336.20 408.24 456.27 432.25 408.24 0.00 432.25 432.25 528.31 360.21 2.40 2.25 2.10 2.55
2.85 2.70 2.55 0.00 2.70 2.70 3.30 2.25 11.0 11.0 11.0 11.0 11.0 11.0 11.0 0.0 11.0 11.0 11.0 11.0 5.0 5.0 5.0
5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 0.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
0.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0
17.0 17.0 17.0 17.0 208.0 195.0 182.0 221.0 247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0 5.0 5.0 5.0 5.0 5.0
5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 208.0 195.0 182.0 221.0
247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 176.0 165.0
154.0 187.0 209.0 198.0 187.0 0.0 198.0 198.0 242.0 165.0 1 1 2 1 20.6 0 16.0 15.0 14.0 17.0 19.0 18.0 17.0
0.0 18.0 18.0 22.0 15.0
 1 1248 A1248 1 344 300.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 1815.36 1701.90 1588.44 1928.82 2155.74
2042.28 1928.82 0.00 2042.28 2042.28 2496.12 1701.90 1160.64 1088.10 1015.56 1233.18 1378.26 1305.72
1233.18 0.00 1305.72 1305.72 1595.88 1088.10 7500.00 6993.00 6702.00 7863.00 8697.00 8262.00 7899.00
1116.00 8262.00 8298.00 9858.00 7101.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 24.0
24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 248.00 232.50 217.00 263.50 294.50 279.00 263.50 0.00 279.00 279.00 341.00
232.50 384.23 360.21 336.20 408.24 456.27 432.25 408.24 0.00 432.25 432.25 528.31 360.21 9.60 9.00 8.40
10.20 11.40 10.80 10.20 0.00 10.80 10.80 13.20 9.00 11.0 11.0 11.0 11.0 11.0 11.0 11.0 0.0 11.0 11.0 11.0
11.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 0.0 17.0 17.0 17.0 17.0 208.0 195.0 182.0 221.0 247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0 5.0
5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 208.0
195.0 182.0 221.0 247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 176.0 165.0 154.0 187.0 209.0 198.0 187.0 0.0 198.0 198.0 242.0 165.0 1 1 0 1 46.0 -1 16.0 15.0 14.0 17.0
19.0 18.0 17.0 0.0 18.0 18.0 22.0 15.0
 1 1249 A1249 1 344 800.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 2326.78 2181.36 2035.94 2472.21 2763.06
2617.63 2472.21 0.00 2617.63 2617.63 3199.33 2181.36 1487.62 1394.64 1301.66 1580.59 1766.54 1673.57
1580.59 0.00 1673.57 1673.57 2045.47 1394.64 1982.68 1846.39 1779.95 2072.40 2286.77 2173.77 2084.04
360.84 2173.77 2185.41 2579.23 1881.31 61.32 57.10 55.05 64.09 70.72 67.23 64.45 11.16 67.23 67.59 79.77
58.18 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 286.08 268.20 250.32 303.96 339.72 321.84 303.96 0.00 321.84
321.84 393.36 268.20 443.22 415.52 387.82 470.92 526.33 498.63 470.92 0.00 498.63 498.63 609.43 415.52
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 11.0 11.0 11.0 11.0 11.0 11.0 11.0 0.0 11.0 11.0
11.0 11.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 208.0 195.0 182.0 221.0 247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0
5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 208.0
195.0 182.0 221.0 247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 176.0 165.0 154.0 187.0 209.0 198.0 187.0 0.0 198.0 198.0 242.0 165.0 1 1 0 1 31.1 -1 16.0 15.0 14.0 17.0
19.0 18.0 17.0 0.0 18.0 18.0 22.0 15.0
 1 1250 A1250 1 344 200.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 21915.77 19875.80 21459.77 21387.78
22447.78 21691.79 22067.77 18723.73 21615.79 22371.78 22527.80 21611.77 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 2 0 2 0 60.4 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1251 A1251 1 344 200.00 9.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 1262.55 1183.64 1104.73 1341.46 1499.28
1420.37 1341.46 0.00 1420.37 1420.37 1736.01 1183.64 807.21 756.76 706.31 857.66 958.56 908.11 857.66
```

0.00 908.11 908.11 1109.91 756.76 869.42 808.71 783.97 906.14 997.58 948.86 912.14 185.88 948.86 954.86 1119.75 826.70 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 147.84 138.60 129.36 157.08 175.56 166.32 157.08 0.00 166.32 166.32 203.28 138.60 147.84 138.60 129.36 157.08 175.56 166.32 157.08 0.00 166.32 166.32 203.28 138.60 1.92 1.80 1.68 2.04 2.28 2.16 2.04 0.00 2.16 2.16 2.64 1.80 14.0 14.0 14.0 14.0 14.0 14.0 14.0 0.0 14.0 14.0 14.0 14.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 0.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 0.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 0.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 0.0 16.0 16.0 16.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 0.0 16.0 16.0 16.0 16.0 160.0 150.0 140.0 170.0 190.0 180.0 170.0 0.0 180.0 180.0 220.0 150.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 0.0 16.0 16.0 16.0 16.0 160.0 150.0 140.0 170.0 190.0 180.0 170.0 0.0 180.0 180.0 220.0 150.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 128.0 120.0 112.0 136.0 152.0 144.0 136.0 0.0 144.0 144.0 176.0 120.0 1 1 0 0 15.5 0 16.0 15.0 14.0 17.0 19.0 18.0 17.0 0.0 18.0 18.0 22.0 15.0

1 1252 A1252 1 344 50.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 1355.20 1270.50 1185.80 1439.90 1609.30 1524.60 1439.90 0.00 1524.60 1524.60 1863.40 1270.50 1355.20 1270.50 1185.80 1439.90 1609.30 1524.60 1439.90 0.00 1524.60 1524.60 1863.40 1270.50 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 193.60 181.50 169.40 205.70 229.90 217.80 205.70 0.00 217.80 217.80 266.20 181.50 193.60 181.50 169.40 205.70 229.90 217.80 205.70 0.00 217.80 217.80 266.20 181.50 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 0.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 0.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 0.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 192.0 180.0 168.0 204.0 228.0 216.0 204.0 0.0 216.0 216.0 264.0 180.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 192.0 180.0 168.0 204.0 228.0 216.0 204.0 0.0 216.0 216.0 264.0 180.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 160.0 150.0 140.0 170.0 190.0 180.0 170.0 0.0 180.0 180.0 220.0 150.0 3 0 2 0 8.5 0 16.0 15.0 14.0 17.0 19.0 18.0 17.0 0.0 18.0 18.0 22.0 15.0

1 1253 A1253 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 2236.38 2046.05 2141.21 2236.38 2379.12 2283.96 2283.96 1475.06 2283.96 2331.54 2474.29 2188.79 4341.20 3971.74 4156.47 4341.20 4618.30 4433.56 4433.56 2863.34 4433.56 4525.93 4803.03 4248.83 1488.00 1344.00 1488.00 1440.00 1488.00 1440.00 1488.00 1488.00 1440.00 1488.00 1440.00 1488.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 616.65 564.17 590.41 616.65 656.01 629.77 629.77 406.72 629.77 642.89 682.25 603.53 649.45 592.59 627.58 645.07 682.25 656.01 660.38 474.51 656.01 671.31 699.74 638.51 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 3.0 3.0 7.0 3.0 7.0 7.0 7.0 7.0 7.0 3.0 3.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 15.0 20.0 20.0 20.0 20.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 15.0 20.0 20.0 20.0 20.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 15.0 20.0 20.0 20.0 20.0 359.0 327.0 349.0 355.0 374.0 360.0 364.0 279.0 360.0 369.0 380.0 354.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 15.0 20.0 20.0 20.0 20.0 359.0 327.0 349.0 355.0 374.0 360.0 364.0 279.0 360.0 369.0 380.0 354.0 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 1 2 297.0 271.0 287.0 295.0 312.0 300.0 302.0 217.0 300.0 307.0 320.0 292.0 1 1 2 0 9.9 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1254 A1254 1 344 200.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 1108.80 1039.50 970.20 1178.10 1316.70 1247.40 1178.10 0.00 1247.40 1247.40 1524.60 1039.50 1108.80 1039.50 970.20 1178.10 1316.70 1247.40 1178.10 0.00 1247.40 1247.40 1524.60 1039.50 1250.00 1165.50 1117.00 1310.50 1449.50 1377.00 1316.50 186.00 1377.00 1383.00 1643.00 1183.50 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 13.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 190.08 178.20 166.32 201.96 225.72 213.84 201.96 0.00 213.84 213.84 261.36 178.20 253.44 237.60 221.76 269.28 300.96 285.12 269.28 0.00 285.12 285.12 348.48 237.60 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 0.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 224.0 210.0 196.0 238.0 266.0 252.0 238.0 0.0 252.0 252.0 308.0 210.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 224.0 210.0 196.0 238.0

266.0 252.0 238.0 0.0 252.0 252.0 308.0 210.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 192.0 180.0 168.0 204.0 228.0 216.0 204.0 0.0 216.0 216.0 264.0 180.0 1 1 1 0 12.7 0 16.0 15.0 14.0 17.0 19.0 18.0 17.0 0.0 18.0 18.0 22.0 15.0
1 1255 A1255 1 344 500.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 623.08 584.14 545.19 662.02 739.91 700.96 662.02 0.00 700.96 700.96 856.73 584.14 1016.60 953.06 889.53 1080.14 1207.21 1143.68 1080.14 0.00 1143.68 1143.68 1397.83 953.06 946.38 880.86 851.34 987.90 1088.94 1035.42 993.90 186.08 1035.42 1041.42 1225.49 898.86 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 91.09 85.40 79.71 96.79 108.17 102.48 96.79 0.00 102.48 102.48 125.25 85.40 121.46 113.87 106.28 129.05 144.23 136.64 129.05 0.00 136.64 136.64 167.00 113.87 4.48 4.20 3.92 4.76 5.32 5.04 4.76 0.00 5.04 5.04 6.16 4.20 10.0 10.0 10.0 10.0 10.0 10.0 10.0 0.0 10.0 10.0 10.0 10.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 224.0 210.0 196.0 238.0 266.0 252.0 238.0 0.0 252.0 252.0 308.0 210.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 18.0 18.0 18.0 18.0 224.0 210.0 196.0 238.0 266.0 252.0 238.0 0.0 252.0 252.0 308.0 210.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 192.0 180.0 168.0 204.0 228.0 216.0 204.0 0.0 216.0 216.0 264.0 180.0 1 1 2 1 10.5 -1 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
