basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

SENIOR CERTIFICATE EXAMINATIONS/ NATIONAL SENIOR CERTIFICATE EXAMINATIONS

INFORMATION TECHNOLOGY P1

MAY/JUNE 2024

MARKS: 150

TIME: 3 hours

This question paper consists of 22 pages, 2 data pages and 2 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: **DataJun2024.exe**
- Click on the 'Extract' button.
- Enter the following password: **#Fun@World$24**

Once extracted, the following list of files will be available in the folder **DataJun2024**:

Question 1:
Question1_P.dpr
Question1_P.dproj
Question1_P.res
Question1_U.dfm
Question1_U.pas

Question 2:
ConnectDB_U.pas
Question2_P.dpr
Question2_P.dproj
Question2_P.res
Question2_U.dfm
Question2_U.pas
VideoUploadsDB - Copy.mdb
VideoUploadsDB.mdb

Question 3:
Album_U.pas
Question3_P.dpr
Question3_P.dproj
Question3_P.res
Question3_U.dfm
Question3_U.pas

Question 4:
Question4_P.dpr
Question4_P.dproj
Question4_P.res
Question4_U.dfm
Question4_U.pas
Top20.txt

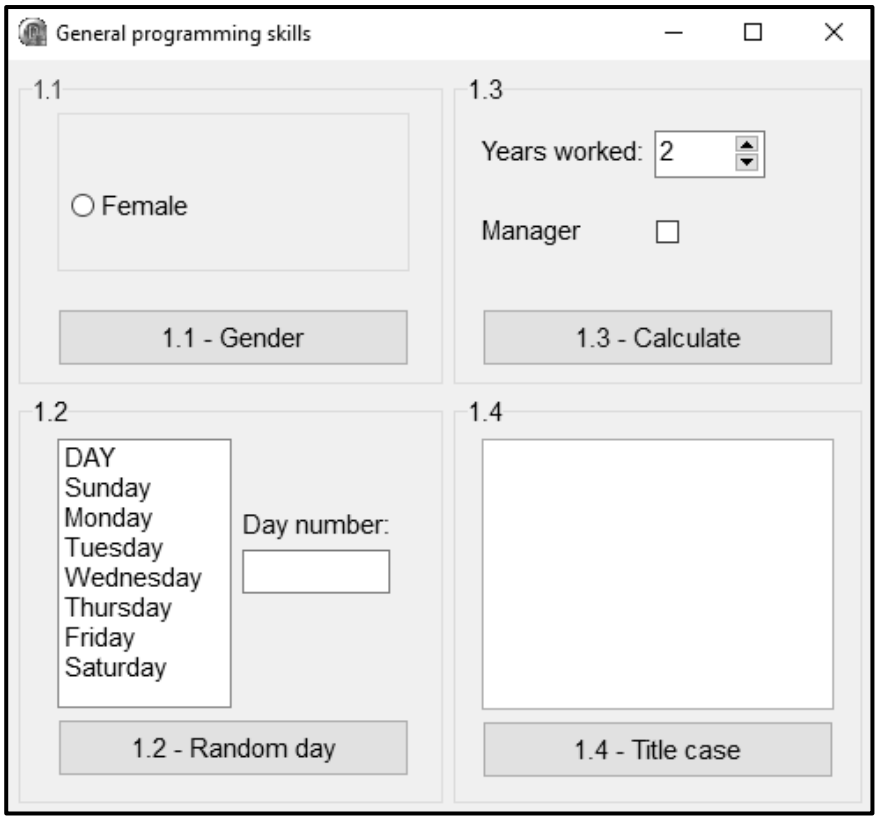
SECTION A

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):



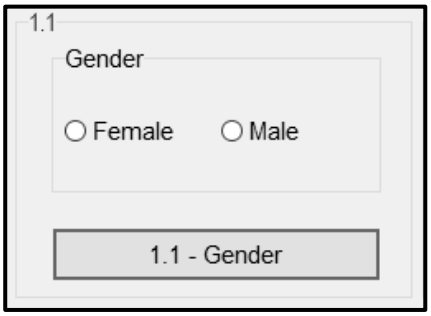
- Complete the code for each section of QUESTION 1, as described in QUESTION 1.1 to QUESTION 1.4.

1.1 **Button [1.1 - Gender]**

Write code to improve the radio group **rgpQ1_1** as follows:

- Set the caption to 'Gender'.
- Add the option 'Male'.
- Set the column property to 2.

Example of the improved radio group **rgpQ1_1**:



(3)

1.2 **Button [1.2 - Random day]**

A list box **lstQ1_2** contains the days of the week from Sunday to Saturday. A day must be selected randomly from the list box and set to be indicated as selected. The number of the selected day and whether the day falls on a weekday or weekend must be displayed.

Write code to do the following:

- Declare a variable to store the random number.
- Generate a number randomly in the range 1 to 7 to select an index from the list box.
- Display the randomly generated number in the label **lblQ1_2** in the format:

  Day: <random number>

- Use the randomly generated number as an index to indicate the corresponding day in the list box.
- Test whether the selected day falls on a weekday or a weekend (Sunday or Saturday) and display, accordingly, either the word 'Weekday' or 'Weekend' in the edit box **edtQ1_2**.

Example of output if the number 3 was randomly generated:

Example of output if the number 7 was randomly generated:

(13)

1.3 **Button [1.3 - Calculate]**

The formula below is used by the owner of a company to calculate the bonus amount of an employee:

$$Bonus = P^{Years} \times \sqrt[2]{P^2 \ / \ 7 \times 20}$$

where P is a constant value.

NOTE: Code has been provided to set the constant P to the value of 8.

Write code to do the following:

- Extract the number of years worked from the spin edit component **spnQ1_3**.
- Use mathematical functions, the constant P and the number of years to calculate the bonus amount.
- If the 'Manager' check box **chkQ1_3** is selected, add 10% to the bonus amount.
- Use a message dialogue box to display the bonus amount formatted as currency and to TWO decimal places.

Example of output if the number of years is 4 and the employee is not a manager:

Example of output if the number of years is 4 and the employee is a manager:

(11)

1.4 **Button [1.4 - Title case]**

A program is required which accepts a sentence as input and displays the sentence in title case. Title case is where every first letter of a word is capitalised.

