SECTION C

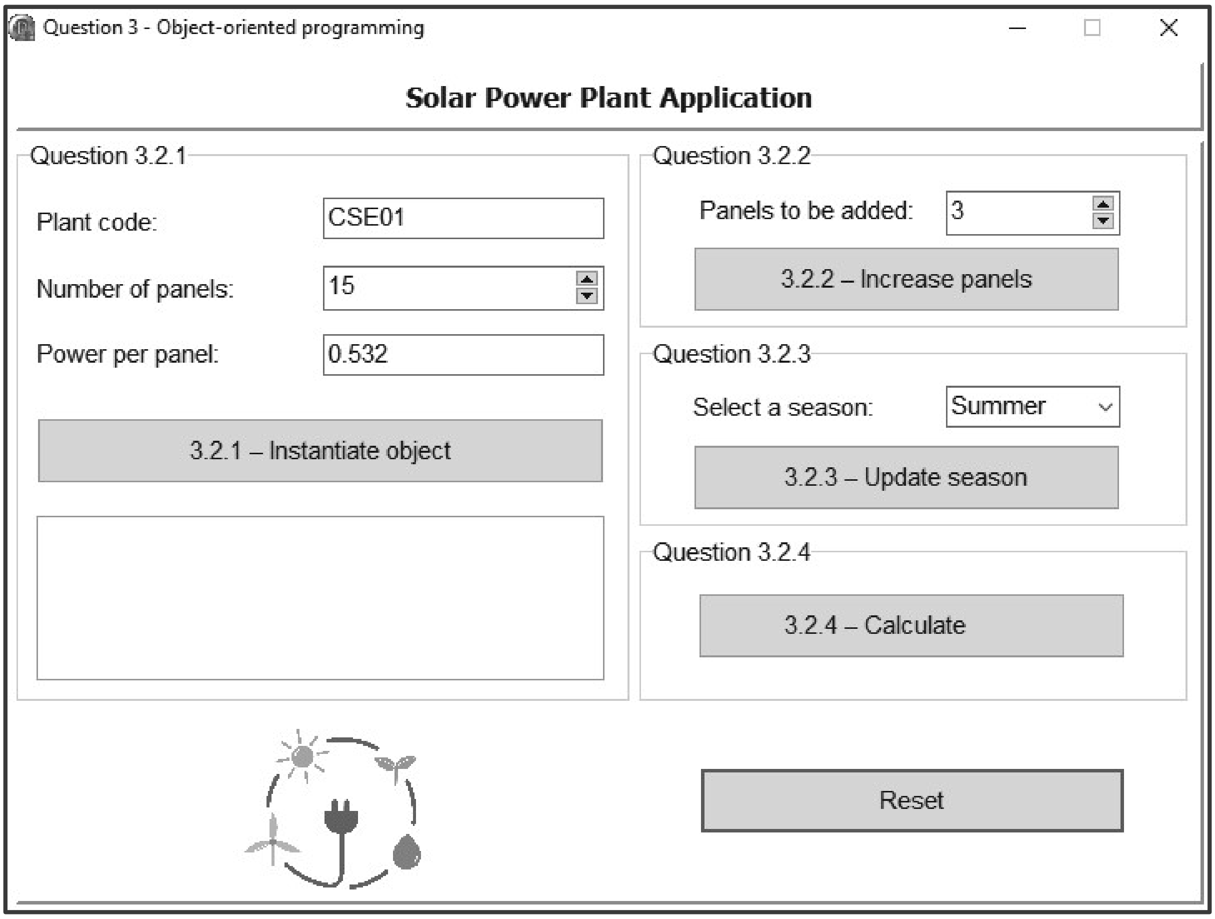
QUESTION 3: OBJECT-ORIENTATED PROGRAMMING

The sustainable energy sector wants to keep track of the solar power plants in the country and their capacity to increase the power that they generate.

Do the following:

- Open the incomplete program in the **Question 3** folder.
- Open the incomplete object class **SolarPowerPlant_U.pas**.
- Enter your examination number as a comment in the first line of both the **Question3_U.pas** file and the **SolarPowerPlant_U.pas** file.
- Compile and execute the program. The program has limited functionality currently.