1 1256 A1256 1 344 280.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 4392.08 4117.58 3843.07 4666.58 5215.60 4941.09 4666.58 0.00 4941.09 4941.09 6039.11 4117.58 4392.08 4117.58 3843.07 4666.58 5215.60 4941.09 4666.58 0.00 4941.09 4941.09 6039.11 4117.58 1175.00 1095.57 1049.98 1231.87 1362.53 1294.38 1237.51 174.84 1294.38 1300.02 1544.42 1112.49 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 345.09 323.52 301.96 366.66 409.80 388.23 366.66 0.00 388.23 388.23 474.50 323.52 534.65 501.23 467.82 568.07 634.90 601.48 568.07 0.00 601.48 601.48 735.14 501.23 19.17 17.97 16.77 20.37 22.76 21.57 20.37 0.00 21.57 21.57 26.36 17.97 11.0 11.0 11.0 11.0 11.0 11.0 11.0 0.0 11.0 11.0 11.0 11.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 208.0 195.0 182.0 221.0 247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 208.0 195.0 182.0 221.0 247.0 234.0 221.0 0.0 234.0 234.0 286.0 195.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 176.0 165.0 154.0 187.0 209.0 198.0 187.0 0.0 198.0 198.0 242.0 165.0 1 1 0 1 43.4 -1 16.0 15.0 14.0 17.0 19.0 18.0 17.0 0.0 18.0 18.0 22.0 15.0
1 1257 A1257 1 344 400.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 822.90 771.47 720.04 874.33 977.19 925.76 874.33 0.00 925.76 925.76 1131.48 771.47 519.51 487.04 454.57 551.98 616.92 584.45 551.98 0.00 584.45 584.45 714.33 487.04 2221.35 2068.66 1994.21 2321.87 2562.05 2435.44 2334.91 404.28 2435.44 2448.48 2889.71 2107.78 16.79 15.63 15.07 17.55 19.36 18.40 17.64 3.05 18.40 18.50 21.84 15.93 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 16.20 16.20 16.20 16.20 16.20 16.20 16.20 16.20 16.20 16.20 16.20 16.20 11.4 11.4 11.4 11.4 11.4 11.4 11.4 11.4 11.4 11.4 11.4 11.4 109.00 102.19 95.37 115.81 129.44 122.62 115.81 0.00 122.62 122.62 149.87 102.19 168.87 158.32 147.76 179.43 200.54 189.98 179.43 0.00 189.98 189.98 232.20 158.32 3.54 3.32 3.10 3.77 4.21 3.99 3.77 0.00 3.99 3.99 4.87 3.32 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 176.0 165.0 154.0 187.0 209.0 198.0 187.0 0.0 198.0 198.0 242.0 165.0 1 1 0 1 19.1 -1 16.0 15.0 14.0 17.0 19.0 18.0 17.0 0.0 18.0 18.0 22.0 15.0
1 1258 A1258 1 344 125.00 0.00 0 0 0 0 1 0 0 2 2 -1 -1 -1 2611.22 2358.52 2611.22 2526.99 2611.22 2526.99 2611.22 2611.22 2526.99 2611.22 2526.99 2611.22 1045.95 944.73 1045.95 1012.21 1045.95 1012.21 1045.95 1045.95 1012.21 1045.95 1012.21 1045.95 538.97 486.81 538.97 521.58 538.97 521.58 538.97 538.97 521.58 538.97 521.58 538.97 126.42 114.19 126.42 122.35 126.42 122.35 126.42 126.42 122.35 126.42 122.35 126.42 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 261.23 235.95 261.23 252.80 261.23 252.80 261.23 261.23 252.80 261.23 252.80 261.23 328.40 296.62 328.40 317.81 328.40 317.81 328.40 328.40

```
317.81 328.40 317.81 328.40 227.42 205.41 227.42 220.09 227.42 220.09 227.42 227.42 220.09 227.42
220.09 227.42 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 0.0 22.0 22.0 22.0 22.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0
720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0
744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 341.0 308.0 341.0 330.0 341.0 330.0 341.0
341.0 330.0 341.0 330.0 341.0 1 1 0 1 13.1 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1259 A1259 1 344 125.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 2726.91 2463.01 2726.91 2638.94 2726.91
2638.94 2726.91 2726.91 2638.94 2726.91 2638.94 2726.91 1092.29 986.59 1092.29 1057.06 1092.29
1057.06 1092.29 1092.29 1057.06 1092.29 1057.06 1092.29 350.34 317.11 347.91 340.89 353.99 342.10
351.56 330.88 342.10 352.78 346.97 349.13 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 27.0
27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 327.36 295.68 327.36 316.80 327.36 316.80 327.36 327.36 316.80 327.36
316.80 327.36 603.03 544.67 603.03 583.58 603.03 583.58 603.03 603.03 583.58 603.03 583.58 603.03 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 22.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 527.0 476.0 527.0 510.0 527.0 510.0 527.0 527.0 510.0 527.0 510.0 527.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 527.0 476.0 527.0
510.0 527.0 510.0 527.0 527.0 510.0 527.0 510.0 527.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 217.0
196.0 217.0 210.0 217.0 210.0 217.0 217.0 210.0 217.0 210.0 217.0 1 1 0 0 24.7 0 31.0 28.0 31.0 30.0 31.0
30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1260 A1260 1 344 150.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 1417.15 1352.74 1288.32 1610.40 1739.23
1610.40 1545.98 0.00 1674.82 1545.98 1932.48 1417.15 906.05 864.86 823.68 1029.60 1111.97 1029.60
988.42 0.00 1070.78 988.42 1235.52 906.05 2168.38 2022.07 2055.84 2307.19 2449.75 2307.19 2280.93
930.38 2363.46 2280.93 2588.55 2168.38 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 24.0
24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 232.32 221.76 211.20 264.00 285.12 264.00 253.44 0.00 274.56 253.44 316.80
232.32 401.85 383.58 365.32 456.65 493.18 456.65 438.38 0.00 474.91 438.38 547.98 401.85 46.93 44.80
42.67 53.33 57.60 53.33 51.20 0.00 55.47 51.20 64.00 46.93 6.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 6.0 6.0 6.0 6.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 0.0 1.0 1.0 1.0 1.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 0.0 22.0 22.0 22.0 22.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 0.0 22.0 22.0 22.0 22.0 5.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
0.0 22.0 22.0 22.0 22.0 396.0 378.0 360.0 450.0 486.0 450.0 432.0 0.0 468.0 432.0 540.0 396.0 5.0 5.0 5.0 5.0
5.0 5.0 5.0 0.0 5.0 5.0 5.0 5.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 0.0 22.0 22.0 22.0 22.0 396.0 378.0 360.0
450.0 486.0 450.0 432.0 0.0 468.0 432.0 540.0 396.0 1 1 2 1 1 1 1 2 1 1 1 1 1 1 2 1 1 1 1 2 1 1 1 1 352.0
336.0 320.0 400.0 432.0 400.0 384.0 0.0 416.0 384.0 480.0 352.0 1 1 0 0 12.2 0 22.0 21.0 20.0 25.0 27.0 25.0
24.0 0.0 26.0 24.0 30.0 22.0
 1 1261 A1261 1 344 300.00 0.00 0 0 0 0 1 0 0 2 2 -1 -1 -1 1742.40 1633.50 1524.60 1851.30 2069.10
1960.20 1851.30 0.00 1960.20 1960.20 2395.80 1633.50 1742.40 1633.50 1524.60 1851.30 2069.10 1960.20
1851.30 0.00 1960.20 1960.20 2395.80 1633.50 9709.00 9041.62 8716.25 10148.37 11198.12 10644.75
10205.37 1767.00 10644.75 10701.75 12630.25 9212.62 511.00 475.87 458.75 534.12 589.37 560.25 537.12
93.00 560.25 563.25 664.75 484.87 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00
15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 232.32 217.80 203.28 246.84 275.88
261.36 246.84 0.00 261.36 261.36 319.44 217.80 232.32 217.80 203.28 246.84 275.88 261.36 246.84 0.00
261.36 261.36 319.44 217.80 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 198.0 180.0 270.0
78.0 60.0 126.0 174.0 624.0 102.0 180.0 60.0 120.0 6.0 6.0 6.0 6.0 6.0 12.0 12.0 96.0 12.0 6.0 6.0 12.0 8.0 8.0
8.0 8.0 8.0 8.0 8.0 0.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0
8.0 8.0 8.0 0.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 6.0 6.0 6.0 6.0 6.0 6.0
6.0 0.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 192.0 180.0 168.0 204.0
228.0 216.0 204.0 0.0 216.0 216.0 264.0 180.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 6.0 6.0 6.0 6.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 0.0 17.0 17.0 17.0 17.0 192.0 180.0 168.0 204.0 228.0 216.0 204.0 0.0 216.0 216.0 264.0
180.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 160.0 150.0 140.0 170.0 190.0 180.0 170.0 0.0 180.0
180.0 220.0 150.0 1 1 0 0 62.9 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1262 A1262 1 344 150.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 1636.80 1478.40 1636.80 1584.00 1636.80
```