Code has been provided to do the following:

- Declare a string variable called **sSentence**.
- Clear the rich edit component.
- Store a string obtained from an input box in the **sSentence** variable.

Write code to do the following:

- Use the sentence in the variable **sSentence** and capitalise the first letter of each word in the sentence.
- Display each word from the sentence one below the other in the **redQ1_4** rich edit component.

NOTE: The program must work for any sentence as input and not only the provided sentence in the given code.

Example of output if the sentence used as input was 'Unlock the power of technology and ignite innovation':

(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 A: 40

SECTION B

QUESTION 2: DATABASE PROGRAMMING

A database management system is required for a digital media company that specialises in creating and distributing online video content. The company has several video creators who upload their videos onto an online platform.

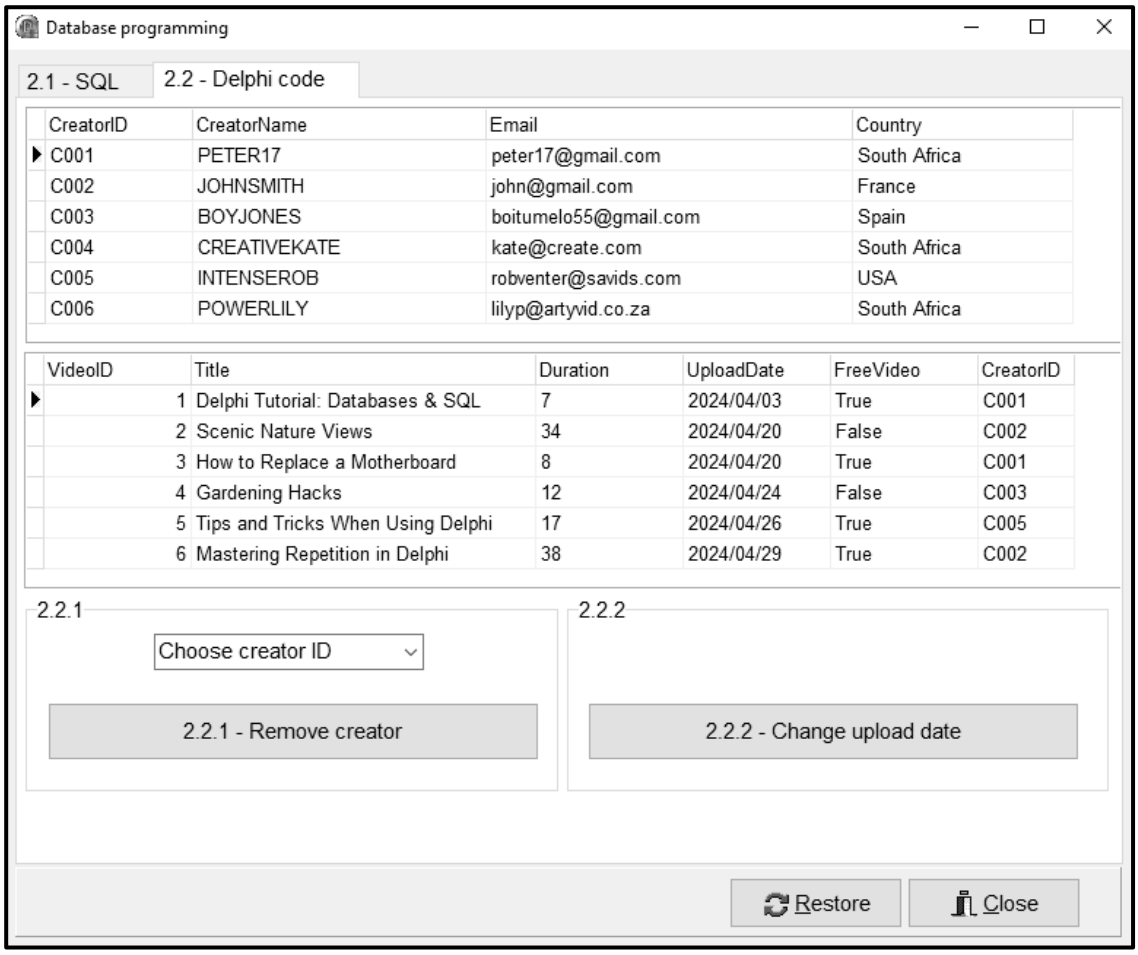
A database called **VideoUploadsDB.mdb** has been developed, which contains information about the different content creators and the videos they uploaded.

The database contains two tables called **tblCreators** and **tblVideos**.

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

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**.



- 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.

