



basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

SENIOR CERTIFICATE EXAMINATIONS/ NATIONAL SENIOR CERTIFICATE EXAMINATIONS

INFORMATION TECHNOLOGY P1

2023

MARKS: 150

TIME: 3 hours



**This question paper consists of 24 pages, 2 data pages and
2 blank pages for planning.**

INSTRUCTIONS AND INFORMATION

1. This question paper is divided into FOUR sections. Candidates must answer ALL the questions from all FOUR sections.
2. Two blank pages have been provided at the end of the question paper which may be used for planning purposes.
3. The duration of this examination is three hours. Because of the nature of this examination, it is important to note that you will not be permitted to leave the examination room before the end of the examination session.
4. This question paper is set with programming terms that are specific to Delphi programming language. The Delphi programming language must be used to answer the questions.
5. Make sure that you answer the questions according to the specifications that are given in each question. Marks will be awarded according to the set requirements.
6. Answer only what is asked in each question. For example, if the question does not ask for data validation, no marks will be awarded for data validation.
7. Your programs must be coded in such a way that they will work with any data and not just the sample data supplied or any data extracts that appear in the question paper.
8. Routines, such as search, sort and selection, must be developed from first principles. You may NOT use the built-in features of the Delphi programming language for any of these routines.
9. All data structures must be defined by you, the programmer, unless the data structures are supplied.
10. You must save your work regularly on the disk/CD/DVD/flash disk you have been given, or on the disk space allocated to you for this examination session.
11. Make sure that your examination number appears as a comment in every program that you code, as well as on every event indicated.
12. If required, print the programming code of all the programs/classes that you completed. Your examination number must appear on all the printouts. You will be given half an hour printing time after the examination session.
13. At the end of this examination session, you must hand in a disk/CD/DVD/flash disk with all your work saved on it OR you must make sure that all your work has been saved on the disk space allocated to you for this examination session. Ensure that all files can be read.

14. The files that you need to complete this question paper have been provided to you on the disk/CD/DVD/flash disk or on the disk space allocated to you. The files are provided in the form of password-protected executable files.

Do the following:

- Double click on the following password-protected executable file:
DataJun2023.exe.
- Click on the 'Extract' button.
- Enter the following password: **E#Homes%2023**

Once extracted, the following list of files will be available in the folder **DataJun2023**:

Question1:

Question1_P.dpr
Question1_P.dproj
Question1_P.res
Question1_U.dfm
Question1_U.pas
Words.txt

Question2:

ConnectDB_U.pas
Question2_P.dpr
Question2_P.dproj
Question2_P.res
Question2_U.dfm
Question2_U.pas
SmartDevicesDB - Copy.mdb
SmartDevicesDB.mdb

Question3:

log.txt
Question3_P.dpr
Question3_P.dproj
Question3_P.res
Question3_U.dfm
Question3_U.pas
SmartSwitch_U.pas

Question4:

Question4_P.dpr
Question4_P.dproj
Question4_P.res
Question4_U.dfm
Question4_U.pas



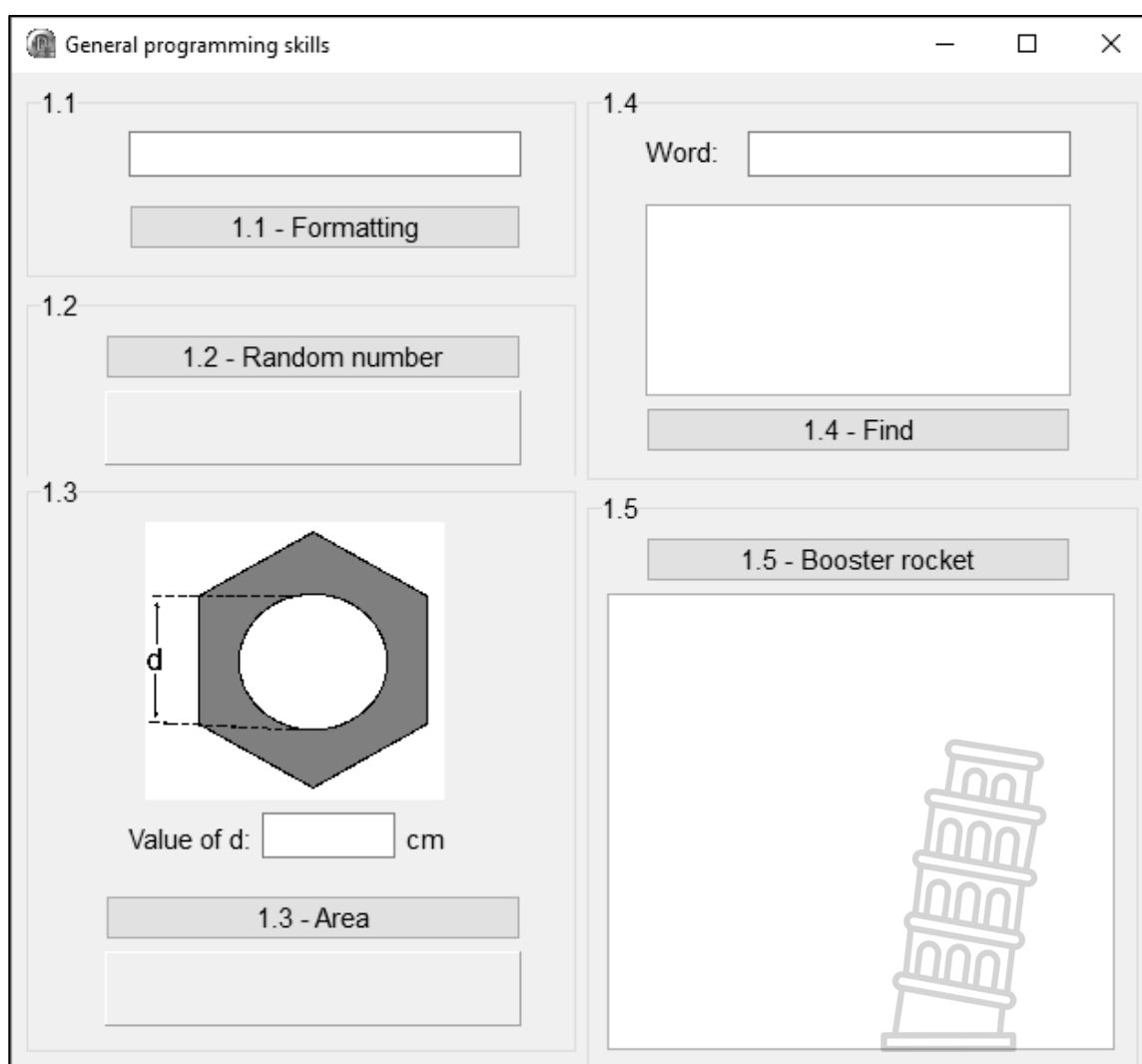
SECTION A

QUESTION 1: GENERAL PROGRAMMING SKILLS

Do the following:

- Open the incomplete program in the **Question 1** folder.
- Enter your examination number as a comment in the first line of the **Question1_U.pas** file.
- Compile and execute the program. The program has no functionality currently.

Example of graphical user interface (GUI):



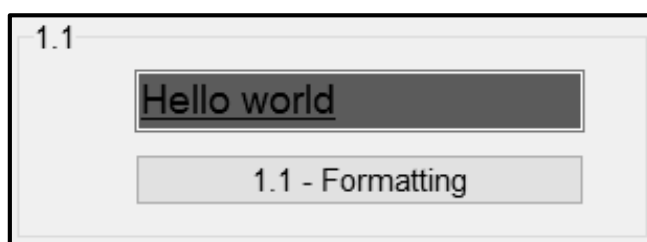
- Complete the code for each section of QUESTION 1, as described in QUESTION 1.1 to QUESTION 1.5 that follow.

1.1 Button [1.1 - Formatting]

Write code to change the content and appearance of the edit box **edtQ1_1** as follows when the **btnQ1_1** button is clicked:

- The font size of the text must be 14 pt.
- The words 'Hello world' must appear as text in the **edtQ1_1** edit box.
- The text must be underlined.
- The background colour of the **edtQ1_1** component must be green.

Example of output:



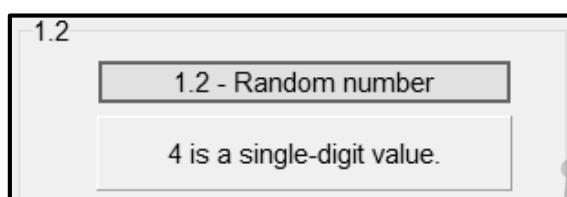
(4)

1.2 Button [1.2 - Random number]

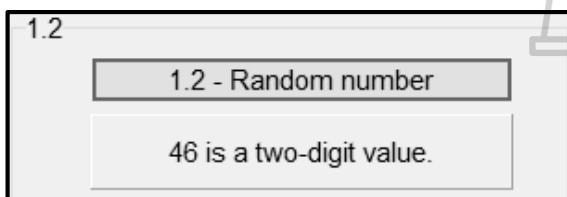
Write code to do the following:

- Generate a random integer between 1 and 99 (inclusive of these values).
- Determine whether the number that has been generated is a single digit or a two-digit number.
- Display the number and result in the panel **pnlQ1_2** as shown in the examples below.