1584.00 1636.80 1636.80 1584.00 1636.80 1584.00 1636.80 1636.80 1478.40 1636.80 1584.00 1636.80 1584.00 1636.80 1636.80 1584.00 1636.80 1584.00 1636.80 355.21 323.19 347.91 340.89 356.43 329.94 330.88 330.88 323.85 358.86 346.97 339.39 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 327.36 295.68 327.36 316.80 327.36 316.80 327.36 327.36 316.80 327.36 316.80 327.36 603.03 544.67 603.03 583.58 603.03 583.58 603.03 603.03 583.58 603.03 583.58 603.03 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 527.0 476.0 527.0 510.0 527.0 510.0 527.0 527.0 510.0 527.0 510.0 527.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 527.0 476.0 527.0 510.0 527.0 510.0 527.0 527.0 510.0 527.0 510.0 527.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 217.0 196.0 217.0 210.0 217.0 210.0 217.0 217.0 210.0 217.0 210.0 217.0 1 1 0 0 15.5 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1263 A1263 1 344 80.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 1991.85 1799.09 1991.85 1927.60 1991.85 1927.60 1991.85 1991.85 1927.60 1991.85 1927.60 1991.85 1273.48 1150.24 1273.48 1232.40 1273.48 1232.40 1273.48 1273.48 1232.40 1273.48 1232.40 1273.48 538.97 486.81 538.97 521.58 538.97 521.58 538.97 538.97 521.58 538.97 521.58 538.97 126.42 114.19 126.42 122.35 126.42 122.35 126.42 126.42 122.35 126.42 122.35 126.42 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 326.53 294.93 326.53 316.00 326.53 316.00 326.53 326.53 316.00 326.53 316.00 326.53 410.50 370.77 410.50 397.26 410.50 397.26 410.50 410.50 397.26 410.50 397.26 410.50 248.00 224.00 248.00 240.00 248.00 240.00 248.00 248.00 240.00 248.00 240.00 248.00 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 11.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 0.0 0.0 7.0 7.0 0.0 22.0 22.0 22.0 22.0 22.0 22.0 0.0 0.0 0.0 22.0 22.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 403.0 364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0 403.0 390.0 403.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 403.0 364.0 403.0 390.0 403.0 390.0 403.0 403.0 390.0 403.0 390.0 403.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 1 1 0 1 10.8 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1264 A1264 1 344 100.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 2323.97 2099.07 2323.97 2249.00 2323.97 2249.00 2323.97 2323.97 2249.00 2323.97 2249.00 2323.97 2323.97 2099.07 2323.97 2249.00 2323.97 2249.00 2323.97 2323.97 2249.00 2323.97 2249.00 2323.97 1765.87 1638.91 1629.66 1655.45 1788.57 1451.14 1311.85 1311.85 1337.64 1833.97 1768.95 1470.76 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 332.00 299.87 332.00 321.29 332.00 321.29 332.00 332.00 321.29 332.00 321.29 332.00 373.49 337.35 373.49 361.45 373.49 361.45 373.49 373.49 361.45 373.49 361.45 373.49 3720.00 3360.00 3720.00 3600.00 3720.00 3600.00 3720.00 3720.00 3600.00 3720.00 3600.00 3720.00 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 0.0 0.0 0.0 9.0 9.0 0.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 0.0 0.0 17.0 17.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 1 0 1 0 14.2 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1265 A1265 1 344 100.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 1669.52 1490.64 1699.33 1565.17 1684.42 1669.52 1848.39 1848.39 1714.24 1714.24 1639.71 1714.24 1669.52 1490.64 1699.33 1565.17 1684.42 1669.52 1848.39 1848.39 1714.24 1714.24 1639.71 1714.24 565.81 517.08 541.59 523.62 574.46 501.13 503.54 503.54 485.57 591.76 566.87 508.73 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 238.50 212.95 242.76 223.60 240.63 238.50 264.06 264.06 244.89 244.89 234.24 244.89 238.50 212.95 242.76 223.60 240.63 238.50 264.06 264.06 244.89 244.89 234.24 244.89 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 14.0 12.0 14.0 14.0 12.0 14.0 14.0 14.0 14.0 12.0 12.0

14.0 6.0 6.0 6.0 6.0 6.0 7.0 7.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 9.0 9.0 9.0 8.0 8.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 16.0 16.0 16.0 17.0 17.0 16.0 8.0 8.0 8.0 8.0 8.0 8.0 9.0 9.0 9.0 8.0 8.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 16.0 16.0 16.0 17.0 17.0 16.0 6.0 6.0 6.0 6.0 6.0 6.0 7.0 7.0 7.0 6.0 6.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 16.0 16.0 16.0 17.0 17.0 16.0 270.0 240.0 278.0 254.0 272.0 276.0 310.0 310.0 286.0 276.0 264.0 284.0 6.0 6.0 6.0 6.0 6.0 6.0 7.0 7.0 7.0 6.0 6.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 16.0 16.0 16.0 17.0 17.0 16.0 270.0 240.0 278.0 254.0 272.0 276.0 310.0 310.0 286.0 276.0 264.0 284.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 224.0 200.0 228.0 210.0 226.0 224.0 248.0 248.0 230.0 230.0 220.0 230.0 3 1 0 0 9.3 0 23.0 20.0 25.0 22.0 23.0 26.0 31.0 31.0 28.0 23.0 22.0 27.0

1 1266 A1266 1 344 300.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 2904.90 2563.15 3047.30 2748.27 2919.14 3075.78 3531.45 3531.45 3232.42 2947.62 2819.47 3175.46 2904.90 2563.15 3047.30 2748.27 2919.14 3075.78 3531.45 3531.45 3232.42 2947.62 2819.47 3175.46 1461.05 1335.30 1398.18 1351.78 1483.50 1293.41 1299.40 1299.40 1252.99 1528.40 1464.03 1312.87 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 207.49 183.08 217.66 196.30 208.51 219.70 252.25 252.25 230.89 210.54 201.39 226.82 268.52 244.11 260.38 248.18 272.59 244.11 252.25 252.25 240.04 280.73 268.52 248.18 3.40 3.00 3.57 3.22 3.42 3.60 4.13 4.13 3.78 3.45 3.30 3.72 14.0 10.0 14.0 14.0 10.0 14.0 14.0 14.0 14.0 10.0 10.0 14.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 7.0 7.0 9.0 18.0 18.0 18.0 18.0 18.0 18.0 16.0 16.0 16.0 18.0 18.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 7.0 7.0 9.0 18.0 18.0 18.0 18.0 18.0 18.0 16.0 16.0 16.0 18.0 18.0 16.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 5.0 5.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 16.0 16.0 16.0 18.0 18.0 16.0 310.0 280.0 306.0 288.0 314.0 292.0 310.0 310.0 292.0 322.0 308.0 298.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 5.0 5.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 16.0 16.0 16.0 18.0 18.0 16.0 310.0 280.0 306.0 288.0 314.0 292.0 310.0 310.0 292.0 322.0 308.0 298.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 264.0 240.0 256.0 244.0 268.0 240.0 248.0 248.0 236.0 276.0 264.0 244.0 1 1 0 1 19.0 -1 23.0 20.0 25.0 22.0 23.0 26.0 31.0 31.0 28.0 23.0 22.0 27.0

1 1267 A1267 1 344 150.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 1803.65 1803.65 1288.32 1610.40 1868.06 772.99 0.00 0.00 322.08 1996.90 1932.48 708.58 1153.15 1153.15 823.68 1029.60 1194.34 494.21 0.00 0.00 205.92 1276.70 1235.52 453.02 2506.02 2415.98 2055.84 2307.19 2562.29 1575.64 930.38 930.38 1181.73 2674.84 2588.55 1549.38 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 295.68 295.68 211.20 264.00 306.24 126.72 0.00 0.00 52.80 327.36 316.80 116.16 511.45 511.45 365.32 456.65 529.71 219.19 0.00 0.00 91.33 566.24 547.98 200.93 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 6.0 6.0 6.0 6.0 6.0 6.0 0.0 0.0 6.0 6.0 6.0 6.0 1.0 1.0 1.0 1.0 1.0 1.0 0.0 0.0 1.0 1.0 1.0 1.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 0.0 0.0 7.0 7.0 0.0 22.0 22.0 22.0 22.0 22.0 22.0 0.0 0.0 0.0 22.0 22.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 0.0 0.0 7.0 7.0 0.0 22.0 22.0 22.0 22.0 22.0 22.0 0.0 0.0 0.0 22.0 22.0 0.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 0.0 0.0 5.0 5.0 0.0 22.0 22.0 22.0 22.0 22.0 22.0 0.0 0.0 0.0 22.0 22.0 0.0 504.0 504.0 360.0 450.0 522.0 216.0 0.0 0.0 90.0 558.0 540.0 198.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 0.0 0.0 5.0 5.0 0.0 22.0 22.0 22.0 22.0 22.0 22.0 0.0 0.0 0.0 22.0 22.0 0.0 504.0 504.0 360.0 450.0 522.0 216.0 0.0 0.0 90.0 558.0 540.0 198.0 1 1 2 1 1 2 2 2 2 1 1 2 1 1 2 1 1 2 2 2 2 1 1 2 448.0 448.0 320.0 400.0 464.0 192.0 0.0 0.0 80.0 496.0 480.0 176.0 1 1 0 0 12.2 0 28.0 28.0 20.0 25.0 29.0 12.0 0.0 0.0 5.0 31.0 30.0 11.0

1 1268 A1268 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 1334.50 1334.50 951.65 1186.83 1383.72 568.80 0.00 0.00 235.18 1482.17 1432.95 519.58 2590.50 2590.50 1847.32 2303.85 2686.05 1104.15 0.00 0.00 456.52 2877.15 2781.60 1008.60 654.46 647.24 486.22 582.25 677.73 315.58 74.58 74.58 170.61 724.26 698.59 294.72 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 130.83 130.83 93.30 116.36 135.66 55.77 0.00 0.00 23.06 145.31 140.48 50.94 167.30 167.30 119.04 147.99 173.73 70.78 0.00 0.00 28.95 186.60 180.16 64.34 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 13.0 13.0 13.0 13.0 13.0 13.0 0.0 0.0 13.0 13.0 13.0 13.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 0.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 0.0 0.0 7.0 7.0 0.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 0.0 0.0 18.0 18.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 0.0 0.0 7.0 7.0 0.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 0.0 0.0 18.0 18.0 0.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 0.0 0.0 5.0 5.0 0.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 0.0 0.0 18.0 18.0 0.0 368.0 368.0 262.0 326.0 382.0 156.0 0.0 0.0 64.0 410.0 396.0 142.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 0.0 0.0 5.0 5.0 0.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 0.0 0.0 18.0 18.0 0.0 368.0 368.0 262.0 326.0 382.0 156.0 0.0 0.0 64.0 410.0 396.0 142.0 2 1 2 2 1 2 2 2 2 1 1 2 2 1 2 2 1 2 2 2 2 1 1 2 312.0 312.0 222.0 276.0 324.0 132.0 0.0 0.0 54.0 348.0 336.0 120.0 3 2 2 1 7.4 -1 28.0 28.0 20.0 25.0 29.0 12.0 0.0 0.0 5.0 31.0 30.0 11.0

1 1269 A1269 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 2670.85 2546.30 2255.69 2421.76 2740.04 1799.02 1286.99 1286.99 1453.05 2878.43 2767.72 1771.34 1707.59 1627.96 1442.16 1548.34 1751.83 1150.19 822.83 822.83 929.00 1840.31 1769.53 1132.50 6047.80 5587.72 5659.72 5700.40 6112.48 5118.29 4754.21 4754.21 4794.89 6241.84 6023.80 5206.97 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00