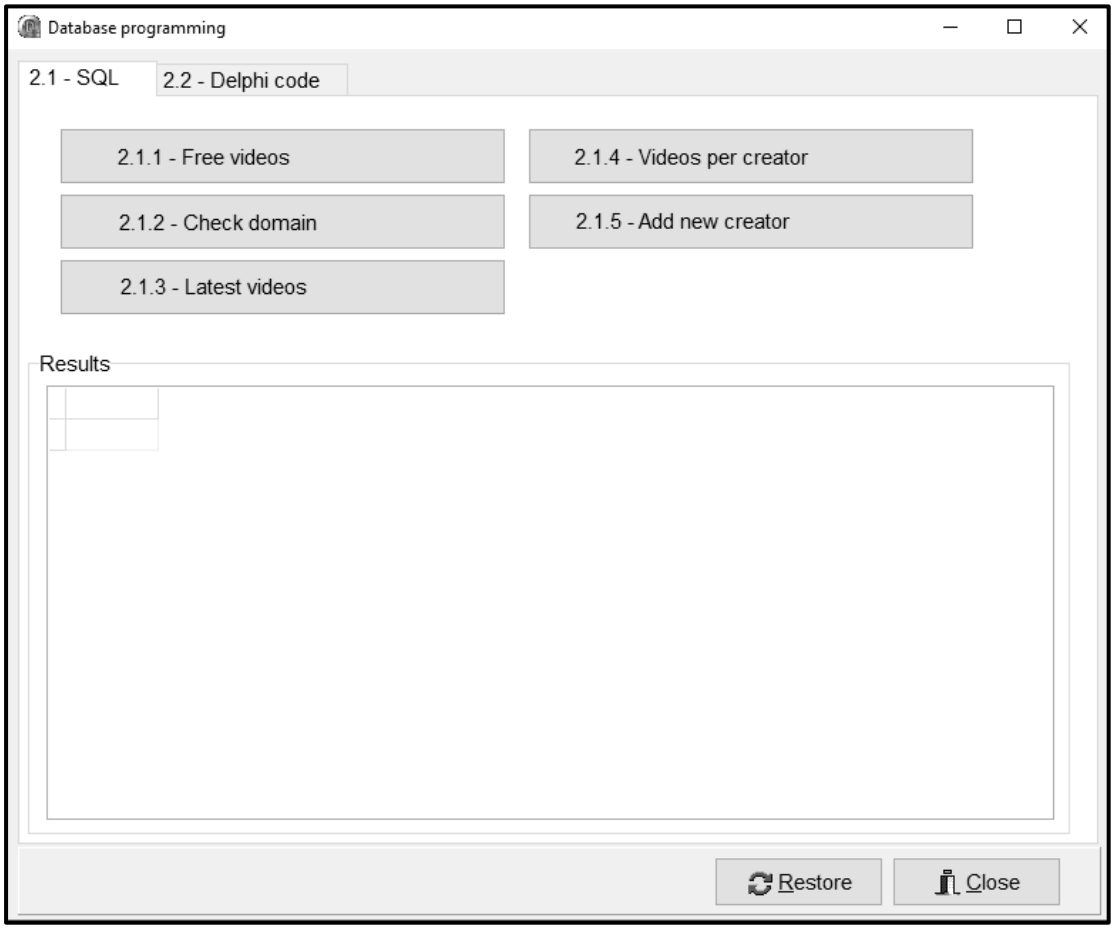
NOTE:

- 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 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 provided code.
- TWO variables are declared as public variables, as described in the table below.

Variable	Data type	Description
`tblCreators`	`TADOTable`	Refers to the table **tblCreators**
`tblVideos`	`TADOTable`	Refers to the table **tblVideos**

2.1 Tab sheet [2.1 - SQL]

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



NOTE:

- Use ONLY SQL statements to answer QUESTION 2.1.1 to QUESTION 2.1.5.
- 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 - Free videos]**

Display the **Title**, **Duration**, **UploadDate** and **CreatorID** of all videos in the **tblVideos** table that are free videos.

Example of output of the first five records:

Title	Duration	UploadDate	CreatorID
Delphi Tutorial: Databases & SQL	7	2024/04/03	C001
How to Replace a Motherboard	8	2024/04/20	C001
Tips and Tricks When Using Delphi	17	2024/04/26	C005
Mastering Repetition in Delphi	38	2024/04/29	C002
Epic Adventure: Hiking the Inca Trail	48	2024/03/17	C005

(3)

2.1.2 **Button [2.1.2 - Check domain]**

Display the **CreatorName**, **Email** and **Country** of all creators from South Africa who do not use an '@gmail' domain/account for their e-mail addresses.

Example of output:

CreatorName	Email	Country
CREATIVEKATE	kate@create.com	South Africa
POWERLILY	lilyp@artyvid.co.za	South Africa

(5)

2.1.3 **Button [2.1.3 - Latest videos]**

Display the **UploadDate**, **VideoID** and **Title** of the three latest videos uploaded.

Example of output:

UploadDate	VideoID	Title
2024/05/02	7	Exploring the Hidden Gems of Tokyo
2024/04/29	6	Mastering Repetition in Delphi
2024/04/26	5	Tips and Tricks When Using Delphi

(4)

2.1.4 **Button [2.1.4 - Videos per creator]**

The company wants a list of creators that have uploaded more than five videos.

Display the **CreatorID** and the total number of videos uploaded by the creator only if the creator uploaded more than five videos.

The number of videos uploaded must be saved in a new field called **NumberUploaded**.

Example of output:

CreatorID	NumberUploaded
C001	6
C005	8

(8)

2.1.5 **Button [2.1.5 - Add new creator]**

New creators join the platform daily and their details must be added to the database.

Add a new creator using the following details:

- `Creator ID: C011`
- `Creator name: TRISHKALOM`
- `Email: trish@rsmarketing.co.za`
- `Country: South Africa`

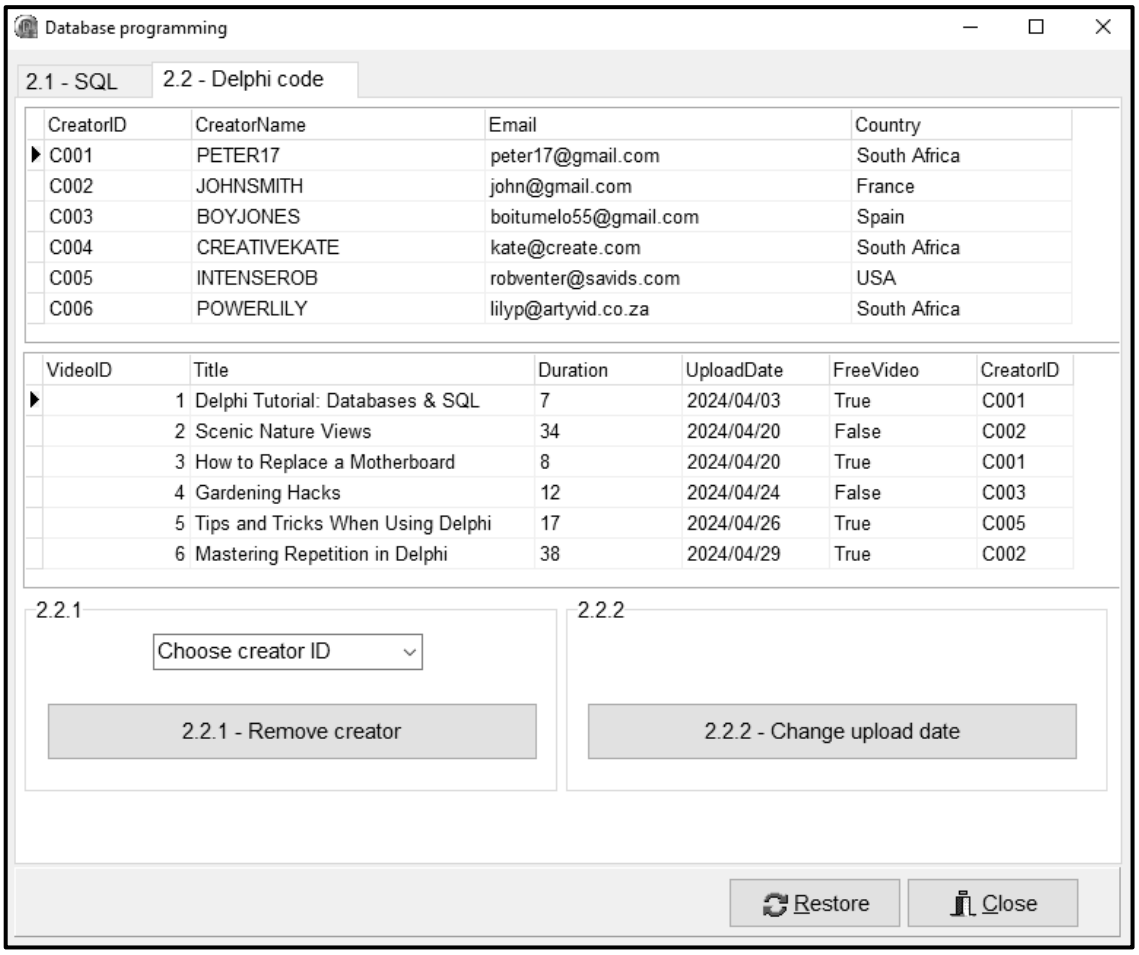
Code has been provided to display a message to indicate that the content of the database has been changed. (4)