Example of output if the number 4 was generated:



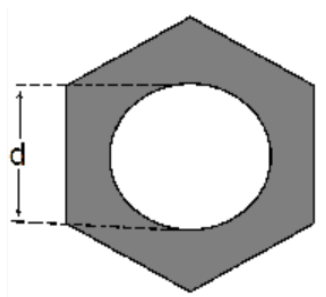
Example of output if the number 46 was generated:



(8)

1.3 Button [1.3 - Area]

Shown below is a diagram of a nut, made up of a hexagon and a circle. The diameter (d) of the circle has the same length as the side of the hexagon.



The shaded area in the diagram is calculated using the following formula:

$$\text{Shaded area} = \left(\frac{3\sqrt{3}}{2}\right) d^2 - \pi \left(\frac{d}{2}\right)^2$$

where d is the length of one side of the hexagon and d is also the diameter of the circle.

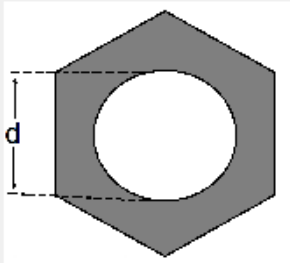
The user is required to enter the value of d in the edit box **edtQ1_3**.

Write code to do the following:

- Extract the value of d from the edit box **edtQ1_3**.
- Use the given formula to calculate the area of the shaded part of the diagram provided above.
- Display the area of the shaded part, rounded to ONE decimal place, on the panel **pnlQ1_3**.

Example of input and output:

1.3



Value of d: cm

1.3 - Area

45.3 cm squared

1.4 Button [1.4 - Find]

A text file **Words.txt** containing a list of words has been supplied. The program must count the number of times a word entered by the user occurs in the text file.

Example of the first five lines of the text file **Words.txt**:

accessory
 archive
 bandwidth
 capacity
 bus

The user must enter the word to be counted in the edit box **edtQ1_4**.

Code has been provided to open the text file **Words.txt** for reading.

Write code to do the following:

- Extract the word that was entered in the edit box **edtQ1_4**.
- Count the number of times the word entered occurs in the file, regardless of whether the word was entered in small letters or capital letters.
- If the word occurs in the text file, display the number of occurrences in the **redQ1_4** rich edit component.
- If the word does NOT occur in the text file, display the message 'Word not found' in the **redQ1_4** rich edit component.

Two examples of input and output:

The screenshot shows a window titled '1.4'. It contains a label 'Word:' followed by a text box containing 'Capacity'. Below this is a larger text area labeled 'Occurrences: 4'. At the bottom is a button labeled '1.4 - Find'.

The screenshot shows a window titled '1.4'. It contains a label 'Word:' followed by a text box containing 'school'. Below this is a larger text area labeled 'Word not found'. At the bottom is a button labeled '1.4 - Find'.

(11)

1.5 Button [1.5 - Booster rocket]

A booster rocket uses 7,5% of its fuel per second from lift-off. The rocket shuts down when the fuel capacity is below 200 litres.

Do the following:

Convert the algorithm on the next page into programming code to calculate and display the seconds travelled, the fuel used and the fuel left per second until the rocket shuts down.

Algorithm

1. Enter the total litres of fuel before lift-off (use an input box for input)
2. Assign the value of total litres of fuel to a variable for fuel left
3. Initialise a counter for number of seconds
4. While fuel left is more than or equal to 200
5. Increment counter by one
6. Calculate fuel used (7,5% of fuel left)
7. Subtract fuel used from fuel left
8. Display counter, fuel used and fuel left
9. End while loop

NOTE:

- Code is provided to display the headings in the rich edit **redQ1_5**.
- The amount of fuel used per second and the amount of fuel left must be formatted to TWO decimal places, as shown in the example of output.

Example of the input of 550 litres of fuel and the output:

Fuel ✕

Total litres of fuel:

550

OK

Cancel

1.5 - Booster rocket

Seconds	Fuel used	Fuel left
1	41.25	508.75
2	38.16	470.59
3	35.29	435.30
4	32.65	402.65
5	30.20	372.45
6	27.93	344.52
7	25.84	318.68
8	23.90	294.78
9	22.11	272.67
10	20.45	252.22
11	18.92	233.30
12	17.50	215.81
13	16.19	199.62

(9)

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

TOTAL SECTION A: 40

SECTION B

QUESTION 2: DATABASE PROGRAMMING

Smart home devices are becoming more popular among people that have access to fast reliable internet connections.

A database called **SmartDevicesDB.mdb** has been developed and contains information about the different manufacturers of smart devices and the stock available of the different devices.

The **SmartDevicesDB.mdb** database contains two tables called **tblManufacturers** and **tblDevices**.

The data pages attached at the end of the question paper provide information on the design and contents of the database.

Do the following:

- Open the incomplete project file called **Question2_P.dpr** in the **Question 2** folder.
- Enter your examination number as a comment in the first line of the **Question2_U.pas** unit file.
- Compile and execute the program. The program has no functionality currently. The contents of the tables are displayed as shown below on the selection of tab sheet **2.2 - Delphi code**.

Database programming

2.1 - SQL 2.2 - Delphi code

ManufacturerID	ManufacturerName	ContactNumber	OnlineSupport
▶ M100	Berry Tech	0122035878	True
M101	Gem Smart	0169725478	False
M102	Gemini	0512025306	False
M103	Jupiter Inc.	0311257412	True

DeviceID	DeviceName	Category	OperatingSystem	Price	NumInStock	ManufacturerID
▶ D001	Always Summer	Smart heating	Android	R300.00	107	M105
D002	Boom XL	Smart speaker	iOS	R1 500.00	80	M104
D003	Boom XXL	Smart speaker	iOS	R2 000.00	50	M104
D004	Bulb 6	Smart lighting	Android	R500.00	100	M101

2.2.1

2.2.1 - Display products

2.2.2

2.2.2 - Update stock

Restore database Close

- Follow the instructions below to complete the code for each section as described in QUESTION 2.1 and QUESTION 2.2.
- Use SQL statements to answer QUESTION 2.1 and Delphi code to answer QUESTION 2.2.

NOTE:

- The 'Restore database' button is provided to restore the data contained in the database to the original content.
- The content of the database is password-protected, i.e. you will NOT be able to gain direct access to the content of the database using Microsoft Access.
- Code is provided to link the GUI components to the database. Do NOT change any of the code provided.
- TWO variables are declared as public variables, as described in the table below:

Variable	Data type	Description
tblManufacturers	TADOTable	Refers to the table tblManufacturers
tblDevices	TADOTable	Refers to the table tblDevices

2.1 Tab sheet [2.1 - SQL]

Example of graphical user interface (GUI) for QUESTION 2.1:

NOTE: Code to execute the SQL statements and display the results of the queries is provided. The SQL statements assigned to the variables **sSQL1**, **sSQL2**, **sSQL3**, **sSQL4** and **sSQL5** are incomplete.

Complete the SQL statements to perform the tasks described in QUESTION 2.1.1 to QUESTION 2.1.5 below.

2.1.1 Button [2.1.1 - iOS devices]

Display the **DeviceID** and **DeviceName** of all devices in the **tblDevices** table, that use the iOS operating system.

Example of output of the first five records:

DeviceID	DeviceName
D005	Core Sound
D028	Ultra Light
D002	Boom XL
D014	Lock Smart
D021	Sound Wave

(3)

2.1.2 Button [2.1.2 - Category selected]

The user must select a device type from the combo box **cmbQ2_1_2**. Code has been provided to extract and store the device type selected in a variable **sDeviceType**.

Display the **DeviceName**, **Category** and **NumInStock** of all devices where the **Category** field contains the device type selected from the combo box.

Example of output if TV was selected as a device type:

DeviceName	Category	NumInStock
Crystal TV	Smart TV	389
Look TV	Smart TV	91
View Tube	Smart TV	204
Telly Wiz	Smart TV	46

(3)

2.1.3 Button [2.1.3 - Online support]

Display the **DeviceName**, **Category** and **OperatingSystem** of all devices that receive online support from the manufacturer. The results must be sorted in alphabetical order according to the **DeviceName**. See the next page.



2.1.4

Example of output of the first five records:

DeviceName	Category	OperatingSystem
Always Summer	Smart heating	Android
Core Sound	Smart speaker	iOS
Crystal TV	Smart TV	HarmonyOS
Ice Box	Smart refrigerator	Android
Key Away	Smart door locks	HarmonyOS

(6)

Button [2.1.4 - Profit per manufacturer]

All the products in stock are available to be purchased at a clearance sale. The price shown in the **tblDevices** table refers to the selling price of the product. The profit after selling the stock is 60% of the selling price. The total profit made by each manufacturer must be calculated assuming that all products will be sold at the sale.