0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 398.04 379.48 336.17 360.92 408.35 268.11 191.80 191.80 216.55 428.98 412.48 263.99 461.97 437.23 400.10 422.79 472.29 329.98 255.74 255.74 278.42 492.91 474.35 327.92 144.75 138.00 122.25 131.25 148.50 97.50 69.75 69.75 78.75 156.00 150.00 96.00 18.0 14.0 18.0 18.0 14.0 18.0 18.0 18.0 18.0 14.0 14.0 18.0 6.0 6.0 7.0 6.0 6.0 7.0 9.0 9.0 7.0 6.0 6.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 11.0 11.0 11.0 9.0 9.0 11.0 17.0 17.0 17.0 17.0 17.0 17.0 14.0 14.0 14.0 17.0 17.0 14.0 9.0 9.0 9.0 9.0 9.0 9.0 0.0 0.0 0.0 9.0 9.0 0.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 0.0 0.0 17.0 17.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 7.0 7.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 14.0 14.0 14.0 17.0 17.0 14.0 318.0 300.0 280.0 297.0 323.0 236.0 186.0 186.0 203.0 333.0 322.0 237.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 7.0 7.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 14.0 14.0 14.0 17.0 17.0 14.0 318.0 300.0 280.0 297.0 323.0 236.0 186.0 186.0 203.0 333.0 322.0 237.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 224.0 212.0 194.0 205.0 229.0 160.0 124.0 124.0 135.0 239.0 230.0 159.0 1 1 0 0 31.0 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1270 A1270 1 344 150.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 4537.30 4537.30 3235.62 4035.22 4704.66 1933.93 0.00 0.00 799.61 5039.38 4872.02 1766.57 8807.70 8807.70 6280.90 7833.08 9132.57 3754.10 0.00 0.00 1552.18 9782.32 9457.45 3429.23 4896.33 4842.33 3637.66 4356.11 5070.40 2360.99 557.94 557.94 1276.39 5418.54 5226.47 2204.92 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 1334.50 1334.50 951.65 1186.83 1383.72 568.80 0.00 0.00 235.18 1482.17 1432.95 519.58 1706.41 1706.41 1214.18 1509.52 1772.04 721.94 0.00 0.00 295.34 1903.30 1837.67 656.31 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 13.0 13.0 13.0 13.0 13.0 13.0 0.0 0.0 13.0 13.0 13.0 13.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 0.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 0.0 0.0 7.0 7.0 0.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 0.0 0.0 18.0 18.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 0.0 0.0 7.0 7.0 0.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 0.0 0.0 18.0 18.0 0.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 0.0 0.0 5.0 5.0 0.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 0.0 0.0 18.0 18.0 0.0 368.0 368.0 262.0 326.0 382.0 156.0 0.0 0.0 64.0 410.0 396.0 142.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 0.0 0.0 5.0 5.0 0.0 18.0 18.0 18.0 18.0 18.0 18.0 0.0 0.0 0.0 18.0 18.0 0.0 368.0 368.0 262.0 326.0 382.0 156.0 0.0 0.0 64.0 410.0 396.0 142.0 2 1 2 2 1 2 2 2 2 1 1 2 2 1 2 2 1 2 2 2 2 1 1 2 312.0 312.0 222.0 276.0 324.0 132.0 0.0 0.0 54.0 348.0 336.0 120.0 1 1 0 1 32.7 0 28.0 28.0 20.0 25.0 29.0 12.0 0.0 0.0 5.0 31.0 30.0 11.0

1 1271 A1271 1 344 500.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 1361.29 1252.87 1289.01 1289.01 1373.34 1180.59 1120.35 1120.35 1120.35 1397.43 1349.24 1204.68 2528.11 2326.75 2393.87 2393.87 2550.48 2192.52 2080.66 2080.66 2080.66 2595.23 2505.74 2237.26 7674.56 7184.71 6890.71 7119.35 7805.20 5943.57 5061.72 5061.72 5290.37 8066.48 7772.56 5976.21 3611.56 3381.04 3242.69 3350.28 3673.03 2796.98 2381.99 2381.99 2489.58 3795.99 3657.67 2812.34 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 21.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 14.50 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 540.19 497.17 511.51 511.51 544.97 468.49 444.58 444.58 444.58 554.54 535.41 478.05 614.29 564.10 585.61 583.22 619.07 540.19 518.68 518.68 516.29 628.63 607.12 552.15 9.32 8.58 8.83 8.83 9.41 8.09 7.67 7.67 7.67 9.57 9.24 8.25 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 6.0 6.0 7.0 6.0 6.0 7.0 7.0 7.0 7.0 6.0 6.0 7.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 15.0 15.0 15.0 17.0 17.0 15.0 9.0 9.0 9.0 9.0 9.0 9.0 0.0 0.0 0.0 9.0 9.0 0.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 0.0 0.0 17.0 17.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 15.0 15.0 15.0 17.0 17.0 15.0 319.0 292.0 307.0 304.0 321.0 286.0 279.0 279.0 276.0 325.0 314.0 293.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 15.0 15.0 15.0 17.0 17.0 15.0 319.0 292.0 307.0 304.0 321.0 286.0 279.0 279.0 276.0 325.0 314.0 293.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 257.0 236.0 245.0 244.0 259.0 226.0 217.0 217.0 216.0 263.0 254.0 231.0 1 1 0 0 29.2 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1272 A1272 1 344 300.00 0.00 1 1 0 0 2 0 0 2 2 -1 -1 -1 3410.02 3100.02 3306.69 3151.69 3461.69 3100.02 3203.35 3203.35 3048.35 3565.02 3410.02 3151.69 2180.18 1981.98 2114.11 2015.01 2213.21 1981.98 2048.05 2048.05 1948.95 2279.28 2180.18 2015.01 565.81 517.08 541.59 523.62 574.46 501.13 503.54 503.54 485.57 591.76 566.87 508.73 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 399.30 363.00 387.20 369.05 405.35 363.00 375.10 375.10 356.95 417.45 399.30 369.05 526.35 484.00 494.08 481.98 536.43 447.70 437.62 437.62 425.52 556.60 532.40 451.73 3.30 3.00 3.20 3.05 3.35 3.00 3.10 3.10 2.95 3.45 3.30 3.05 15.0 10.0 15.0 15.0 10.0 15.0 15.0 15.0 15.0 10.0 10.0 15.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 7.0 7.0 9.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 15.0 15.0 18.0 18.0 15.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 7.0 7.0 9.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 15.0 15.0 18.0 18.0 15.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 5.0 5.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 15.0 15.0 18.0 18.0 15.0 307.0 280.0 295.0 283.0 312.0 274.0 279.0 279.0 267.0 322.0 308.0 278.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 5.0 5.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 15.0 15.0 18.0 18.0 15.0 307.0

```
280.0 295.0 283.0 312.0 274.0 279.0 279.0 267.0 322.0 308.0 278.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 261.0 240.0 245.0 239.0 266.0 222.0 217.0 217.0 211.0 276.0 264.0 224.0 1 1 2 1 19.1 0 23.0 20.0 25.0
22.0 23.0 26.0 31.0 31.0 28.0 23.0 22.0 27.0
 1 1273 A1273 1 344 300.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 2708.40 2708.40 1895.88 2302.14 2843.82
1083.36 0.00 0.00 406.26 3114.66 2979.24 947.94 1731.60 1731.60 1212.12 1471.86 1818.18 692.64 0.00
0.00 259.74 1991.34 1904.76 606.06 7956.66 7848.65 5904.49 6894.57 8298.69 3816.32 1116.09 1116.09
2106.17 8982.75 8604.71 3510.29 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 24.0 24.0 24.0
24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00
17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 370.00 370.00 259.00 314.50 388.50 148.00 0.00 0.00 55.50 425.50 407.00 129.50 416.25
416.25 291.38 353.81 437.06 166.50 0.00 0.00 62.44 478.69 457.88 145.69 12.00 12.00 8.40 10.20 12.60 4.80
0.00 0.00 1.80 13.80 13.20 4.20 13.0 13.0 13.0 13.0 13.0 13.0 0.0 0.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0
6.0 0.0 0.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 0.0 0.0 0.0 9.0 9.0 0.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 0.0
0.0 17.0 17.0 0.0 9.0 9.0 9.0 9.0 9.0 9.0 0.0 0.0 0.0 9.0 9.0 0.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 0.0 0.0 17.0
17.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 0.0 0.0 7.0 7.0 0.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 0.0 0.0 17.0 17.0 0.0
220.0 220.0 154.0 187.0 231.0 88.0 0.0 0.0 33.0 253.0 242.0 77.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 0.0 0.0 7.0 7.0 0.0
17.0 17.0 17.0 17.0 17.0 17.0 0.0 0.0 0.0 17.0 17.0 0.0 220.0 220.0 154.0 187.0 231.0 88.0 0.0 0.0 33.0 253.0
242.0 77.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 180.0 180.0 126.0 153.0 189.0 72.0 0.0 0.0 27.0
207.0 198.0 63.0 1 1 0 1 44.7 -1 20.0 20.0 14.0 17.0 21.0 8.0 0.0 0.0 3.0 23.0 22.0 7.0
 1 1275 A1275 1 344 800.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 2053.26 1866.60 1991.04 1897.71 2084.37
1866.60 1928.82 1928.82 1835.49 2146.59 2053.26 1897.71 1312.74 1193.40 1272.96 1213.29 1332.63
1193.40 1233.18 1233.18 1173.51 1372.41 1312.74 1213.29 2580.24 2401.61 2340.75 2353.06 2636.12
2041.72 1845.81 1845.81 1858.12 2747.88 2632.46 2045.38 79.80 74.28 72.39 72.77 81.53 63.15 57.09 57.09
57.47 84.99 81.42 63.26 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 252.45 229.50 244.80 233.32 256.27 229.50
237.15 237.15 225.68 263.92 252.45 233.32 332.78 306.00 312.38 304.73 339.15 283.05 276.67 276.67
269.02 351.90 336.60 285.60 7.26 6.60 7.04 6.71 7.37 6.60 6.82 6.82 6.49 7.59 7.26 6.71 15.0 10.0 15.0 15.0
10.0 15.0 15.0 15.0 15.0 10.0 10.0 15.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0
9.0 9.0 7.0 7.0 9.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 15.0 15.0 18.0 18.0 15.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0