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:



2.2.1 Button [2.2.1 - Remove creator]

When a creator leaves the platform, their details must be removed from the database.

The user must select the name of the creator to be removed from the database from the combo box **cmbQ2_2_1**.

The creator and all the videos uploaded by the creator must be removed.

Code has been provided to do the following:

- Extract and store the creator name from the combo box **cmbQ2_2_1**.
- Display a suitable message to confirm that the records have been removed successfully.

Example of output:

(12)

2.2.2 **Button [2.2.2 - Change upload date]**

When the upload date of a video changes, the database must be updated.

The user must select a record of a video from the **tblVideos** table in the dbgrid (**dbgVideos**).

Write code to change the upload date of the record selected to the current system date.

Example of output if VideoID 1 was selected and the current system date is 2024/05/19:

VideoID	Title	Duration	UploadDate	FreeVideo	CreatorID
1	Delphi Tutorial: Databases & SQL	7	2024/05/19	True	C001

NOTE: The output of your program may differ from the provided example based on what the system date is of the computer that you are working on. (4)

- 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

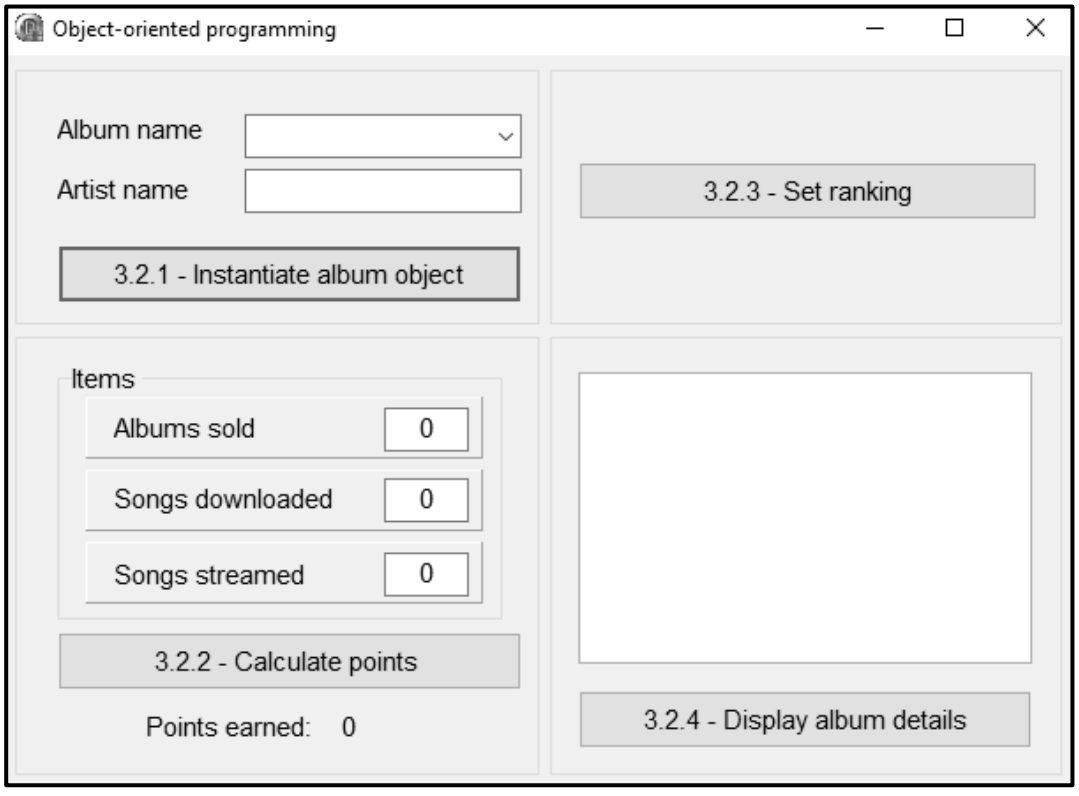
QUESTION 3: OBJECT-ORIENTED PROGRAMMING

A music album is a compilation of many songs recorded by one artist. Music albums are evaluated after every 10 weeks to determine the number of points earned, based on the number of copies sold and the number of songs downloaded and/or streamed from the albums. The total number of points earned and the ranking position of an album over a period of 10 weeks are used to establish whether the album has reached a Gold or Platinum status.