Write code to display the **ManufacturerID** and the total profit in a new field called **Profit**, formatted to currency.

Example of output:

ManufacturerID	Profit
M100	R672 900.00
M101	R438 000.00
M102	R1 090 502.40
M103	R4 382 044.20
M104	R265 860.00
M105	R859 089.00
M106	R475 419.60

(6)

2.1.5 Button [2.1.5 - Remove devices]

The entire range of smart speakers, manufactured by Mobile Comp with ManufacturerID M104, have been recalled due to a fault during the manufacturing process.

Remove all records of smart speakers that were manufactured by **ManufacturerID M104**, from the **tblDevices** table.

NOTE: Code has been provided to display a message to indicate the number of records that have been deleted.

Example of output:



(4)

2.2 Tab sheet [2.2 - Delphi code]

NOTE:

- Use ONLY Delphi programming code to answer QUESTION 2.2.1 and QUESTION 2.2.2.
- NO marks will be awarded for SQL statements in QUESTION 2.2.

Example of graphical user interface (GUI) for QUESTION 2.2:

Database programming

2.1 - SQL 2.2 - Delphi code

ManufacturerID	ManufacturerName	ContactNumber	OnlineSupport
▶ M100	Berry Tech	0122035878	True
M101	Gem Smart	0169725478	False
M102	Gemini	0512025306	False
M103	Jupiter Inc.	0311257412	True

DeviceID	DeviceName	Category	OperatingSystem	Price	NumInStock	ManufacturerID
▶ D001	Always Summer	Smart heating	Android	R300.00	107	M105
D002	Boom XL	Smart speaker	iOS	R1 500.00	80	M104
D003	Boom XXL	Smart speaker	iOS	R2 000.00	50	M104
D004	Bulb 6	Smart lighting	Android	R500.00	100	M101

2.2.1

2.2.1 - Display products

2.2.2

2.2.2 - Update stock

Restore database Close



2.2.1

Button [2.2.1 - Display products]

Retailers want to see which products are made by each manufacturer.

Code has been provided to clear the content in the **redQ2_2_1** component.

Write code to do the following:

- Display the manufacturer name and contact number in the following format:

`<Manufacturer Name>: <Contact Number>`

- Display the following headings for each manufacturer:

`'Device name' 'In stock' 'Price'`

- Display all the devices made by each manufacturer using the appropriate headings.

Example of output for the first two manufacturers (Berry Tech and Gem Smart):

Berry Tech: 0122035878		
Device name	In stock	Price
Key Away	240	R600.00
Look TV	91	R10 000.00
Stay Safe 2	150	R450.00
Gem Smart: 0169725478		
Device name	In stock	Price
Bulb 6	100	R500.00
Freeze 3	2	R12 000.00
Noise Maker	296	R1 000.00
Time Keeper	80	R4 500.00

(12)

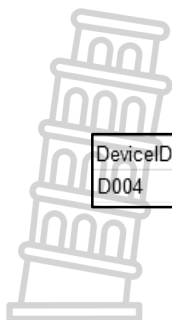
2.2.2 **Button [2.2.2 -Update stock]**

When devices are sold, the number in stock must be decreased in the table **tblDevices**.

The user must select the record of the device to be sold from the **tblDevices** table in the DBGrid component **dbgDevices**.

Write code to do the following:

- Provide an input dialog box for the user to enter the number of devices to be sold for the record selected.
- If stock is sufficient, decrease the number in stock for the selected item with the number that was entered. If stock is insufficient, display a suitable message.



Example of output if the 'Bulb 6' item had been selected **before** any items were sold:

DeviceID	DeviceName	Category	OperatingSystem	Price	NumInStock	ManufacturerID
D004	Bulb 6	Smart lighting	Android	R500.00	100	M101

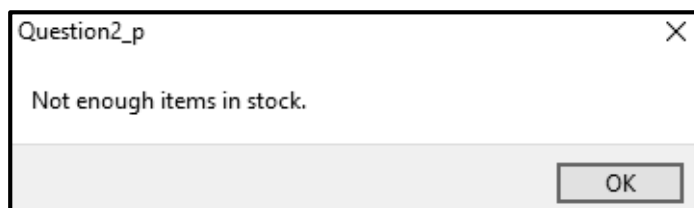
Example of output if the value of 15 was entered and 15 units of the selected item were sold:

DeviceID	DeviceName	Category	OperatingSystem	Price	NumInStock	ManufacturerID
D004	Bulb 6	Smart lighting	Android	R500.00	85	M101

Example of output if the 'Bulb 6' item had been selected **before** any items were sold:

DeviceID	DeviceName	Category	OperatingSystem	Price	NumInStock	ManufacturerID
D004	Bulb 6	Smart lighting	Android	R500.00	100	M101

Example of output if the value of 101 was entered for units to be sold:



(6)

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

TOTAL SECTION B: 40



SECTION C

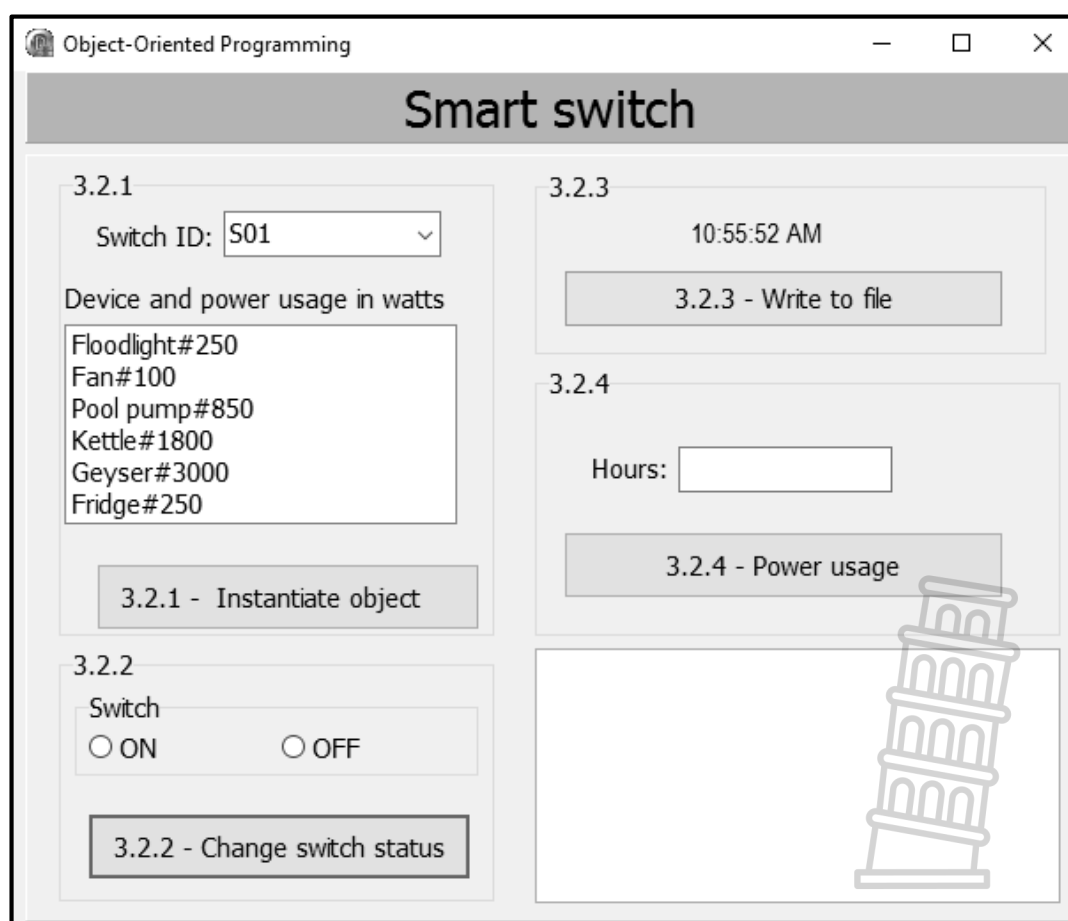
QUESTION 3: OBJECT-ORIENTATED PROGRAMMING

Smart switches have been installed to monitor and automate home devices. The program to be developed will determine the device power usage of the switch and the user will have the option of switching the device ON or OFF.