9.0 7.0 7.0 9.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 15.0 15.0 18.0 18.0 15.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0
5.0 5.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 15.0 15.0 18.0 18.0 15.0 307.0 280.0 295.0 283.0 312.0 274.0
279.0 279.0 267.0 322.0 308.0 278.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 5.0 5.0 7.0 18.0 18.0 18.0 18.0 18.0
18.0 15.0 15.0 15.0 18.0 18.0 15.0 307.0 280.0 295.0 283.0 312.0 274.0 279.0 279.0 267.0 322.0 308.0 278.0 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 261.0 240.0 245.0 239.0 266.0 222.0 217.0 217.0 211.0 276.0
264.0 224.0 1 1 0 1 18.8 -1 23.0 20.0 25.0 22.0 23.0 26.0 31.0 31.0 28.0 23.0 22.0 27.0
 1 1276 A1276 1 344 300.00 0.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 1931.79 1742.40 1931.79 1912.85 1950.73
1874.97 1931.79 1931.79 1893.91 1912.85 1856.03 1988.61 1931.79 1742.40 1931.79 1912.85 1950.73
1874.97 1931.79 1931.79 1893.91 1912.85 1856.03 1988.61 9708.91 8759.29 9708.91 9574.05 9786.77
9418.33 9708.91 9708.91 9496.19 9631.05 9340.46 9942.50 511.00 461.02 511.00 503.90 515.09 495.70
511.00 511.00 499.80 506.90 491.60 523.29 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 18.0
18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00
15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 257.57 232.32 257.57 255.05
260.10 250.00 257.57 257.57 252.52 255.05 247.47 265.15 285.35 257.57 285.35 279.04 286.61 276.51
285.35 285.35 277.77 284.09 275.25 289.14 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 15.0
15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 9.0 9.0 9.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 288.0 260.0 288.0 281.0
289.0 279.0 288.0 288.0 280.0 287.0 278.0 291.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 15.0 15.0 15.0
15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 288.0 260.0 288.0 281.0 289.0 279.0 288.0 288.0 280.0 287.0
278.0 291.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 226.0 204.0 226.0 221.0 227.0 219.0 226.0 226.0
220.0 225.0 218.0 229.0 1 1 0 0 50.8 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1277 A1277 1 344 200.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 30477.03 27527.64 30477.03 29493.90
30477.03 29493.90 30477.03 30477.03 29493.90 30477.03 29493.90 30477.03 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 -9999 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
```

```
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 2 0 2 0 64.3 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1278 A1278 1 344 200.00 9.00 0 0 0 0 2 0 0 2 2 -1 -1 -1 1916.38 1916.38 1341.46 1628.92 2012.19 766.55
0.00 0.00 287.46 2203.83 2108.01 670.73 1225.22 1225.22 857.66 1041.44 1286.49 490.09 0.00 0.00 183.78
1409.01 1347.75 428.83 1324.68 1306.70 983.02 1147.86 1381.62 635.37 185.81 185.81 350.65 1495.51
1432.57 584.42 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00
16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
224.40 224.40 157.08 190.74 235.62 89.76 0.00 0.00 33.66 258.06 246.84 78.54 224.40 224.40 157.08 190.74
235.62 89.76 0.00 0.00 33.66 258.06 246.84 78.54 2.40 2.40 1.68 2.04 2.52 0.96 0.00 0.00 0.36 2.76 2.64 0.84
12.0 12.0 12.0 12.0 12.0 12.0 0.0 0.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 0.0 6.0 6.0 6.0 6.0 8.0 8.0
8.0 8.0 8.0 8.0 0.0 0.0 0.0 8.0 8.0 0.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 0.0 0.0 17.0 17.0 0.0 8.0 8.0 8.0 8.0 8.0
8.0 0.0 0.0 0.0 8.0 8.0 0.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 0.0 0.0 17.0 17.0 0.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 0.0
0.0 6.0 6.0 0.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 0.0 0.0 17.0 17.0 0.0 240.0 240.0 168.0 204.0 252.0 96.0 0.0
0.0 36.0 276.0 264.0 84.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 0.0 0.0 6.0 6.0 0.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 0.0
0.0 17.0 17.0 0.0 240.0 240.0 168.0 204.0 252.0 96.0 0.0 0.0 36.0 276.0 264.0 84.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 200.0 200.0 140.0 170.0 210.0 80.0 0.0 0.0 30.0 230.0 220.0 70.0 1 1 0 0 14.2 0 20.0 20.0
14.0 17.0 21.0 8.0 0.0 0.0 3.0 23.0 22.0 7.0
 1 1279 A1279 1 344 50.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 1848.00 1848.00 1293.60 1570.80 1940.40 739.20
0.00 0.00 277.20 2125.20 2032.80 646.80 1848.00 1848.00 1293.60 1570.80 1940.40 739.20 0.00 0.00 277.20
2125.20 2032.80 646.80 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00
16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 264.00 264.00 184.80 224.40 277.20
105.60 0.00 0.00 39.60 303.60 290.40 92.40 264.00 264.00 184.80 224.40 277.20 105.60 0.00 0.00 39.60
303.60 290.40 92.40 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 9.0 9.0 9.0 9.0 9.0 9.0 0.0 0.0
9.0 9.0 9.0 9.0 4.0 4.0 4.0 4.0 4.0 4.0 0.0 0.0 4.0 4.0 4.0 4.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 0.0 0.0 7.0 7.0 0.0 19.0
19.0 19.0 19.0 19.0 19.0 0.0 0.0 0.0 19.0 19.0 0.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 0.0 0.0 7.0 7.0 0.0 19.0 19.0 19.0
19.0 19.0 19.0 0.0 0.0 0.0 19.0 19.0 0.0 5.0 5.0 5.0 5.0 5.0 5.0 0.0 0.0 0.0 5.0 5.0 0.0 19.0 19.0 19.0 19.0 19.0
19.0 0.0 0.0 0.0 19.0 19.0 0.0 300.0 300.0 210.0 255.0 315.0 120.0 0.0 0.0 45.0 345.0 330.0 105.0 5.0 5.0 5.0
5.0 5.0 5.0 0.0 0.0 0.0 5.0 5.0 0.0 19.0 19.0 19.0 19.0 19.0 19.0 0.0 0.0 0.0 19.0 19.0 0.0 300.0 300.0 210.0
255.0 315.0 120.0 0.0 0.0 45.0 345.0 330.0 105.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 260.0 260.0
182.0 221.0 273.0 104.0 0.0 0.0 39.0 299.0 286.0 91.0 3 0 2 0 7.1 0 20.0 20.0 14.0 17.0 21.0 8.0 0.0 0.0 3.0
23.0 22.0 7.0
 1 1280 A1280 1 344 300.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 2633.95 2398.07 2574.98 2525.84 2643.78
2437.38 2437.38 2437.38 2388.24 2663.43 2574.98 2506.18 5112.96 4655.08 4998.49 4903.10 5132.04
4731.39 4731.39 4731.39 4636.00 5170.19 4998.49 4864.94 1488.00 1344.00 1488.00 1440.00 1488.00
1440.00 1488.00 1488.00 1440.00 1488.00 1440.00 1488.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0 28.0
28.0 28.0 28.0 28.0 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 20.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 726.27 661.23 710.01 696.46 728.98 672.07 672.07 672.07
658.52 734.40 710.01 691.04 918.68 845.51 869.90 869.90 926.81 796.73 756.08 756.08 756.08 943.07
910.55 812.99 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 13.0 13.0 13.0 13.0 13.0 13.0 13.0
13.0 13.0 13.0 13.0 13.0 5.0 5.0 5.0 5.0 5.0 5.0 6.0 6.0 6.0 5.0 5.0 6.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 7.0 7.0
9.0 18.0 18.0 18.0 18.0 18.0 18.0 17.0 17.0 17.0 18.0 18.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 0.0 0.0 0.0 7.0 7.0 0.0
18.0 18.0 18.0 18.0 18.0 18.0 0.0 0.0 0.0 18.0 18.0 0.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 5.0 5.0 7.0 18.0 18.0
18.0 18.0 18.0 18.0 17.0 17.0 17.0 18.0 18.0 17.0 401.0 368.0 383.0 381.0 404.0 354.0 341.0 341.0 339.0
410.0 396.0 362.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 5.0 5.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 17.0 17.0 17.0
18.0 18.0 17.0 401.0 368.0 383.0 381.0 404.0 354.0 341.0 341.0 339.0 410.0 396.0 362.0 1 1 1 1 1 2 2 2 2 1 1
2 1 1 1 1 1 2 2 2 2 1 1 2 339.0 312.0 321.0 321.0 342.0 294.0 279.0 279.0 279.0 348.0 336.0 300.0 1 1 2 0
11.8 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1281 A1281 1 344 200.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 1570.80 1570.80 1099.56 1335.18 1649.34 628.32
0.00 0.00 235.62 1806.42 1727.88 549.78 1570.80 1570.80 1099.56 1335.18 1649.34 628.32 0.00 0.00 235.62
1806.42 1727.88 549.78 3109.98 2957.03 2651.13 2829.58 3186.46 2141.30 1580.48 1580.48 1758.93
3339.41 3211.95 2115.81 548.82 521.83 467.85 499.34 562.32 377.88 278.91 278.91 310.40 589.31 566.81
373.38 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0
```

12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 224.40 224.40 157.08 190.74 235.62 89.76 0.00 0.00 33.66 258.06 246.84 78.54 224.40 224.40 157.08 190.74 235.62 89.76 0.00 0.00 33.66 258.06 246.84 78.54 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 12.0 12.0 12.0 12.0 12.0 12.0 0.0 0.0 12.0 12.0 12.0 12.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 0.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 0.0 0.0 0.0 8.0 8.0 0.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 0.0 0.0 17.0 17.0 0.0 8.0 8.0 8.0 8.0 8.0 8.0 0.0 0.0 0.0 8.0 8.0 0.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 0.0 0.0 17.0 17.0 0.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 0.0 0.0 6.0 6.0 0.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 0.0 0.0 17.0 17.0 0.0 240.0 240.0 168.0 204.0 252.0 96.0 0.0 0.0 36.0 276.0 264.0 84.0 6.0 6.0 6.0 6.0 6.0 6.0 0.0 0.0 0.0 6.0 6.0 0.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 0.0 0.0 17.0 17.0 0.0 240.0 240.0 168.0 204.0 252.0 96.0 0.0 0.0 36.0 276.0 264.0 84.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 200.0 200.0 140.0 170.0 210.0 80.0 0.0 0.0 30.0 230.0 220.0 70.0 1 1 0 0 15.9 0 20.0 20.0 14.0 17.0 21.0 8.0 0.0 0.0 3.0 23.0 22.0 7.0