Do the following:

- Open the incomplete program in the **Question 3** folder.
- Open the incomplete object class **Album_U.pas**.
- Enter your examination number as a comment in the first line of both the **Question3_U.pas** and the **Album_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 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 (**TAlbum**) contains the following:

- The declaration of four attributes which describes a music album object
- A completed **toString** method
- A definition (heading) for the **determineStatus** method

The attributes of the album object have been declared as follows:

Attribute	Description
fAlbumTitle	The title of the album
fArtist	The name of the artist who recorded the songs on the album
fHighRanking	A Boolean value which indicates whether an album has a high ranking (true) or not (false)
fPoints	Total number of points earned, based on three items (values), namely the number of copies sold, the number of songs downloaded off the album and the number of songs from the album that was streamed

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 to receive the title of the album and the name of the artist as parameters and do the following:

- Assign the parameter values to the respective attributes.
- Assign the value of FALSE to the **fHighRanking** attribute.
- Assign the value of 0 to the **fPoints** attribute. (5)

3.1.2 Write code for an accessor method called **getPoints** that will return the value of the **fPoints** attribute. (2)

3.1.3 Write code for a method called **updatePoints** that will receive three integer values as parameters, namely the number of albums sold, the number of songs from the album that were downloaded, and the number of songs from the album that were streamed.

Use the three parameter values and the information in the table below to calculate the total number of points earned and update the value of the **fPoints** attribute.

Activity/Item	Number of points allocated
An album sold	100 points
A song downloaded from the album	10 points
A song streamed from the album	1 point

(6)

3.1.4 Write code for a mutator method called **setRanking** that will receive an integer containing the number of weeks that the album was ranked number one. Set the **fHighRanking** attribute to TRUE if the album was ranked number one for more than four weeks. (3)

3.1.5 A partially completed method called **determineStatus** has been provided. Code has been provided to set the status of the album to 'None'.

Add code to complete the method using the **fRanking** and the **fPoints** attributes and the information below to establish the current status of the album.

An album must have a high ranking to receive a status of either 'Gold' or 'Platinum'. The information in the table below is used to determine the status of an album.

Status	High ranking	Points required
None	False	Less than 5 000
Gold	True	At least 5 000 but less than 10 000
Platinum	True	10 000 or more

(7)

3.2 An incomplete program has been supplied in the **Question 3** folder.

The program contains code for the following:

- The declaration of the object variable called **objAlbum**
- Declaration of global integer variables **iSold**, **iDownloaded** and **iStreamed**

Code has been provided in the **OnChange** event of the combo box **cmbQ3_2_1** to do the following when the user selects a title:

- Display the name of the artist in the edit box **edtQ3_2_1** of the album selected.
- Randomly generate and assign values to **iSold**, **iDownloaded**, **iStreamed** respectively. The values refer to the number of copies sold, the number of songs downloaded from the album and the number of songs streamed from the album.
- Populate the edit boxes **edtSold**, **edtDownloaded** and **edtStreamed** with the randomly generated values.

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

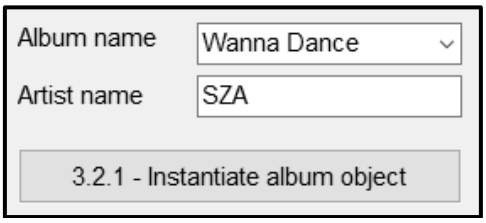
3.2.1 **Button [3.2.1 - Instantiate album object]**

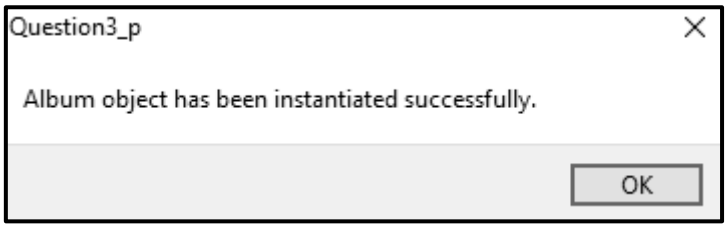
The user must select an album from the combo box **cmbQ3_2_1**.

Write code to instantiate a new album object using the title of the album extracted from the combo box **cmbQ3_2_1** and the name of the artist extracted from the edit box **edtQ3_2_1**.

NOTE: Code has been provided to display a message in a dialogue box to show that the object has been instantiated.

Example of output if the album name selected is 'Wanna Dance':





(5)

3.2.2 **Button [3.2.2 - Calculate points]**

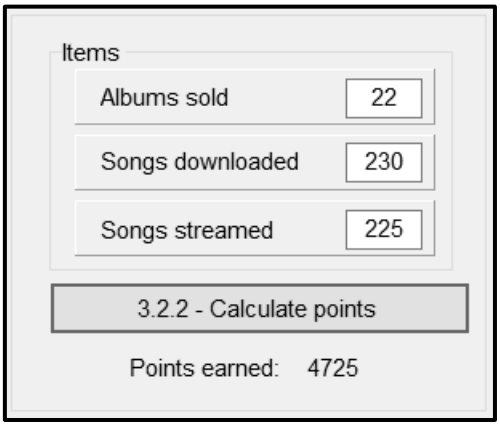
The points allocated to an album is calculated using the number of albums sold, the number of songs downloaded and the number of songs streamed.

Write code to do the following:

- Call the **updatePoints** method and use the randomly generated values as arguments.
- Call the **getPoints** method and display the total number of points that was earned in the label **lblQ3_2_2**.

NOTE: Your output may differ from the example output as the values have been randomly generated.

Example of output:



(5)

3.2.3 Button [3.2.3 - Set ranking]

The ranking of the album object will need to be set to determine whether it is high ranking or not.

Write code to do the following:

- Use an input dialogue box to enter the number of weeks the album was ranked number one and store the value in the provided variable **iNumWeeks**.
- Call the **setRanking** method using **iNumWeeks** as an argument to update the **fHighRanking** attribute of the album object.