Do the following:

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

Example of graphical user interface (GUI):



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

3.1 The given incomplete object class (**TSmartSwitch**) contains:

- The declaration of four attributes which describe a **SmartSwitch** object
- Code for a method called **determineSwitchStatus**

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

Attribute	Type	Description
fSwitchID	String	The identifying code of the switch, e.g. S01
fDevice	String	The name of the device that will be used on the switch
fPowerUsage	Integer	The amount of energy/power in watts used by the device
fSwitchStatus	Boolean	The current status of the switch, either True for ON or False for OFF

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

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 switch ID, the name of the device and the power usage of the switch as parameters. Assign the parameter values to the respective attributes. Allocate the value 'false' to the **fSwitchStatus** attribute. (5)

3.1.2 Write code for an accessor method called **getSwitchID** to return the **fSwitchID** attribute. (2)

3.1.3 Write code for a method called **energyUsed** to receive the number of hours (as an integer) that the device was switched on, as a parameter. Use the formula below to calculate and return the amount of energy (kilowatt-hour (kWh)) used by the device.

$$\text{Energy used} = \frac{\text{PowerUsage} * \text{Hours}}{1000} \quad (4)$$

3.1.4 Write code for a mutator method called **setSwitchStatus** which can be used to switch the device ON or OFF. The method must receive a Boolean value as a parameter and set the **fSwitchStatus** attribute to the value received. (3)

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

Switch ID: <SwitchID>
Device: <Device>
Power usage: <PowerUsage> W
Switch status: <ON> or <OFF>

Call the provided method **determineSwitchStatus** to determine if the switch is ON or OFF. (4)

3.2 An incomplete program, **Question3_P**, 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 **objSmartSwitch**.

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

Code has been provided in each of the buttons to clear the **redQ3** rich edit component.

3.2.1 Button [3.2.1 - Instantiate object]

A list box **lstQ3_2_1** contains devices and their power usage in the following format:

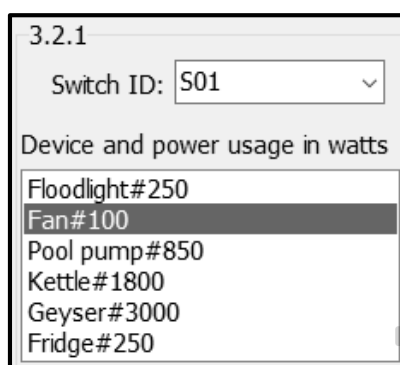
<Device>#<Power usage in watts>

The user needs to select a switch ID from the combo box **cmbQ3_2_1** and any device from the list box **lstQ3_2_1**.

Write code to do the following:

- Extract the switch ID from the combo box **cmbQ3_2_1**.
- Extract the name of the device and the power usage in watts from the list box **lstQ3_2_1**. Separate the device from the power usage.
- Use the switch ID, name of the device and power usage to instantiate a new **SmartSwitch** object.
- Call the **toString** method to display the information of the **SmartSwitch** object in the rich edit **redQ3**.

Example of input:



Example of output after instantiation of the object:

```
Switch ID: S01
Device: Fan
Power usage: 100 W
Switch status: OFF
```



3.2.2 Button [3.2.2 - Change switch status]

Use the radio group **rgpQ3_2_2** to select the switch to be ON or OFF.

Write code to do the following:

- Call the **setSwitchStatus** method using the value selected from the radio group as an argument. The value ON represents True and the value OFF represents False.
- Display the ID and the status of the **objSmartSwitch** object in the rich edit **redQ3**.

Example of input if the 'ON' option has been selected for the fan device:

Example of output if the 'ON' option has been selected for the fan device:

S03: ON

(5)

3.2.3 Button [3.2.3 - Write to file]

A text file called **log.txt** keeps a record of the date, time, switch ID and the status of the switch.

Code has been provided in the **FormCreate** event to show the current time in the label **lblTime**.

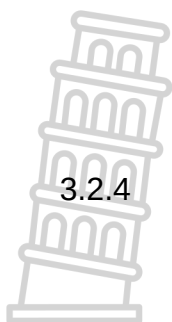
Write code to do the following:

- Open the text file **log.txt** to add a line of text.
- Add a line of text to the file which contains the current system date, current system time, the switch ID and the switch status in the following format:

Current date,Current time,Switch ID#Switch status

Example of output to the file if the current system date and time is 23 May 2023, 9:18:06 AM:

23/05/2023,9:18:06 AM,S03#ON



3.2.4

NOTE: A date function must be used to extract the current system date. The format of the date and time are dependent on the regional settings of your computer, which may differ from the example output provided.

(6)

Button [3.2.4 - Power usage]

The switch will be left on for a specific number of hours.

The user must enter the number of hours in the edit box **edtQ3_2_4**.

Write code to do the following:

- Obtain the number of hours that was entered from the edit box **edtQ3_2_4**.
- Call the **energyUsed** method using the number of hours that was entered as an argument.
- Display a suitable message in the rich edit **redQ3** to indicate the amount of energy used in kWh.

Example of input and output if the device selected is a fan 100 W and the number of hours is 4:

Example of input and output if the device selected is a pool pump of 850 W and the number of hours is 6:

(3)

- 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

SECTION D

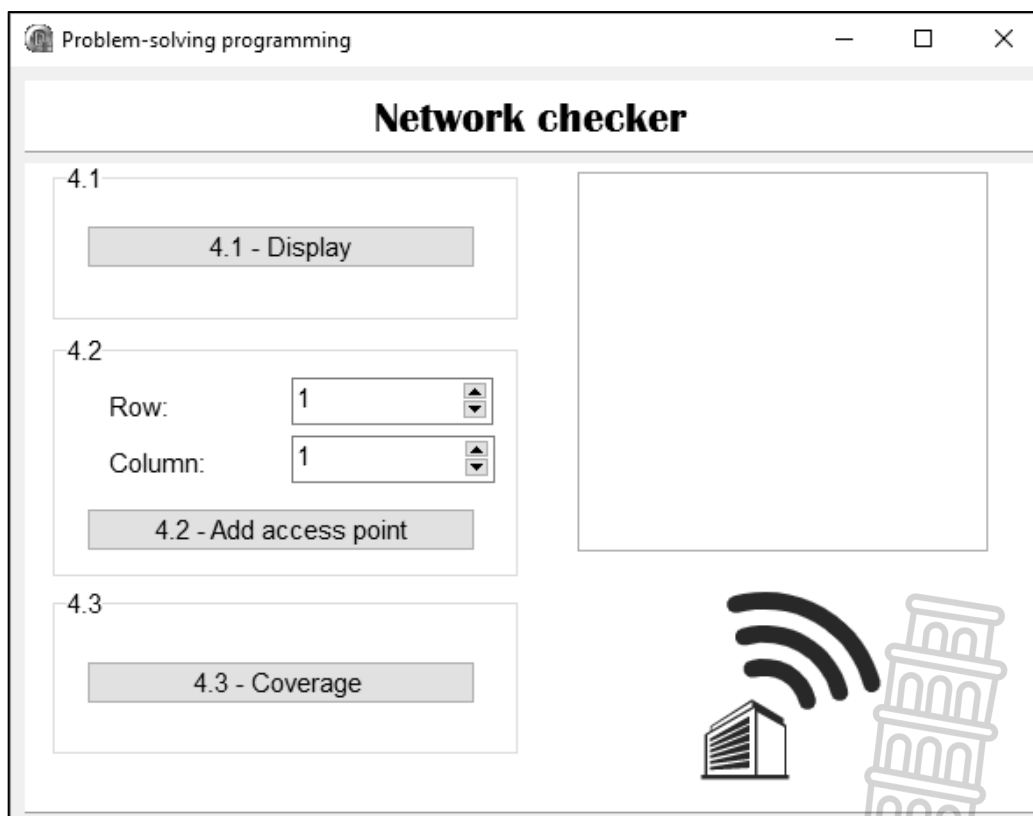
QUESTION 4: PROBLEM-SOLVING PROGRAMMING

You are developing a program to test the wireless network for a college campus. The area of the campus has been broken up into a grid where access points can be added and Wi-Fi coverage will be determined.

Do the following:

- Open the incomplete program in the **Question 4** folder.
- Enter your examination number as a comment in the first line of the **Question4_U.pas** file.
- Compile and execute the program. The program has no functionality currently.

Example of graphical user interface (GUI):



The following two-dimensional array **arrNetwork** has been provided. The array represents the wireless network of the college campus broken up into a grid with 5 rows and 6 columns.

```
arrNetwork: array [1 .. 5, 1 .. 6] of char =
    ((_, 'A', _, _, _, _),
     (_, _, _, _, _, _),
     (_, _, _, _, 'A', _),
     (_, _, _, _, _, _),
     (_, 'A', _, _, _, _));
```

The following table contains the descriptions of the symbols used in the array:

Symbol	Description
Letter ('A')	Access point
Underscore ('_')	Empty space

Complete the code for each section of QUESTION 4, as described in QUESTION 4.1 to QUESTION 4.3 below.

4.1 Button [4.1 - Display]

Write code to display the provided array **arrNetwork** in the rich edit **redQ4**. The row and column headings representing the wireless network of the college campus must be displayed as shown in the example of output.

Example of output:

	Access points					
	1	2	3	4	5	6
1	_	A	_	_	_	_
2	_	_	_	_	_	_
3	_	_	_	_	A	_
4	_	_	_	_	_	_
5	_	A	_	_	_	_

(7)

4.2 Button [4.2 - Add access point]

The program must be able to capture the addition of an access point to the campus network at an available space in the grid. A maximum of three access points in a row is allowed.