1 1282 A1282 1 344 200.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 1669.52 1490.64 1699.33 1565.17 1684.42 1669.52 1848.39 1848.39 1714.24 1714.24 1639.71 1714.24 1669.52 1490.64 1699.33 1565.17 1684.42 1669.52 1848.39 1848.39 1714.24 1714.24 1639.71 1714.24 1412.98 1291.29 1352.51 1307.62 1434.58 1251.47 1257.47 1257.47 1212.59 1477.78 1415.62 1270.43 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 286.20 255.54 291.31 268.32 288.76 286.20 316.87 316.87 293.87 293.87 281.09 293.87 286.20 255.54 291.31 268.32 288.76 286.20 316.87 316.87 293.87 293.87 281.09 293.87 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 14.0 12.0 14.0 14.0 12.0 14.0 14.0 14.0 14.0 12.0 12.0 14.0 6.0 6.0 6.0 6.0 6.0 6.0 7.0 7.0 6.0 6.0 6.0 6.0 8.0 8.0 8.0 8.0 8.0 8.0 9.0 9.0 9.0 8.0 8.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 16.0 16.0 16.0 17.0 17.0 16.0 8.0 8.0 8.0 8.0 8.0 8.0 9.0 9.0 9.0 8.0 8.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 16.0 16.0 16.0 17.0 17.0 16.0 6.0 6.0 6.0 6.0 6.0 6.0 7.0 7.0 7.0 6.0 6.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 16.0 16.0 16.0 17.0 17.0 16.0 270.0 240.0 278.0 254.0 272.0 276.0 310.0 310.0 286.0 276.0 264.0 284.0 6.0 6.0 6.0 6.0 6.0 6.0 7.0 7.0 7.0 6.0 6.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 16.0 16.0 16.0 17.0 17.0 16.0 270.0 240.0 278.0 254.0 272.0 276.0 310.0 310.0 286.0 276.0 264.0 284.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 224.0 200.0 228.0 210.0 226.0 224.0 248.0 248.0 230.0 230.0 220.0 230.0 1 1 1 0 12.0 0 23.0 20.0 25.0 22.0 23.0 26.0 31.0 31.0 28.0 23.0 22.0 27.0

1 1283 A1283 1 344 500.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 650.63 591.48 630.92 601.34 660.49 591.48 611.20 611.20 581.63 680.21 650.63 601.34 1759.12 1599.20 1705.81 1625.85 1785.77 1599.20 1652.50 1652.50 1572.54 1839.08 1759.12 1625.85 1328.50 1236.53 1205.19 1211.53 1357.27 1051.23 950.36 950.36 956.69 1414.81 1355.38 1053.11 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 133.88 121.70 129.82 123.73 135.90 121.70 125.76 125.76 119.68 139.96 133.88 123.73 176.47 162.27 165.65 161.60 179.85 150.10 146.72 146.72 142.66 186.61 178.50 151.45 4.16 3.78 4.03 3.84 4.22 3.78 3.91 3.91 3.72 4.35 4.16 3.84 15.0 10.0 15.0 15.0 10.0 15.0 15.0 15.0 15.0 10.0 10.0 15.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 7.0 7.0 9.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 15.0 15.0 18.0 18.0 15.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 7.0 7.0 9.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 15.0 15.0 18.0 18.0 15.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 5.0 5.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 15.0 15.0 18.0 18.0 15.0 307.0 280.0 295.0 283.0 312.0 274.0 279.0 279.0 267.0 322.0 308.0 278.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 5.0 5.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 15.0 15.0 15.0 18.0 18.0 15.0 307.0 280.0 295.0 283.0 312.0 274.0 279.0 279.0 267.0 322.0 308.0 278.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 261.0 240.0 245.0 239.0 266.0 222.0 217.0 217.0 211.0 276.0 264.0 224.0 1 1 2 1 7.6 -1 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

1 1284 A1284 1 344 400.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 1469.59 1296.70 1541.63 1390.35 1476.80 1556.04 1786.57 1786.57 1635.28 1491.21 1426.37 1606.47 1006.55 888.14 1055.90 952.28 1011.49 1065.76 1223.65 1223.65 1120.04 1021.36 976.95 1100.30 2954.68 2941.04 2105.09 2557.16 3109.91 1255.51 122.73 122.73 561.16 3365.83 3224.24 1154.82 22.33 22.22 15.91 19.32 23.50 9.49 0.93 0.93 4.24 25.43 24.36 8.73 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 199.69 176.20 209.48 188.92 200.67 211.44 242.76 242.76 222.20 202.63 193.82 218.29 258.42 234.93 250.59 238.84 262.34 234.93 242.76 242.76 231.01 270.17 258.42 238.84 5.66 5.00 5.94 5.36 5.69 5.99 6.88 6.88 6.30 5.74 5.49 6.19 14.0 10.0 14.0 14.0 10.0 14.0 14.0 14.0 14.0 10.0 10.0 14.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 7.0 7.0 9.0 18.0 18.0 18.0 18.0 18.0 18.0 16.0 16.0 16.0 18.0 18.0 16.0 7.0 7.0 7.0 7.0 7.0 7.0 9.0 9.0 9.0 7.0 7.0 9.0 18.0 18.0 18.0 18.0 18.0 18.0 16.0 16.0 16.0 18.0 18.0 16.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 5.0 5.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 16.0 16.0 16.0 18.0 18.0 16.0 310.0 280.0 306.0 288.0 314.0 292.0 310.0 310.0 292.0 322.0 308.0 298.0 5.0 5.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 5.0 5.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 16.0 16.0 16.0 18.0 18.0 16.0 310.0 280.0 306.0 288.0 314.0 292.0 310.0 310.0 292.0 322.0 308.0 298.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 264.0 240.0 256.0 244.0 268.0 240.0 248.0 248.0 236.0 276.0 264.0 244.0 1 1 0 1 17.6 -1 23.0 20.0 25.0 22.0 23.0 26.0 31.0 31.0 28.0 23.0 22.0 27.0

1 1285 A1285 1 344 600.00 5.00 0 1 0 1 2 0 0 2 2 -1 -1 -1 922.25 833.00 922.25 892.50 922.25 892.50 922.25 922.25 892.50 922.25 892.50 922.25 922.25 833.00 922.25 892.50 922.25 892.50 922.25 922.25 892.50 922.25 892.50 922.25 652.53 589.89 652.53 622.41 648.57 630.33 652.53 652.53 626.37 656.49 634.29 640.65 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 131.75 119.00 131.75 127.50 131.75 127.50 131.75 131.75 127.50 131.75 127.50 131.75 148.22 133.88 148.22 143.44 148.22 143.44 148.22 148.22 143.44 148.22 143.44 148.22 0.69 0.62 0.69 0.67 0.69 0.67 0.69 0.69 0.67 0.69 0.67 0.69 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 1 2 2 1 5.8 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1286 A1286 1 344 600.00 5.00 0 1 0 1 2 0 0 2 2 -1 -1 -1 922.25 833.00 922.25 892.50 922.25 892.50 922.25 922.25 892.50 922.25 892.50 922.25 922.25 833.00 922.25 892.50 922.25 892.50 922.25 922.25 892.50 922.25 892.50 922.25 7440.00 6720.00 7440.00 7200.00 7440.00 7200.00 7440.00 7440.00 7200.00 7440.00 7200.00 7440.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 131.75 119.00 131.75 127.50 131.75 127.50 131.75 131.75 127.50 131.75 127.50 131.75 148.22 133.88 148.22 143.44 148.22 143.44 148.22 148.22 143.44 148.22 143.44 148.22 0.69 0.62 0.69 0.67 0.69 0.67 0.69 0.69 0.67 0.69 0.67 0.69 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 1 2 2 1 13.7 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1287 A1287 1 344 600.00 5.00 0 1 0 1 2 0 0 2 2 -1 -1 -1 922.25 833.00 922.25 892.50 922.25 892.50 922.25 922.25 892.50 922.25 892.50 922.25 922.25 833.00 922.25 892.50 922.25 892.50 922.25 922.25 892.50 922.25 892.50 922.25 4897.10 4427.03 4897.10 4671.06 4867.38 4730.50 4897.10 4897.10 4700.78 4926.82 4760.23 4807.94 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 19.0 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 17.50 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 131.75 119.00 131.75 127.50 131.75 127.50 131.75 131.75 127.50 131.75 127.50 131.75 148.22 133.88 148.22 143.44 148.22 143.44 148.22 148.22 143.44 148.22 143.44 148.22 0.61 0.55 0.61 0.59 0.61 0.59 0.61 0.61 0.59 0.61 0.59 0.61 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 1 2 2 1 19.3 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1288 A1288 1 344 150.00 0.00 0 0 0 0 1 0 0 2 1 -1 -1 -1 1298.88 1180.80 1298.88 1121.76 1239.84 1239.84 1298.88 1298.88 1180.80 1357.92 1298.88 1121.76 1298.88 1180.80 1298.88 1121.76 1239.84 1239.84 1298.88 1298.88 1180.80 1357.92 1298.88 1121.76 346.17 312.82 346.17 332.33 345.01 334.67 346.17 346.17 333.50 347.34 335.83 342.67 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 259.78 236.16 259.78 224.35 247.97 247.97 259.78 259.78 236.16 271.58 259.78 224.35 478.53 435.03 478.53 413.28 456.78 456.78 478.53 478.53 435.03 500.29 478.53 413.28 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 48.0 48.0 48.0 48.0 48.0 48.0 48.0 48.0 48.0 48.0 48.0 48.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0

```
22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 374.0 340.0 374.0 323.0 357.0 357.0 374.0 374.0 340.0 391.0 374.0 323.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 374.0
340.0 374.0 323.0 357.0 357.0 374.0 374.0 340.0 391.0 374.0 323.0 1 1 1 2 2 2 1 1 2 1 1 2 1 1 1 2 2 2 1 1 2 1
1 2 154.0 140.0 154.0 133.0 147.0 147.0 154.0 154.0 140.0 161.0 154.0 133.0 1 1 0 0 17.0 0 22.0 20.0 22.0
19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1289 A1289 1 344 100.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 5267.89 4758.10 5267.89 5097.96 5267.89
5097.96 5267.89 5267.89 5097.96 5267.89 5097.96 5267.89 2110.11 1905.90 2110.11 2042.04 2110.11
2042.04 2110.11 2110.11 2042.04 2110.11 2042.04 2110.11 1364.00 1232.00 1364.00 1320.00 1364.00
1320.00 1364.00 1364.00 1320.00 1364.00 1320.00 1364.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 14.00 10.0 10.0 10.0
10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 527.00 476.00 527.00 510.00 527.00 510.00 527.00 527.00
510.00 527.00 510.00 527.00 592.88 535.50 592.88 573.75 592.88 573.75 592.88 592.88 573.75 592.88