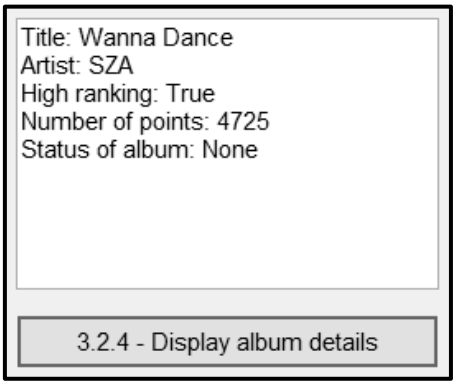
(4)

3.2.4 Button [3.2.4 - Display album details]

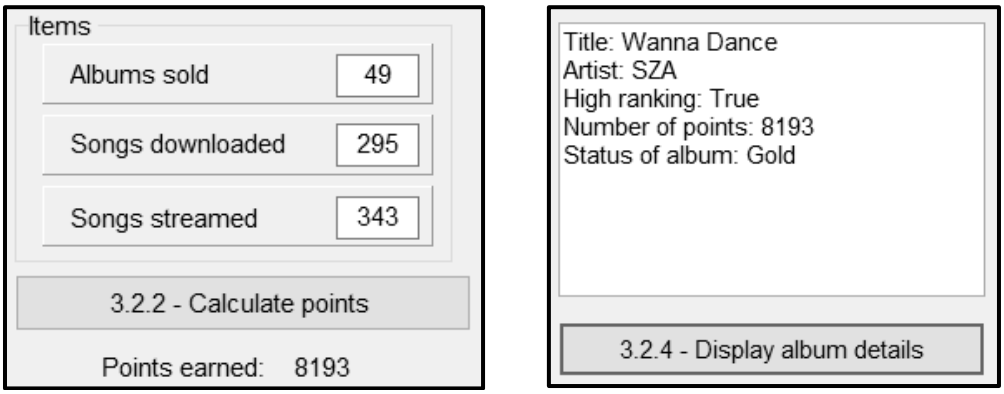
Code has been provided to clear the rich edit **redQ3**. Use the rich edit **redQ3** as the output area for display and write code to do the following:

- Call the **toString** method to display the information of the album object.
- Call the **determineStatus** method to determine the status of the album and display the status.

Example of output:



Another example with a number one ranking for five weeks:



(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

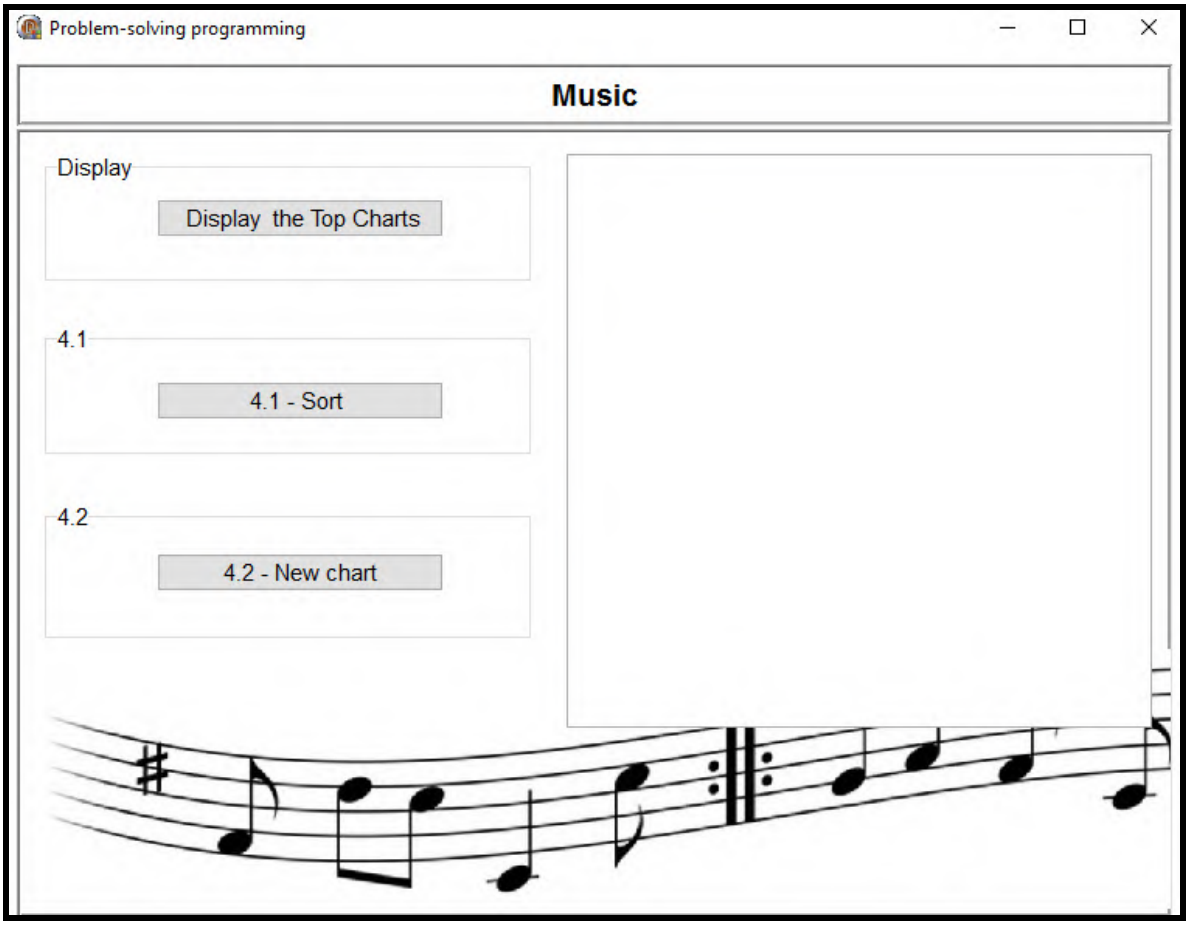
QUESTION 4: PROBLEM-SOLVING PROGRAMMING

The Music Chart Company has contacted you to help them create a program that will display the most popular songs and determine the ranking of new songs.

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):



You have been provided with the following two parallel arrays:

Array	Size	Description
arrSongs	[1..20]	A one-dimensional array that contains a list of twenty songs
arrPosition	[1..20]	A one-dimensional array that contains the ranking position of each song in the array **arrSongs**. The ranking is from 1 to 20.

Code has been provided to display the parallel arrays using the procedure **DisplayArrays**.

Complete the code for each section of QUESTION 4, as described in QUESTION 4.1 and QUESTION 4.2 below.

4.1 **Button [4.1 - Sort]**

The array **arrPosition** contains the current ranking positions of the songs.