The user must select the row and column values from the spin edits **sedQ4_2_Row** and **sedQ4_2_Col** where the access point is to be added.

Write code to do the following:

- Extract the row and column values selected for the location of the access point.
- Add an access point to the location in **arrNetwork** if there is no access point at that location and the total number of access points in that row does not exceed two.

Display an appropriate message if:

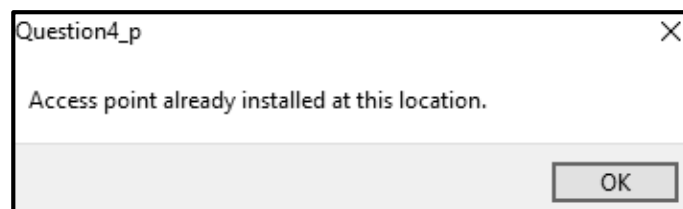
- There is already an access point at the specific location
- The row already contains more than two access points



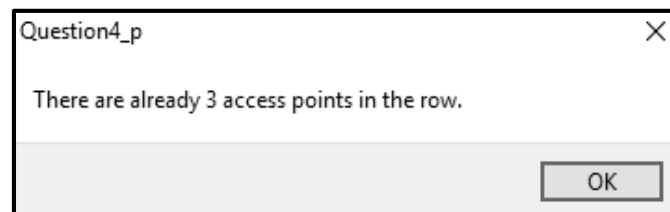
Example of output if the selected row is 3 and the selected column is 1:

	Access points					
	1	2	3	4	5	6
1	-	A	-	-	-	-
2	-	-	-	-	-	-
3	A	-	-	-	A	-
4	-	-	-	-	-	-
5	-	A	-	-	-	-

Example of output if the selected row is 3 and the selected column is 5 and there is already an access point at this location:



Example of output if the user attempts to add an access point in a row that already has 3 access points:



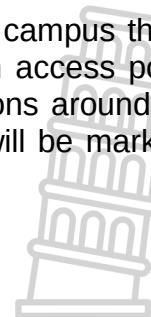
(10)

4.3 Button [4.3 - Coverage]

A grid is used to determine the blocks/locations at the campus that will have coverage and those that will not have coverage. Each access point (A) has the capacity to provide a signal to all the blocks/locations around the access point. The blocks/locations covered per access point will be marked with the asterisk (*) character.

Example:

```
* * *
* A *
* * *
```



Write code to do the following:

- Add coverage symbols (*) around all the access points in the campus network saved in array **arrNetwork**.
- Display array **arrNetwork** after the coverage symbols have been added.

Example of output:



Access points						
	1	2	3	4	5	6
1	*	A	*	-	-	-
2	*	*	*	*	*	*
3	A	*	-	*	A	*
4	*	*	*	*	*	*
5	*	A	*	-	-	-

(13)

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

TOTAL SECTION D: 30
GRAND TOTAL: 150



INFORMATION TECHNOLOGY P1

DATABASE INFORMATION FOR QUESTION 2:

The design of the database tables is as follows:

Table: **tblManufacturers**

This table contains the details of the manufacturers.

Field name	Data type	Description
ManufacturerID	Text (5)	Unique ID of each manufacturer
ManufacturerName	Text (25)	Name of each manufacturer
ContactNumber	Text (10)	The contact number of each manufacturer
OnlineSupport	Yes/No	The value 'true' to indicate that online support will be provided; 'false' if not

Example of the first seven records of the **tblManufacturers** table:

ManufacturerID	ManufacturerName	ContactNumber	OnlineSupport
M100	Berry Tech	0122035878	☒
M101	Gem Smart	0169725478	☐
M102	Gemini	0512025306	☐
M103	Jupiter Inc.	0311257412	☒
M104	Mobile Comp	0412258756	☐
M105	Smart Computing	0169632587	☒
M106	Tech Wiz	0412224568	☒

Table: **tblDevices**

This table contains information on the devices made by each manufacturer.

Field name	Data type	Description
DeviceID	Text (10)	Unique ID for each device
DeviceName	Text (20)	Unique name for each device
Category	Text (20)	Category of each device: smart watch, smart lighting, smart TV, smart speaker, smart refrigerator, smart door lock, smart heating
OperatingSystem	Text(15)	The operating system used to run and connect the manufacturer devices
Price	Currency	The selling price of each device
NumInStock	Number	The amount of stock available for each device
ManufacturerID	Text (5)	ID of the manufacturer of each device

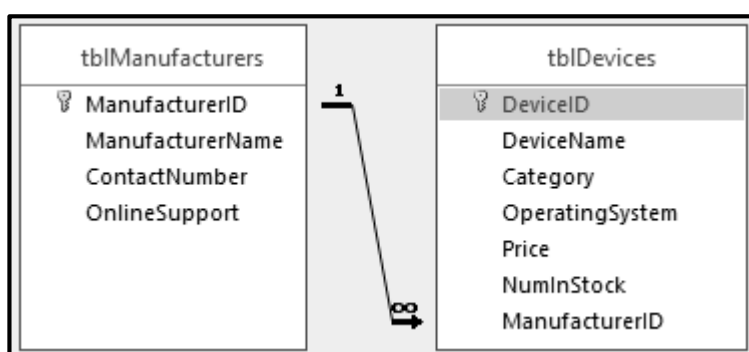
Example of the first ten records of the **tblDevices** table:

DeviceID	DeviceName	Category	OperatingSystem	Price	NumInStock	ManufacturerID
D001	Always Summer	Smart heating	Android	R300.00	107	M105
D002	Boom XL	Smart speaker	iOS	R1 500.00	80	M104
D003	Boom XXL	Smart speaker	iOS	R2 000.00	50	M104
D004	Bulb 6	Smart lighting	Android	R500.00	100	M101
D005	Core Sound	Smart speaker	iOS	R999.00	425	M106
D006	Crystal TV	Smart TV	HarmonyOS	R13 999.00	389	M103
D007	Flash 4	Smart lighting	iOS	R200.00	313	M104
D008	Freeze 3	Smart refrigerator	Android	R12 000.00	2	M101
D009	Ice Box	Smart refrigerator	Android	R14 999.00	14	M105
D010	Jukebox V3	Smart speaker	iOS	R1 299.00	515	M102

NOTE:

- Connection code has been provided.
- The database is password-protected, therefore you will not be able to access the database directly.

The following one-to-many relationship with referential integrity exists between the two tables in the database:





PLANNING PAGE 1





PLANNING PAGE 2





basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

SENIOR CERTIFICATE EXAMINATIONS/ NATIONAL SENIOR CERTIFICATE EXAMINATIONS

INFORMATION TECHNOLOGY P1

2023

MARKING GUIDELINES

MARKS: 150

These marking guidelines consist of 25 pages.



GENERAL INFORMATION:

- These marking guidelines are to be used as the basis for the marking session. They were prepared for use by markers. All markers are required to attend a rigorous standardisation meeting to ensure that the guidelines are consistently interpreted and applied in the marking of candidates' work.
- Note that learners who provide an alternate correct solution to that given as example of a solution in the marking guidelines will be given full credit for the relevant solution, unless the specific instructions in the paper was not followed or the requirements of the question was not met
- **Annexures A, B, C and D** (pages 3 to 10) include the marking grid for each question.
- **Annexures E, F, G and H** (pages 11 to 24) contain examples of solutions for Question 1 to Question 4 in programming code.
- Copies of **Annexures A, B, C, D and the summary for the marks of the learner** (pages 3 to 10) should be made for each learner and completed during the marking session.



ANNEXURE A

QUESTION 1: MARKING GRID – GENERAL PROGRAMMING SKILLS

CENTRE NUMBER:		EXAMINATION NUMBER:	
QUESTION	DESCRIPTION	MAX. MARKS	LEARNER'S MARKS
1.1	Button [1.1 – Formatting] Any ONE of font size or style: ✓✓ Set font size of edtQ1_1 to 14 edtQ1_1.Font.Size (1) := 14 (1) OR Set font style of edtQ1_1 to underline edtQ1_1.Font.Style (1) := [fsUnderline] (1) Set text of edtQ1_1 to 'Hello world' ✓ Set colour of edtQ1_1 to green ✓	4	
1.2	Button [1.2 – Random number] Generate a random number ✓ in the correct range (1 to 99) ✓ Check if ✓ random number is a single digit (≤ 9) ✓ Display the number converted to string ✓ and a message indicating a single digit in pnlQ1_2 ✓ Else ✓ Display the number converted to string and a message indicating a two digit in pnlQ1_2 ✓ Concepts: • Generate a random number (1) in the correct range (1) • Testing if random number is (1): ○ A single digit (1), display number converted to string (1) with applicable message (1) ○ A two digit (1), display number converted to string with applicable message (1)	8	
1.3	Button [1.3 – Area] Extract d from edtQ1_3 converted to integer/float ✓ $area = 3 * \sqrt{3} / 2 * \sqrt{d} - \pi * \sqrt{d/2}$ ✓ Display the value of area in pnlQ1_3 ✓ to one decimal ✓	8	

1.4	Button - [1.4 – Find] Extract sWord from edtQ1_4.text ✓ Create and initialise counter ✓ While NOT EOF(text file) do ✓ Read word from the file ✓ Change words to uppercase/lowercase ✓ If word from file = sWord ✓ (or edtQ1_4.text) Increment counter ✓ If counter > 0 ✓ (or applicable flag) Display occurrences in redQ1_4 ✓ Else ✓ (OR if counter = 0) Display “Word not found” in redQ1_4 ✓	11	
1.5	Button [1.5 – Booster rocket] Create and initialise counter ✓ Get total fuel from input box converted to integer/real ✓ Loop while total fuel >= 200 ✓ Increment counter ✓ fuel used = total fuel * 7.5 / 100 ✓ total fuel = total fuel – fuel used ✓ Display counter, fuel used and total fuel left ✓ in neat columns ✓ formatted to two decimal places ✓	9	
	TOTAL SECTION A:	40	

QUESTION 2: MARKING GRID – SQL AND DATABASE PROGRAMMING

✓

QUESTION 2: MARKING GRID (CONT.)