573.75 592.88 3.72 3.36 3.72 3.60 3.72 3.60 3.72 3.72 3.60 3.72 3.60 3.72 13.0 13.0 13.0 13.0 13.0 13.0 13.0
13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0
341.0 330.0 341.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 3 1 0 0
26.2 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1290 A1290 1 344 100.00 0.00 0 1 0 1 2 0 0 2 1 -1 -1 -1 1086.09 980.98 1086.09 1051.05 1086.09 1051.05
1086.09 1086.09 1051.05 1086.09 1051.05 1086.09 1086.09 980.98 1086.09 1051.05 1086.09 1051.05
1086.09 1086.09 1051.05 1086.09 1051.05 1086.09 74.40 67.20 74.40 72.00 74.40 72.00 74.40 74.40 72.00
74.40 72.00 74.40 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 155.16 140.14 155.16 150.15 155.16 150.15 155.16
155.16 150.15 155.16 150.15 155.16 435.74 393.57 435.74 421.68 435.74 421.68 435.74 435.74 421.68
435.74 421.68 435.74 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0
744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 3 0 0 0 4.2
0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1291 A1291 1 344 200.00 0.00 1 0 0 0 2 0 0 2 1 -1 -1 -1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0
25.0 25.0 25.0 25.0 25.0 25.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 720.0 672.0 744.0 336.0 480.0 720.0 744.0 624.0 720.0 744.0 720.0 576.0 720.0
672.0 744.0 48.0 72.0 720.0 744.0 96.0 720.0 744.0 720.0 120.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 0 0 0 9.9 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0
 1 1292 A1292 1 344 200.00 0.00 0 0 0 0 2 0 0 2 1 -1 -1 -1 1312.85 1185.80 1312.85 1270.50 1312.85
1270.50 1312.85 1312.85 1270.50 1312.85 1270.50 1312.85 1312.85 1185.80 1312.85 1270.50 1312.85
1270.50 1312.85 1312.85 1270.50 1312.85 1270.50 1312.85 2395.07 2163.29 2395.07 2317.81 2395.07
2317.81 2395.07 2395.07 2317.81 2395.07 2317.81 2395.07 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 225.06 203.28 225.06 217.80 225.06 217.80 225.06 225.06
```

```
217.80 225.06 217.80 225.06 225.06 203.28 225.06 217.80 225.06 217.80 225.06 225.06 217.80 225.06
217.80 225.06 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 8.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0
744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 310.0 280.0 310.0 300.0 310.0 300.0 310.0 310.0 300.0 310.0 300.0 310.0 1 0 0 0 9.6 0 31.0 28.0
31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1293 A1293 1 344 100.00 0.00 0 1 0 0 2 0 0 2 1 -1 -1 -1 98.74 89.18 98.74 95.55 98.74 95.55 98.74 98.74
95.55 98.74 95.55 98.74 98.74 89.18 98.74 95.55 98.74 95.55 98.74 98.74 95.55 98.74 95.55 98.74 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 14.11
12.74 14.11 13.65 14.11 13.65 14.11 14.11 13.65 14.11 13.65 14.11 39.61 35.78 39.61 38.33 39.61 38.33
39.61 39.61 38.33 39.61 38.33 39.61 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 720.0 672.0
744.0 336.0 480.0 720.0 744.0 624.0 720.0 744.0 720.0 576.0 720.0 672.0 744.0 48.0 72.0 720.0 744.0 96.0
720.0 744.0 720.0 120.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 0 0 0 0.4 0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1294 A1294 1 344 400.00 0.00 1 0 0 0 2 0 0 2 2 -1 -1 -1 6028.67 5445.25 6028.67 5834.20 6028.67
5834.20 6028.67 6028.67 5834.20 6028.67 5834.20 6028.67 4129.16 3729.56 4129.16 3995.96 4129.16
3995.96 4129.16 4129.16 3995.96 4129.16 3995.96 4129.16 472.59 426.85 472.59 457.34 472.59 457.34
472.59 472.59 457.34 472.59 457.34 472.59 3.57 3.23 3.57 3.46 3.57 3.46 3.57 3.57 3.46 3.57 3.46 3.57 24.0
24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 825.84 745.92 825.84 799.20 825.84 799.20 825.84 825.84 799.20 825.84
799.20 825.84 825.84 745.92 825.84 799.20 825.84 799.20 825.84 825.84 799.20 825.84 799.20 825.84 22.09
19.95 22.09 21.38 22.09 21.38 22.09 22.09 21.38 22.09 21.38 22.09 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0
672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 0 0 1 8.7 0 31.0 28.0 31.0
30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1295 A1295 1 344 100.00 0.00 0 1 0 0 2 0 0 2 1 -1 -1 -1 83.97 75.84 83.97 81.26 83.97 81.26 83.97 83.97
81.26 83.97 81.26 83.97 83.97 75.84 83.97 81.26 83.97 81.26 83.97 83.97 81.26 83.97 81.26 83.97 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 12.00
10.83 12.00 11.61 12.00 11.61 12.00 12.00 11.61 12.00 11.61 12.00 15.99 14.45 15.99 15.48 15.99 15.48
15.99 15.99 15.48 15.99 15.48 15.99 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 720.0 672.0
744.0 336.0 480.0 720.0 744.0 624.0 720.0 744.0 720.0 576.0 720.0 672.0 744.0 48.0 72.0 720.0 744.0 96.0
720.0 744.0 720.0 120.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 372.0 336.0 372.0 360.0 372.0 360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 0 0 0 0.3 0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1296 A1296 1 344 100.00 0.00 0 1 0 0 2 0 0 2 1 -1 -1 -1 43.12 39.20 43.12 37.24 41.16 41.16 43.12 43.12
39.20 45.08 43.12 37.24 43.12 39.20 43.12 37.24 41.16 41.16 43.12 43.12 39.20 45.08 43.12 37.24 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 27.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.00
```

```
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 6.16 5.60
6.16 5.32 5.88 5.88 6.16 6.16 5.60 6.44 6.16 5.32 9.08 8.25 9.08 7.84 8.67 8.67 9.08 9.08 8.25 9.49 9.08 7.84
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 720.0 672.0 744.0 336.0 480.0 720.0 744.0 624.0
720.0 744.0 720.0 576.0 720.0 672.0 744.0 48.0 72.0 720.0 744.0 96.0 720.0 744.0 720.0 120.0 6.0 6.0 6.0 6.0
6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 6.0 6.0 6.0 6.0 6.0
6.0 6.0 6.0 6.0 6.0 6.0 6.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 14.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 154.0 140.0 154.0 133.0 147.0
147.0 154.0 154.0 140.0 161.0 154.0 133.0 1 0 0 0 0.4 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
 1 1297 A1297 1 344 50.00 22.00 0 1 0 1 2 0 0 2 2 -1 -1 -1 533.61 485.10 533.61 460.84 509.36 509.36
533.61 533.61 485.10 557.86 533.61 460.84 533.61 485.10 533.61 460.84 509.36 509.36 533.61 533.61
485.10 557.86 533.61 460.84 629.27 568.37 629.27 608.97 629.27 608.97 629.27 629.27 608.97 629.27
608.97 629.27 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0
25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00
16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0
76.23 69.30 76.23 65.83 72.76 72.76 76.23 76.23 69.30 79.69 76.23 65.83 76.23 69.30 76.23 65.83 72.76
72.76 76.23 76.23 69.30 79.69 76.23 65.83 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 62.0
62.0 62.0 62.0 62.0 62.0 62.0 62.0 62.0 62.0 62.0 62.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 14.0 7.0 7.0 7.0 9.0 9.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0
190.0 210.0 210.0 220.0 220.0 200.0 230.0 220.0 190.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 16.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 220.0 200.0 220.0 190.0 210.0 210.0 220.0 220.0 200.0
230.0 220.0 190.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 176.0 160.0 176.0 152.0 168.0 168.0 176.0
176.0 160.0 184.0 176.0 152.0 1 1 0 0 4.9 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0 22.0 20.0 23.0 22.0 19.0
 1 1298 A1298 1 344 200.00 22.00 0 1 0 1 2 0 0 2 2 -1 -1 -1 15.25 13.86 15.25 13.17 14.55 14.55 15.25 15.25
13.86 15.94 15.25 13.17 15.25 13.86 15.25 13.17 14.55 14.55 15.25 15.25 13.86 15.94 15.25 13.17 366.48
332.73 366.48 324.24 353.20 350.80 366.48 366.48 337.52 379.76 364.08 326.64 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00
16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 2.18 1.98 2.18 1.88 2.08 2.08 2.18
2.18 1.98 2.28 2.18 1.88 2.45 2.23 2.45 2.12 2.34 2.34 2.45 2.45 2.23 2.56 2.45 2.12 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 0.00 0.00 0.00 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0
6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0 15.0
15.0 15.0 15.0 15.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0
17.0 17.0 17.0 242.0 220.0 242.0 209.0 231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 7.0 7.0 7.0 7.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 242.0 220.0 242.0 209.0
231.0 231.0 242.0 242.0 220.0 253.0 242.0 209.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 198.0 180.0
198.0 171.0 189.0 189.0 198.0 198.0 180.0 207.0 198.0 171.0 1 0 0 0 1.8 0 22.0 20.0 22.0 19.0 21.0 21.0 22.0
22.0 20.0 23.0 22.0 19.0
 1 1301 A1301 1 344 100.00 0.00 0 1 0 0 2 0 0 2 2 -1 -1 -1 3277.73 2960.53 3277.73 3172.00 3277.73
3172.00 3277.73 3277.73 3172.00 3277.73 3172.00 3277.73 3277.73 2960.53 3277.73 3172.00 3277.73
3172.00 3277.73 3277.73 3172.00 3277.73 3172.00 3277.73 2246.85 2029.42 2246.85 2174.37 2246.85
2174.37 2246.85 2246.85 2174.37 2246.85 2174.37 2246.85 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 468.25 422.93 468.25 453.14 468.25 453.14 468.25 468.25
453.14 468.25 453.14 468.25 468.25 422.93 468.25 453.14 468.25 453.14 468.25 468.25 453.14 468.25
453.14 468.25 4960.00 4480.00 4960.00 4800.00 4960.00 4800.00 4960.00 4960.00 4800.00 4960.00 4800.00