Write code to sort the parallel arrays in ascending order according to the ranking position in array **arrPosition**.

Example of output for the first five songs:

```
TOP CHARTS
Songs                Position
Me and You           1
Edges of Dawing      2
Sound of Illusion    3
Castle of Hope       4
Heroic Flavor        5
```

(11)

4.2 **Button [4.2 - New chart]**

You have been provided with a text file, **Top20.txt**, that contains the names of the new top 20 songs ranked in order from 1 to 20.

Example of the first five lines of the text file:

```
Edges of Dawing
Me and You
Warm Heart
New York Dirt
Sound of Illusion
```

Write code to do the following:

- Display the headings Songs, Position and Movement for the columns in the rich edit **redQ4**.
- Extract the new top 20 songs from the text file.
- Compare the songs extracted from the text file with the songs in the sorted array **arrSongs** to determine the movement of the songs in the charts.
- Display the song, position and movement of the songs. The movement should show the number of positions the songs moved up, down, have the same position or whether a new song entered the charts.

NOTE: It is not necessary to change or update the provided array – only generate a new display for information needed.

Example of output after first sorting the original data:

Song	Position	Movement
Edges of Dawing	1	1 UP
Me and You	2	1 DOWN
Warm Heart	3	5 UP
New York Dirt	4	6 UP
Sound of Illusion	5	2 DOWN
Castle of Hope	6	2 DOWN
Heroic Flavor	7	2 DOWN
Wait for Friends	8	1 DOWN
Deep Green Hills	9	3 DOWN
So Hard Spring	10	1 DOWN
Heroic Chances	11	NEW
Free Future	12	SAME POSITION
Longer Tears	13	NEW
Pessimistic Adagio	14	NEW
Discover Backseat Kiss	15	NEW
Earning Nocturno	16	4 UP
Not Night	17	3 DOWN
Lighter Apollo	18	NEW
Winter Friends	19	4 DOWN
Unexpected Skies	20	4 DOWN

(19)

- 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: **tblCreators**

This table contains the details of the content creators.

Field name	Data type	Description
CreatorID	Text (10)	Unique ID for each creator
CreatorName	Text (20)	Unique name of the creator
Email	Text (25)	Email address of the creator
Country	Text (25)	Country of origin of the creator

Example of the records of the **tblCreators** table:

CreatorID	CreatorName	Email	Country
C001	PETER17	peter17@gmail.com	South Africa
C002	JOHNSMITH	john@gmail.com	France
C003	BOYJONES	boitumelo55@gmail.com	Spain
C004	CREATIVEKATE	kate@create.com	South Africa
C005	INTENSEROB	robventer@savids.com	USA
C006	POWERLILY	lilyp@artyvid.co.za	South Africa

Table: **tblVideos**

This table contains the details of each video uploaded to the platform.

Field name	Data type	Description
VideoID	Number	Unique ID for each video
Title	Text (40)	The title of the video
Duration	Number	The duration of the video in minutes
UploadDate	Date/Time	The date the video was uploaded
FreeVideo	Yes/No	A true value indicates that the video is free to watch and false indicates that the video is not free
CreatorID	Text (10)	The ID of the creator that made the video

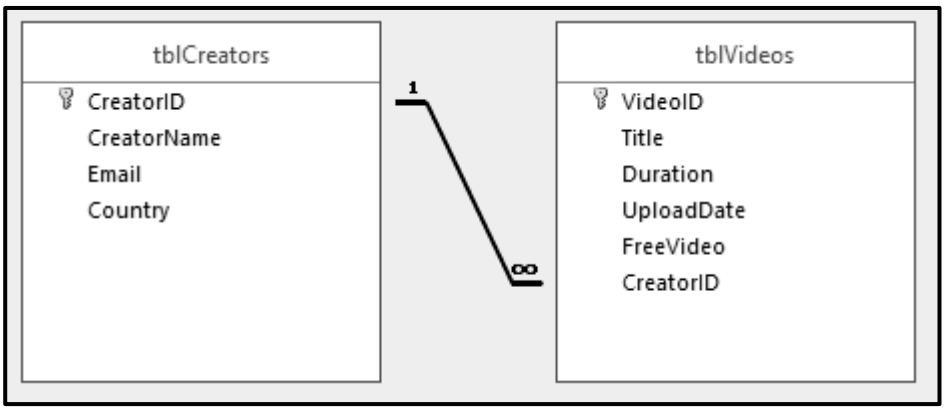
Example of the first ten records of the **tblVideos** table:

VideoID	Title	Duration	UploadDate	FreeVideo
1	Delphi Tutorial: Databases & SQL	7	2024/04/03	☑
2	Scenic Nature Views	34	2024/04/20	☐
3	How to Replace a Motherboard	8	2024/04/20	☑
4	Gardening Hacks	12	2024/04/24	☐
5	Tips and Tricks When Using Delphi	17	2024/04/26	☑
6	Mastering Repetition in Delphi	38	2024/04/29	☑
7	Exploring the Hidden Gems of Tokyo	30	2024/05/02	☐
8	10 Tips for Mastering Chess Strategy	45	2024/03/13	☐
9	Epic Adventure: Hiking the Inca Trail	48	2024/03/17	☑
10	Delicious and Easy Vegan Recipes	12	2024/03/25	☑

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

MAY/JUNE 2024

MARKING GUIDELINES

MARKS: 150

These marking guidelines consist of 29 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 learners' 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 11) include the marking grid for each question.

- **Annexures E, F, G** and **H** (pages 12 to 29) contain examples of solutions for Questions 1 to 4 in programming code.