2.2	Database Manipulation		
2.2.1	Button [2.2.1 – Display products] Go to the first record in the tblManufacturers ✓ Use a loop to step through tblManufacturers ✓ Display the ManufacturerName and ContactNumber ✓ Display the DeviceName, Instock and Price as headings ✓ Go to the first record in the tblDevices table ✓ Use a loop to step through tblDevices ✓ Test if (tblManufacturers ['ManufacturerID'] = tblDevices['ManufacturerID']) ✓ Display DeviceName, NumInStock, converted to a string, ✓ and Price converted to string ✓, in redQ2_2_1 ✓ tblDevices.Next ✓ End loop (tblDevices) tblManufacturers.Next ✓ End loop (tblManufacturers)	12	
2.2.2	Button [2.2.2 – Update stock] Extract iNumSold from input dialog box converted to integer ✓ Test if tblDevices['NumInStock'] >= iNumSold ✓ tblDevices.Edit ✓ tblDevices['NumInStock'] = tblDevices['NumInStock'] - iNumSold ✓ tblDevices.Post ✓ else ShowMessage "Not enough items in stock." ✓ NOTES: • Edit and Post does not need to be inside the if statement. • Any other form of navigation between records can also be used instead of Post.	6	
	Subtotal:	18	
	TOTAL SECTION B:	40	

ANNEXURE C

QUESTION 3: MARKING GRID – OBJECT-ORIENTED PROGRAMMING

CENTRE NUMBER:		EXAMINATION NUMBER:	
QUESTION	DESCRIPTION	MAX. MARKS	LEARNER'S MARKS
3.1.1	Constructor Create: Header with 3 parameter values ✓ and correct data types ✓ Assign correct parameters ✓ to fSwitchID, fDevice and fPowerUsage ✓ Assign False to fSwitchStatus ✓	5	
3.1.2	Function getSwitchID with the string return type ✓ Result := fSwitchID ✓	2	
3.1.3	Function energyUsed with real return type Function with correct return type ✓ and correct parameter ✓ Result = fPowerUsage ✓ * parameter value / 1000 ✓	4	
3.1.4	Procedure setSwitchStatus Correct heading ✓ with Boolean parameter ✓ Set the fSwitchStatus attribute to the parameter value ✓	3	
3.1.5	Function toString with string return type ✓ Result with correct labels and correct attribute names ✓ Call the determineSwitchStatus method ✓ Correct formatting ✓	4	
	Subtotal: Object class	18	

QUESTION 3: MARKING GRID (CONT.)

QUESTION	DESCRIPTION	MAX. MARKS	LEARNER'S MARKS
3.2.1	Button [3.2.1 – Instantiate object] Extract the switchID from the combobox ✓ Extract the selected device from the listbox: ✓ sLine := lstQ3_2_1.Items[lstQ3_2_1.ItemIndex] Extract the device ✓ and power usage ✓ from sLine: sDevice := Copy(sLine, 1, pos('#', sLine) - 1) iWatt := strToInt(Copy(sLine, pos('#', sLine) + 1)) Instantiate the object with the provided values: objSmartSwitch := ✓ TSmartSwitch.create ✓ (sSwitchID, sDevice, iWatt) ✓ Call objSmartSwitch.toString to display in the rich edit ✓	8	
3.2.2	Button [3.2.2 – Change switch status] Use ItemIndex from the radio group with if or case ✓ to call the objSmartSwitch.setSwitchStatus method ✓ with correct True or False argument ✓ Display using the methods objSmartSwitch.getSwitchID ✓ and objSmartSwitch.determineSwitchStatus ✓ in the rich edit	5	
3.2.3	Button [3.2.3 – Write to file] AssignFile(tFile, 'log.txt') and Append(tFile) ✓ WriteLn(tFile, ✓ DateToStr(Date) + ', ' ✓ + lblTime.Caption + ', ' ✓ + objSmartSwitch.getSwitchID + '#' + objSmartSwitch.determineSwitchStatus) ✓ Close file ✓ NOTE: Also accept TimeToStr(Time) instead of lblTime	6	
3.2.4	Button [3.2.4 – Power usage] Extract hours from edtQ3_2_4 and convert to an integer ✓ Use the objSmartSwitch.energyUsed method with hours as an argument ✓ Display the energy used in the rich edit with correct text ✓	3	
	Subtotal Form class:	22	
	TOTAL SECTION C:	40	

ANNEXURE D

QUESTION 4: MARKING GRID – PROBLEM SOLVING

CENTRE NUMBER:		EXAMINATION NUMBER:	
QUESTION	DESCRIPTION	MAX MARKS	LEARNER'S MARKS
4.1	Button [4.1 – Display] Clear redQ4 Add heading and column numbers ✓ Loop iRow from 1 to 5 ✓ Initialise output string with iRow ✓ Correctly nested loop: ✓ Loop iCol from 1 to 6 ✓ Add item at array[iRow, iCol] to output string ✓ Display output string in rich edit ✓	7	
4.2	Button [4.2 – Add access point] Extract the iRow and iCol from the spin edits ✓ Initialise counter ✓ Loop iC from 1 to 6 ✓ Check if array at index [iRow, iC] = 'A' ✓ Increment counter ✓ Check if counter is 2 or less ✓ Check if character at index of the array is NOT an 'A' ✓ Add an 'A' to the correct index of the array ✓ Else Display message that there is an access point ✓ Else Display message that there are already 3 access points ✓	10	

4.3	Button [4.3 – Coverage] Loop through rows iRow from 1 to 5 ✓ Loop through columns iCol from 1 to 6 ✓ If access point is at the current index ✓ Loop ✓ from iRow – 1 ✓ to iRow + 1 ✓ Loop ✓ from iCol – 1 ✓ to iCol + 1 ✓ Check if (row IN [1..5]) and (column IN [1..6]) ✓ If array[row, column] = '_' ✓ (or Not 'A') Set array[row, column] = '*' ✓ Display array after signals have been added ✓ Concepts: Finding the access points: (3) Determine the position (row and column) of an access point Making provision for all surrounding elements: Loop (1) from row index – 1 (1) to row index + 1 (1) Nested loop (1) from Col index – 1 (1) to Col index + 1 (1) Alternative: For individually coded index combinations, subtract 1 mark for each missing combination with a maximum of 6 marks. Test and allocate: Check that indexes does not exceed array size (1) Check if character at indexes is '_' (1) Change character at indexes to '*' (1) NOTE: Testing and allocation needs to be done correctly at least once. Display array after signals have been added (1)	13	
	TOTAL SECTION D: GRAND TOTAL:	30 150	

SUMMARY OF LEARNER'S MARKS:

CENTRE NUMBER:		LEARNER'S EXAMINATION NUMBER:			
	SECTION A	SECTION B	SECTION C	SECTION D	
	QUESTION 1	QUESTION 2	QUESTION 3	QUESTION 4	GRAND TOTAL
MAX. MARKS	40	40	40	30	150
LEARNER'S MARKS					