4960.00 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 10.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0
3.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 7.0
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0
392.0 434.0 420.0 434.0 420.0 434.0 434.0 420.0 434.0 420.0 434.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
7.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 434.0 392.0 434.0 420.0 434.0 420.0 434.0
434.0 420.0 434.0 420.0 434.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 372.0 336.0 372.0 360.0 372.0
360.0 372.0 372.0 360.0 372.0 360.0 372.0 1 0 1 0 14.6 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0
30.0 31.0
 1 1302 A1302 1 344 100.00 0.00 1 0 0 0 2 0 1 2 2 -1 -1 -1 2038.50 1841.22 2038.50 1972.74 2038.50
```

```
1972.74 2038.50 2038.50 1972.74 2038.50 1972.74 2038.50 1303.30 1177.18 1303.30 1261.26 1303.30
1261.26 1303.30 1303.30 1261.26 1303.30 1261.26 1303.30 37200.00 33600.00 37200.00 36000.00 37200.00
36000.00 37200.00 37200.00 36000.00 37200.00 36000.00 37200.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 238.70 215.60 238.70 231.00 238.70 231.00 238.70
238.70 231.00 238.70 231.00 238.70 238.70 215.60 238.70 231.00 238.70 231.00 238.70 238.70 231.00
238.70 231.00 238.70 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 16.0
16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0
744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 248.0 224.0 248.0 240.0 248.0 240.0 248.0 248.0 240.0 248.0 240.0 248.0 1 0 0 0
58.2 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1303 A1303 1 344 300.00 0.00 1 1 0 0 3 1 1 3 1 -1 -1 -1 1953.00 1764.00 1953.00 1890.00 1953.00
1890.00 1953.00 1953.00 1890.00 1953.00 1890.00 1953.00 2929.50 2646.00 2929.50 2835.00 2929.50
2835.00 2929.50 2929.50 2835.00 2929.50 2835.00 2929.50 3720.00 3360.00 3720.00 3600.00 3720.00
3600.00 3720.00 3720.00 3600.00 3720.00 3600.00 3720.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 25.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0 18.0
18.0 18.0 18.0 18.0 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 15.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 271.25 245.00 271.25 262.50 271.25 262.50 271.25 271.25
262.50 271.25 262.50 271.25 372.00 336.00 372.00 360.00 372.00 360.00 372.00 372.00 360.00 372.00
360.00 372.00 6.78 6.13 6.78 6.56 6.78 6.56 6.78 6.78 6.56 6.78 6.56 6.78 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0
744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0
744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0 744.0 3 2 2 1 8.6 -1 31.0 28.0
31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1304 A1304 1 344 400.00 0.00 1 0 0 0 3 0 1 3 2 -1 -1 -1 3767.92 3403.28 3767.92 3646.37 3767.92
3646.37 3767.92 3767.92 3646.37 3767.92 3646.37 3767.92 2580.72 2330.98 2580.72 2497.48 2580.72
2497.48 2580.72 2580.72 2497.48 2580.72 2497.48 2580.72 6466.56 5840.76 6466.56 6257.96 6466.56
6257.96 6466.56 6466.56 6257.96 6466.56 6257.96 6466.56 48.87 44.14 48.87 47.29 48.87 47.29 48.87 48.87
47.29 48.87 47.29 48.87 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 24.0 22.0 22.0 22.0 22.0 22.0
22.0 22.0 22.0 22.0 22.0 22.0 22.0 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00 17.00
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 516.15 466.20 516.15 499.50 516.15 499.50
516.15 516.15 499.50 516.15 499.50 516.15 707.86 639.36 707.86 685.03 707.86 685.03 707.86 707.86
685.03 707.86 685.03 707.86 11.84 10.70 11.84 11.46 11.84 11.46 11.84 11.84 11.46 11.84 11.46 11.84 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0
720.0 744.0 744.0 720.0 744.0 720.0 744.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 23.0 23.0 23.0 23.0
23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0 744.0 720.0
744.0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 744.0 672.0 744.0 720.0 744.0 720.0 744.0 744.0 720.0
744.0 720.0 744.0 1 1 0 1 18.9 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0
 1 1305 A1305 1 344 600.00 10.00 0 1 0 1 2 0 0 2 2 -1 -1 -1 2028.95 1832.60 2028.95 1963.50 2028.95
1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1832.60 2028.95 1963.50 2028.95
1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 1631.32 1474.73 1631.32 1556.02 1621.42
1575.82 1631.32 1631.32 1565.92 1641.22 1585.73 1601.62 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0
20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0
12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 289.85 261.80 289.85 280.50 289.85 280.50 289.85 289.85
280.50 289.85 280.50 289.85 326.08 294.52 326.08 315.56 326.08 315.56 326.08 326.08 315.56 326.08
315.56 326.08 2.48 2.24 2.48 2.40 2.48 2.40 2.48 2.48 2.40 2.48 2.40 2.48 13.0 13.0 13.0 13.0 13.0 13.0 13.0
13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0
```

17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 1 0 2 1 13.4 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1306 A1306 1 344 600.00 10.00 0 1 0 1 2 0 0 2 2 -1 -1 -1 2028.95 1832.60 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1832.60 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 18600.00 16800.00 18600.00 18000.00 18600.00 18000.00 18600.00 18600.00 18000.00 18600.00 18000.00 18600.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 289.85 261.80 289.85 280.50 289.85 280.50 289.85 289.85 280.50 289.85 280.50 289.85 326.08 294.52 326.08 315.56 326.08 315.56 326.08 326.08 315.56 326.08 315.56 326.08 2.48 2.24 2.48 2.40 2.48 2.40 2.48 2.48 2.40 2.48 2.40 2.48 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 1 0 2 1 33.2 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1307 A1307 1 344 600.00 10.00 0 1 0 1 2 0 0 2 2 -1 -1 -1 2028.95 1832.60 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1832.60 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 16325.77 14758.64 16325.77 15572.21 16226.69 15770.37 16325.77 16325.77 15671.29 16424.85 15869.45 16028.53 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 289.85 261.80 289.85 280.50 289.85 280.50 289.85 289.85 280.50 289.85 280.50 289.85 326.08 294.52 326.08 315.56 326.08 315.56 326.08 326.08 315.56 326.08 315.56 326.08 2.48 2.24 2.48 2.40 2.48 2.40 2.48 2.48 2.40 2.48 2.40 2.48 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 1 0 2 1 60.2 0 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1308 A1308 1 344 600.00 10.00 0 1 0 1 2 0 0 2 2 -1 -1 -1 2028.95 1832.60 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1832.60 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 1631.32 1474.73 1631.32 1556.02 1621.42 1575.82 1631.32 1631.32 1565.92 1641.22 1585.73 1601.62 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 289.85 261.80 289.85 280.50 289.85 280.50 289.85 289.85 280.50 289.85 280.50 289.85 326.08 294.52 326.08 315.56 326.08 315.56 326.08 326.08 315.56 326.08 315.56 326.08 2.48 2.24 2.48 2.40 2.48 2.40 2.48 2.48 2.40 2.48 2.40 2.48 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 1 2 2 1 13.4 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1309 A1309 1 344 600.00 10.00 0 1 0 1 2 0 0 2 2 -1 -1 -1 2028.95 1832.60 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1832.60 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 18600.00 16800.00 18600.00 18000.00 18600.00 18000.00 18600.00 18600.00 18000.00 18600.00 18000.00 18600.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00

0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 289.85 261.80 289.85 280.50 289.85 280.50 289.85 289.85 280.50 289.85 280.50 289.85 326.08 294.52 326.08 315.56 326.08 315.56 326.08 326.08 315.56 326.08 315.56 326.08 2.48 2.24 2.48 2.40 2.48 2.40 2.48 2.48 2.40 2.48 2.40 2.48 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 1 2 2 1 33.2 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0

1 1310 A1310 1 344 600.00 10.00 0 1 0 1 2 0 0 2 2 -1 -1 -1 2028.95 1832.60 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1832.60 2028.95 1963.50 2028.95 1963.50 2028.95 2028.95 1963.50 2028.95 1963.50 2028.95 16325.77 14758.64 16325.77 15572.21 16226.69 15770.37 16325.77 16325.77 15671.29 16424.85 15869.45 16028.53 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 23.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 20.0 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 12.0 289.85 261.80 289.85 280.50 289.85 280.50 289.85 289.85 280.50 289.85 280.50 289.85 326.08 294.52 326.08 315.56 326.08 315.56 326.08 326.08 315.56 326.08 315.56 326.08 2.48 2.24 2.48 2.40 2.48 2.40 2.48 2.48 2.40 2.48 2.40 2.48 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 13.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 6.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 9.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 17.0 341.0 308.0 341.0 330.0 341.0 330.0 341.0 341.0 330.0 341.0 330.0 341.0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 279.0 252.0 279.0 270.0 279.0 270.0 279.0 279.0 270.0 279.0 270.0 279.0 1 2 2 1 60.2 -1 31.0 28.0 31.0 30.0 31.0 30.0 31.0 31.0 30.0 31.0 30.0 31.0