Example of graphical user interface (GUI):



Question 3 - Object-oriented programming

Solar Power Plant Application

Question 3.2.1

Plant code: CSE01

Number of panels: 15

Power per panel: 0.532

Question 3.2.2

Panels to be added: 3

Question 3.2.3

Select a season: Summer

Question 3.2.4

- Complete the code as specified in QUESTION 3.1 and QUESTION 3.2 that follow.

NOTE: You are NOT allowed to add any additional attributes or user-defined methods, unless explicitly stated in the question.

3.1 The provided incomplete object class (**TSolarPowerPlant**) contains the declaration of four attributes which describe a **SolarPowerPlant** object.

The attributes for a **SolarPowerPlant** object have been declared as follows:

Attribute	Description
fPlantCode	A unique code for the solar power plant
fNumberOfPanels	The number of panels installed at the plant
fPowerPerPanel	The maximum power per panel in kilowatt-hour (kWh)
fSeason	The season during which power will be generated – Summer, Autumn, Winter, Spring

Code has been provided for the following accessor methods:

- **getPlantCode** to return the fPlantCode attribute
- **getNumOfPanels** to return the fNumberOfPanels attribute
- **getSeason** to return the fSeason attribute

Complete the code in the object class as described in QUESTION 3.1.1 to QUESTION 3.1.5 below.

3.1.1 Write code for a **constructor** method that will receive the plant code, the number of panels and the power per panel as parameters. Assign the parameter values to the respective attributes. Assign the default value 'Summer' to the season attribute. (5)

3.1.2 Write the code for a method called **incNumOfPanels** that receives a value as parameter and increments the **fNumberOfPanels** attribute by the value received. (4)

3.1.3 Write code for a mutator method called **setSeason** that receives a value as a parameter and sets the **fSeason** attribute to the value received. (3)

3.1.4 Write a method called **calculateCapacity** that uses the information that follows to calculate the power generation capacity of the installed panels at the company in the current season. Return the result as a real value.

The season determines the hours of sunlight per day that can be used to generate solar power, as shown in the following table:

SEASON	HOURS OF SUNLIGHT PER DAY
Summer	10
Autumn	8
Winter	6
Spring	8

The formula to calculate the generation capacity (GC) of the solar power plant is as follows:

$$GC = NumberOfPanels \times \ PowerPerPanel \ \times HoursPerDay$$ (8)

3.1.5 Write a **toString** method to return a string with all the attributes of the object in the following format:

```
Plant code: <PlantCode>
Number of panels: <NumberOfPanels>
Power per panel: <PowerPerPanel>
Season: <Season>
```

(3)

3.2 An incomplete program has been supplied in the **Question 3** folder. The program contains code for the object class to be accessible and declares an object variable called **objPlant**.

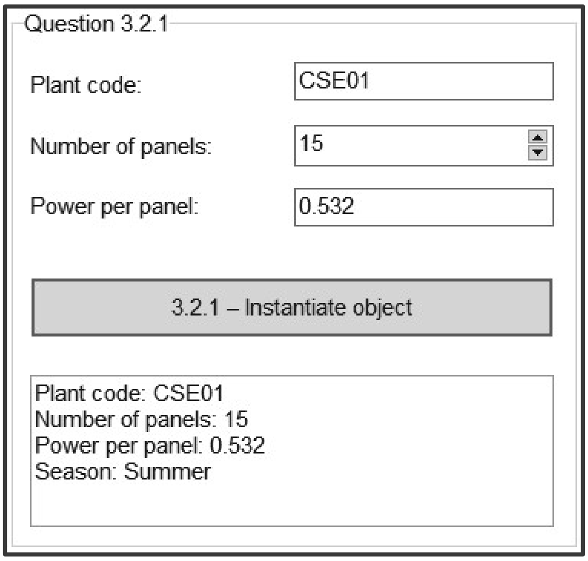
Write code to perform the tasks described in QUESTION 3.2.1 to QUESTION 3.2.4 below.