ANNEXURE E: SOLUTION FOR QUESTION 1

```
unit Question1_u;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
  Forms,
  Dialogs, StdCtrls, ExtCtrls, ComCtrls, pngimage;

type
  TfrmQuestion1 = class(TForm)
    grpQ1_1: TGroupBox;
    edtQ1_1: TEdit;
    btnQ1_1: TButton;
    grpQ1_2: TGroupBox;
    btnQ1_2: TButton;
    pnlQ1_2: TPanel;
    grpQ1_5: TGroupBox;
    edtQ1_5: TEdit;
    redQ1_5: TRichEdit;
    Label1: TLabel;
    btn1_5: TButton;
    grpQ1_3: TGroupBox;
    grpQ1_4: TGroupBox;
    Image1: TImage;
    Label2: TLabel;
    edtQ1_3: TEdit;
    Label3: TLabel;
    btnQ1_3: TButton;
    pnlQ1_3: TPanel;
    btnQ1_4: TButton;
    redQ1_4: TRichEdit;
    procedure btnQ1_1Click(Sender: TObject);
    procedure btnQ1_2Click(Sender: TObject);
    procedure btn1_5Click(Sender: TObject);
    procedure btnQ1_3Click(Sender: TObject);
    procedure btnQ1_4Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmQuestion1: TfrmQuestion1;

implementation

{$R *.dfm}
```



```
// =====
// 1.1 Formatting      4 marks
// =====

procedure TfrmQuestion1.btnQ1_1Click(Sender: TObject);
begin
    edtQ1_1.Font.Size := 14;
    edtQ1_1.Text := 'Hello world';
    edtQ1_1.Font.Style := [fsUnderline];
    edtQ1_1.Color := clGreen;
end;

// =====
// 1.2 Random number  8 marks
// =====

procedure TfrmQuestion1.btnQ1_2Click(Sender: TObject);
var
    iRandom: integer;
begin
    iRandom := Random(99) + 1;
    if iRandom <= 9 then
        pnlQ1_2.Caption := IntToStr(iRandom) + ' is a single digit value'
    else
        pnlQ1_2.Caption := IntToStr(iRandom) + ' is a two digit value';
end;

// =====
// 1.3 Area           8 marks
// =====

procedure TfrmQuestion1.btnQ1_3Click(Sender: TObject);
var
    d, rArea: real;
const
    pi: real = 22 / 7;
begin
    d := StrToFloat(edtQ1_3.Text);
    rArea := (3 * sqrt(3) / 2) * sqr(d) - pi * sqr(d / 2);
    pnlQ1_3.Caption := FloatToStrF(rArea, ffFixed, 7, 1) + ' cm squared';
end;

// =====
// 1.4 Find           11 marks
// =====

procedure TfrmQuestion1.btn1_4Click(Sender: TObject);
var
    tFile: textfile;
    sLine, sWord: String;
    iCount: integer;
begin
    redQ1_4.Clear;
    AssignFile(tFile, 'Words.txt');
    Reset(tFile);
```

```
sWord := (edtQ1_4.Text);
iCount := 0;
while NOT EOF(tFile) do
begin
    Readln(tFile, sLine);
    if UpperCase(sWord) = UpperCase(sLine) then
    begin
        inc(iCount);
    end;
end;
if iCount > 0 then
begin
    redQ1_4.Lines.Add('Occurrences: ' + IntToStr(iCount));
end
else
begin
    redQ1_4.Lines.Add('Word not found');
end;
CloseFile(tFile);
end;

// =====
// 1.5 Booster rocket          9 marks
// =====

procedure TfrmQuestion1.btnQ1_5Click(Sender: TObject);
var
    rTotalFuel, rFuel: real;
    iCounter: integer;
begin

    // Provided code
    redQ1_5.Paragraph.TabCount := 3;
    redQ1_5.Paragraph.tab[0] := 1;
    redQ1_5.Paragraph.tab[1] := 50;
    redQ1_5.Paragraph.tab[2] := 150;

    redQ1_5.Lines.Add('Second' + #9 + 'Fuel used' + #9 + 'Fuel left ');
    //1.5 Booster rocket

    rTotalFuel := StrToFloat(inputbox('Fuel', 'Total litres of fuel: ',
    '550'));

    iCounter := 0;
    while rTotalFuel >= 200 do
    begin
        inc(iCounter);
        rFuel := rTotalFuel * 7.5 / 100;
        rTotalFuel := rTotalFuel - rFuel;
        redQ1_5.Lines.Add(IntToStr(iCounter)+ #9+
                           FloatToStrF(rFuel,FFFixed,5,2)+#9+
                           FloatToStrF(rTotalFuel,FFFixed,5,2));
    end;
end;
end.
```

ANNEXURE F: SOLUTION FOR QUESTION 2

```
//=====
// 2.1 – Section: SQL statements
//=====
```

```
//=====
// 2.1.1 iOS devices 3 marks
//=====
```

```
sSQL1 := 'SELECT DeviceID, DeviceName FROM tblDevices ' +
         'WHERE OperatingSystem = "iOS";
```

```
//=====
// 2.1.2 Category selected 3 marks
//=====
```

```
sSQL2 := 'SELECT DeviceName, Category, NumInStock ' +
         'FROM tblDevices ' +
         'WHERE Category LIKE "%" + sDeviceType + "';
```

```
//=====
// 2.1.3 Online support 6 marks
//=====
```

```
sSQL3 := 'SELECT DeviceName, Category, OperatingSystem ' +
         'FROM tblDevices D, tblManufacturers M ' +
         'WHERE D.ManufacturerID = M.ManufacturerID ' +
         'AND OnlineSupport = True ' +
         'ORDER BY DeviceName';
```

```
//=====
// 2.1.4 Profit per manufacturer 6 marks
//=====
```

```
sSQL4 := 'SELECT ManufacturerID, ' +
         'FORMAT(SUM(NumInStock * (Price * 0.6)), "CURRENCY") ' +
         'AS Profit ' +
         'FROM tblDevices GROUP BY ManufacturerID';
```

```
//=====
// 2.1.5 Remove devices 4 marks
//=====
```

```
sSQL5 := 'Delete FROM tblDevices ' +
         'WHERE Category = "Smart speaker" ' +
         'AND ManufacturerID = "M104";
```

```
//=====
// 2.2 – Section: Delphi code
//=====

//=====
// 2.2.1 Display products          12 marks
// =====
procedure TfrmQuestion2.btnQ2_2_1Click(Sender: TObject);
begin
    // Provided code
    redQ2_2_1.Clear;
    // Question 2.2.1
    tblManufacturers.First;
    while NOT tblManufacturers.Eof do
    begin
        redQ2_2_1.Lines.Add(tblManufacturers['ManufacturerName'] + ': ' +
                             tblManufacturers['ContactNumber']);
        redQ2_2_1.Lines.Add(#9 + 'Device name' + #9 + 'In stock' + #9 +
                             'Price');

        tblDevices.First;
        while NOT tblDevices.Eof do
        begin
            if (tblDevices['ManufacturerID'] =
                tblManufacturers['ManufacturerID']) then
            begin
                redQ2_2_1.Lines.Add(#9 + tblDevices['DeviceName'] + #9
                                     + IntToStr(tblDevices['NumInStock']) +
                                     #9 +
                                     FloatToStrF(tblDevices['Price'],
                                                  ffCurrency, 8, 2));

                end;
                tblDevices.Next;
            end;
            redQ2_2_1.Lines.Add('');
            tblManufacturers.Next;
        end;
    end;
end;
// =====
// 2.2.2 Update stock             6 marks
// =====
procedure TfrmQuestion2.btnQ2_2_2Click(Sender: TObject);
var
    iNumSold: integer;
begin
    // Question 2.2.2
    iNumSold := StrToInt(InputBox('Products sold', 'Amount:', '50'));
    if tblDevices['NumInStock'] - iNumSold >= 0 then
    begin
        tblDevices.Edit;
        tblDevices['NumInStock'] := tblDevices['NumInStock'] - iNumSold;
        tblDevices.Post;
    end
    else
        ShowMessage('Not enough items in stock.');
```

```
// =====
// {$ENDREGION}
// =====
// {$REGION 'Provided code: Setup DB connections - DO NOT CHANGE!'}
// =====

procedure TfrmQuestion2.FormClose(Sender: TObject; var Action:
TCloseAction);
begin
// Disconnects from database and closes all open connections
  dbCONN.dbDisconnect;
end;

procedure TfrmQuestion2.FormCreate(Sender: TObject);
begin
// Provided code
  redQ2_2_1.Paragraph.TabCount := 2;
  redQ2_2_1.Paragraph.Tab[0] := 100;
  redQ2_2_1.Paragraph.Tab[1] := 150;
  redQ2_2_1.Paragraph.Tab[2] := 200;
end;

procedure TfrmQuestion2.FormShow(Sender: TObject);
begin
// Sets up the connection to database and opens the tables.
  dbCONN := TConnection.Create;
  dbCONN.dbConnect;
  tblManufacturers := dbCONN.tblOne;
  tblProducts := dbCONN.tblMany;
  dbCONN.setupGrids(dbgManufacturers, dbgProducts, dbgSQL);
  pgcDBAdmin.ActivePageIndex := 0;
end;
// =====
// {$ENDREGION}

end.
```



ANNEXURE F: SOLUTION FOR QUESTION 3

Object class

```
// =====  
// 3.1.1 Constructor 5 marks  
// =====  
constructor TSmartSwitch.create(sSwitchID: String; sDevice: String;  
iPowerUsage: Integer);  
begin  
    fSwitchID := sSwitchID;  
    fDevice:=sDevice;  
    fPowerUsage := iPowerUsage;  
    fSwitchStatus := False;  
end;  
// =====  
// 3.1.2 getSwitchID 2 marks  
// =====  
function TSmartSwitch.getSwitchID: String;  
begin  
    Result := fSwitchID;  
end;  
  
// =====  
// 3.1.3 energyUsed 4 marks  
// =====  
function TSmartSwitch.energyUsed(iHours: Integer): Real;  
begin  
    Result := fPowerUsage * iHours / 1000;  
end;  
  
// =====  
// 3.1.4 setSwitchStatus 3 marks  
// =====  
procedure TSmartSwitch.setSwitchStatus(bStatus: Boolean);  
begin  
    fSwitchStatus := bStatus;  
end;  
  
// =====  
// 3.1.5 toString 4 marks  
// =====  
function TSmartSwitch.toString: String;  
begin  
    Result := 'Switch ID: ' + fSwitchID + #13 +  
        'Device: ' + fDevice + #13 +  
        'Power usage: ' + intToStr(fPowerUsage) + 'W' + #13 +  
        'Switch status:' + determineSwitchStatus;  
end;
```

```
// =====  
// Provided Code  
// =====  
  
function TSwitch.determineSwitchStatus: String;  
var  
    sStatus: String;  
begin  
    case fSwitchStatus of  
        True: sStatus := 'ON';  
        False: sStatus := 'OFF';  
    end;  
    Result := sStatus;  
end;  
  
// =====  
// End of provided Code  
// =====
```



Main form unit

```
unit Question3_u;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
  Forms,
  Dialogs, SmartSwitch_u, StdCtrls, ComCtrls, Spin, ExtCtrls;

type
  TfrmQuestion3 = class(TForm)
    redQ3: TRichEdit;
    btnQ3_2_1: TButton;
    Panel1: TPanel;
    btnQ3_2_2: TButton;
    GroupBox3: TGroupBox;
    Label3: TLabel;
    GroupBox2: TGroupBox;
    rgpQ3_2_2: TRadioGroup;
    Panel2: TPanel;
    GroupBox1: TGroupBox;
    lstQ3_2_1: TListBox;
    btnQ3_2_4: TButton;
    cmbQ3_2_1: TComboBox;
    Label1: TLabel;
    edtQ3_2_4: TEdit;
    Label2: TLabel;
    GroupBox4: TGroupBox;
    btnQ3_2_3: TButton;
    lblTime: TLabel;
    procedure btnQ3_2_1Click(Sender: TObject);
    procedure btnQ3_2_2Click(Sender: TObject);
    procedure btnQ3_2_4Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmQuestion3: TfrmQuestion3;
  objSmartSwitch: TSmartSwitch;

implementation

{$R *.dfm}
```



```
// =====
// 3.2.1 Instantiate object      8 marks
// =====
procedure TfrmQuestion3.btnQ3_2_1Click(Sender: TObject);
Var
  sLine, sDevice:String;
  iWatt:Integer;

begin
  redQ3.Clear;
  sLine := lstQ3_2_1.Items[lstQ3_2_1.ItemIndex];
  sDevice := Copy(sLine, 1, pos('#',sLine) - 1);
  iWatt := strToInt(Copy(sLine,pos('#',sLine)+1

  objSmartSwitch:=TSmartSwitch.create(cmbQ3_2_1.Text,sDevice,iWatt);
  redQ3.Lines.Add(objSmartSwitch.toString);
end;

// =====
// 3.2.2 Change switch status    5 marks
// =====
procedure TfrmQuestion3.btnQ3_2_2Click(Sender: TObject);
begin
  redQ3.Lines.Clear;
  case rgpQ3_2_2.ItemIndex of
    0: objSmartSwitch.setSwitchStatus(True);
    1: objSmartSwitch.setSwitchStatus(False);
  end;
  redQ3.Lines.Add(objSmartSwitch.getSwitchID + ': ' +
objSmartSwitch.determineSwitchStatus);
end;

// =====
// 3.2.3 Write to file           6 marks
// =====

procedure TfrmQuestion3.Button1Click(Sender: TObject);
var
  tFile : textFile;
begin
  AssignFile(tFile, 'log.txt');
  Append(tFile);
  writeln(tFile,DateToStr(now)+' '+lblTime.Caption+', '
+objSmartSwitch.getSwitchID+'#'+ objSmartSwitch.determineSwitchStatus);
  CloseFile(tFile);
end;
```

```
// =====  
// 3.2.4 Power usage          3 marks  
// =====  
  
procedure TfrmQuestion3.btnQ3_2_4Click(Sender: TObject);  
var  
    iHours : Integer;  
    rEnergy: Real;  
begin  
    redQ3.Lines.Clear;  
    iHours := StrToInt(edtQ3_2_4.Text);  
    rEnergy := objSmartSwitch.energyUsed(iHours);  
    redQ3.Lines.Add('Energy used is: ' + FloatToStr(rEnergy) + ' kWh');  
end;  
  
end.
```



ANNEXURE H: SOLUTION FOR QUESTION 4

```
unit Question4_u;

interface

uses
  Windows, Messages, SysUtils, Variants,
  Classes, Graphics,
  Controls, Forms, Dialogs, StdCtrls, ComCtrls,
  ExtCtrls, Buttons, Spin, pngimage;

type
  TfrmQuestion4 = class(TForm)
    Panel1: TPanel;
    Panel2: TPanel;
    btnQ4_1: TButton;
    redQ4: TRichEdit;
    btnQ4_2: TButton;
    btnQ4_3: TButton;
    gbxQ4_3: TGroupBox;
    sedQ4_2_Row: TSpinEdit;
    sedQ4_3_Col: TSpinEdit;
    Label1: TLabel;
    Label2: TLabel;
    gbxQ4_1: TGroupBox;
    gbxQ4_2: TGroupBox;
    Image1: TImage;
    procedure btnQ4_1Click(Sender: TObject);
    procedure btnQ4_2Click(Sender: TObject);
    procedure btnQ4_3Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TfrmQuestion4;

  arrNetwork: array [1 .. 5, 1 .. 6] of char =
    ((' ', 'A', ' ', ' ', ' ', ' '), (' ', ' ', ' ', ' ', ' ', ' '),
    (' ', ' ', ' ', ' ', 'A', ' '), (' ', ' ', ' ', ' ', ' ', ' '),
    (' ', 'A', ' ', ' ', ' ', ' '));

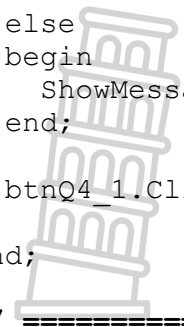
implementation

{$R *.dfm}
```

```
// =====  
// 4.1 Display 7 marks  
// =====  
procedure TfrmQuestion4.btnQ4_1Click(Sender: TObject);  
var  
    I: Integer;  
    J: Integer;  
    sLine: String;  
begin  
    redQ4.Clear;  
    redQ4.Lines.Add('Access points');  
    sLine := ' 1 2 3 4 5 6' + #13;  
    for I := 1 to Length(arrNetwork) do  
        begin  
            sLine := sLine + intToStr(I) + ' ';  
            for J := 1 to Length(arrNetwork[I]) do  
                begin  
                    sLine := sLine + arrNetwork[I, J] + ' ';  
                end;  
  
            sLine := sLine + #13;  
  
        end;  
        redQ4.Lines.Add(sLine);  
    end;  
  
// =====  
// 4.2 Add access point 10 marks  
// =====  
procedure TfrmQuestion4.btnQ4_2Click(Sender: TObject);  
var  
    I, iCounter, iRow, iCol: Integer;  
begin  
    redQ4.Clear;  
    iRow := sedQ4_2_Row.Value;  
    iCol := sedQ4_3_Col.Value;  
  
    iCounter := 0;  
  
    for I := 1 to Length(arrNetwork[iRow]) do  
        begin  
            if arrNetwork[iRow, I] = 'A' then  
                inc(iCounter);  
        end;  
  
        if iCounter < 3 then  
            begin  
                if arrNetwork[iRow, iCol] <> 'A' then  
                    begin  
                        arrNetwork[iRow, iCol] := 'A';  
                    end  
                else  
                    begin  
                        ShowMessage('Access point already on this location.');                    end;  
            end;  
        end;  
    end;
```



```
else
begin
    ShowMessage('There are already 3 access points in the row.');
```



```
end;

btnQ4_1.Click;

end;

// =====
// 4.3 Coverage 13 marks
// =====
procedure TfrmQuestion4.btnQ4_3Click(Sender: TObject);
var
    I: Integer;
    J: Integer;
    K: Integer;
    L: Integer;
begin
    for I := 1 to Length(arrNetwork) do
    begin
        for J := 1 to Length(arrNetwork[I]) do
        begin
            if arrNetwork[I, J] = 'A' then
            begin
                for K := J - 1 to J + 1 do
                begin
                    for L := I - 1 to I + 1 do
                    begin
                        if (K IN [1..5]) AND (L IN [1..6]) then
                        begin
                            if arrNetwork[L, K] = '_' then
                            begin
                                arrNetwork[L, K] := '*';
                            end;
                        end;
                    end;
                end;
            end;
        end;
    end;

    btnQ4_1.Click;
end;
end.
```