3.2.1 **Button [3.2.1 - Instantiate object]**

Write code to do the following:

- Extract the plant code from the edit box **edtQ3_2_1_Code**, the number of panels from the spin edit **sedQ3_2_1** and the power per panel from the edit box **edtQ3_2_1_Power**.
- Use the information to instantiate a new **SolarPowerPlant** object.
- Use the **toString** method to display the information of the **SolarPowerPlant** object in the rich edit **redQ3**.

Example of input and output:



Plant code: CSE01
Number of panels: 15
Power per panel: 0.532

```
Plant code: CSE01
Number of panels: 15
Power per panel: 0.532
Season: Summer
```

(6)

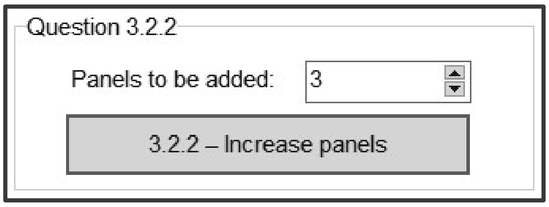
3.2.2 **Button [3.2.2 - Increase panels]**

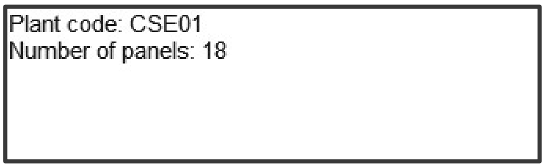
As the need for more power arises, panels can be added to increase the plant's power generation capacity.

Write code to do the following:

- Extract the value of the number of panels to be added from the spin edit **sedQ3_2_2**.
- Call the **incNumOfPanels** method with the value from the spin edit as argument.
- Call the relevant object methods to display the following in the rich edit **redQ3**:
  - o The plant code
  - o The number of panels after increasing the attribute value

Example of input and output:




Plant code: CSE01
Number of panels: 18

(4)

3.2.3 **Button [3.2.3 - Update season]**

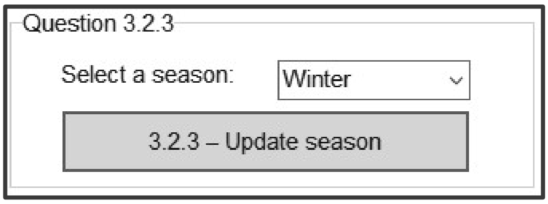
The number of hours for the generation of power differs per season.

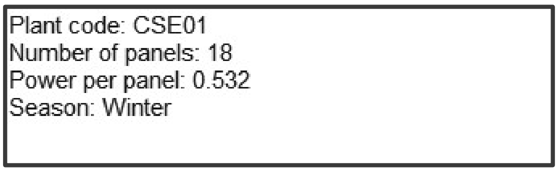
The user must select the season from the combo box **cmbQ3_2_3**.

Write code to do the following:

- Extract the season selected from the combo box **cmbQ3_2_3**.
- Call the **setSeason** method using the value from the combo box as an argument.
- Use the **toString** method to display the information of the updated **Plant** object in the rich edit **redQ3**.

Example of input and output:





```
Plant code: CSE01
Number of panels: 18
Power per panel: 0.532
Season: Winter
```

(3)

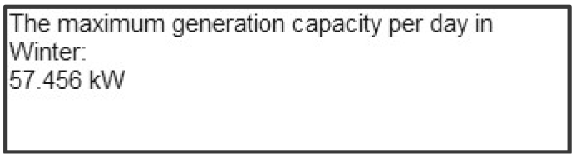
3.2.4 **Button [3.2.4 - Calculate]**

The maximum capacity of power that can be generated by the panels must be calculated.

Write code to display the following in the rich edit **redQ3**:

- The season as a part of the output string.
- The result of the **calculateCapacity** method converted into a string. The unit 'kW' must also be displayed.

Example of output:



```
The maximum generation capacity per day in
Winter:
57.456 kW
```

(4)

- Enter your examination number as a comment in the first line of the object class and the form class.
- Save your program.
- Print the code in the object class and the form class if required.

TOTAL SECTION C: 40