- Copies of **Annexures A, B, C, D** and **the summary for the marks of the learner** (pages 3 to 11) 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 – Gender]** Set the caption to "Gender" ✓ Add the option "Male" ✓ Set the Columns property to 2 ✓	**3**	

1.2	**Button [1.2 – Random day]**
	
	Declare an integer variable for the random number ✓
	Generate random number in correct range (1 – 7) ✓
	Display "Day:" ✓ and the random number converted to string ✓ in the label lblQ1_2 ✓
	Set the list box item index ✓ to the random number ✓
	Test if ✓ (random number = 1) ✓ or (random number = 7) ✓
	  Display 'Weekend' in edtQ1_2 ✓
	Else ✓
	  Display 'Weekday' in edtQ1_2 ✓
	
	Alternative:
	case lstQ1_2.ItemIndex of (2)
	  1, 7 : edtQ1_2.Text := 'Weekend'; (2)
	  2..6 : edtQ1_2.Text := 'Weekday'; (2)
	end;
		13	

1.3	**Button [1.3 – Calculate]**
	
	Declare a suitable variable/s ✓
	Extract years from the spin edit ✓
	Calculate bonus:
	`rBonus := Power (P,iYears)` ✓
	`  * SQRT` ✓ `(SQR(P)` ✓ `/ 7 * 20` ✓`)`
	
	Test if checkbox manager is ticked ✓
	`  rBonus:= rBonus *` ✓ `1.1` ✓
	`OR  rBonus:= rBonus + rBonus * 0.1`
	
	Display bonus in a dialogue box ✓ formatted to currency and 2 decimal places ✓
		11	

1.4	**Button [1.4 – Title case]** ***Solution 1: Use of the for.. loop*** Add a space to the end of the sentence ✓ Initialise word string ✓ Loop ✓ from 1 to length of sSentence ✓ Check if a character in sSentence ✓ is <> to a space ✓ Concatenate word string ✓ with character in sSentence ✓ Else Convert first letter ✓ of word to uppercase ✓ Display word in redQ1_4 ✓ Clear ✓ word string ✓ ***Alternatives:*** ***Solution 2: Use of the while.. loop)*** ***Solution 3: Use of the repeat..until loop*** ***See examples code in the code section*** **Concepts*:*** Mechanism to include last word (1) Initialise word/ Counter/ Temp (1) Loop (1) with correct lower, upper limit/condition (1) Separate each word in two possible ways (4 marks): *Method 1 (concatenate characters):* Test if character (1) <> space (1) Join character (1) to word (1) *Method 2 (copy from sentence):* Test if character (1) = space (1) Copy word (1) from index 1 to space (1) Convert the first character (1) of the word to uppercase (1) Display each word in redQ1_4 (1) Delete word (1) from index 1 to space (1) OR Clear (1) word (1)	**13**	
	TOTAL SECTION A:	**40**	

ANNEXURE B

QUESTION 2: MARKING GRID – SQL AND DATABASE PROGRAMMING

CENTRE NUMBER:	EXAMINATION NUMBER:		
QUESTION	**DESCRIPTION**	**MAX. MARKS**	**LEARNER'S MARKS**
2.1	**SQL statements**		

2.1.1	**Button [2.1.1 – Free videos]**
	`SELECT Title, Duration, UploadDate, CreatorID` ✓
	`FROM tblVideos` ✓
	`WHERE FreeVideo = True` ✓
	
	Also ACCEPT:
	`WHERE FreeVideo`
	`WHERE FreeVideo = Yes`
	`WHERE FreeVideo = -1`
		3	

2.1.2	**Button [2.1.2 – Check domain]**
	`SELECT CreatorName, Email, Country`
	`FROM tblCreators` ✓
	`WHERE NOT Email` ✓ `LIKE "%@gmail%"` ✓ `AND` ✓
	`Country = "South Africa"` ✓
	
	Also ACCEPT:
	`Where Email NOT LIKE "%@gmail.com"`
		5	

2.1.3	**Button [2.1.3 – Latest videos]** `SELECT Top 3` ✓ `UploadDate, VideoID, Title` `FROM tblVideos` ✓ `ORDER BY UploadDate` ✓ `DESC` ✓	4	
2.1.4	**Button [2.1.4 – Videos per creator]** `SELECT CreatorID,` ✓ `Count(*)` ✓ `AS NumberUploaded` ✓ `FROM tblVideos` ✓ `GROUP BY` ✓ `CreatorID` ✓ `HAVING` ✓ `Count(*) > 5` ✓ **Also ACCEPT**: `Count(field name)`	8	
2.1.5	**Button [2.1.5 – Add new creator]** `INSERT INTO` ✓ `tblCreators` ✓ `VALUES` ✓ `("C011", "TRISHKALOM",` `"trish@rsmarketing.co.za", "South Africa")` ✓	4	
	Subtotal:	**24**	

QUESTION 2: MARKING GRID (CONT.)

2.2	**Database Manipulation**		
2.2.1	**Button [2.2.1 – Remove creator]** Go to the first record in the tblCreators ✓ Use a loop to step through tblCreators ✓ Test if tblCreators['CreatorName'] = sCreatorName ✓ Go to the first record in the tblVideos ✓ Use a loop to step through tblVideos ✓ Test if (tblCreators ['CreatorID'] = ✓ tblVideos ['CreatorID']) ✓ tblVideos.Delete ✓ else ✓ tblVideos.Next ✓ End loop (tblVideos) tblCreators.Delete ✓ // End if tblCreators.Next ✓ End loop (tblCreators) **NOTE:** Loops don't have to be nested. Can first find the CreatorID and use the CreatorID to delete the video records from tblVideos and then remove creator from tblCreators.	12	
2.2.2	**Button [2.2.2 – Change upload date]** tblVideos.Edit; ✓ tblVideos ['UploadDate'] ✓ := Date ✓ tblVideos.Post; ✓ **Also ACCEPT:** DateToStr(Date) DateToStr(DateOf(Now)) Formatdatetime('yyyy/mm/dd',Now())	4	
	Subtotal:	**16**	
	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 method:** Heading ✓ with two parameter values of type string ✓ Assign correct parameter values to fAlbumTitle and fArtist ✓ Assign FALSE to fHighRanking ✓ Assign 0 to fPoints ✓	**5**	
3.1.2	**getPoints function:** Function heading with integer return type ✓ fPoints assigned to result ✓	**2**	
3.1.3	**updatePoints procedure:** Procedure heading ✓ with three integer parameters ✓ fPoints = ✓ album sales * 100 ✓ + downloadSongs * 10 ✓ + streamSongs ✓	**6**	
3.1.4	**setRanking procedure:** Procedure heading with integer parameter ✓ If iNumWeeks > 4 ✓ Set fHighRanking to true ✓	**3**	
3.1.5	**determineStatus function:** Test if fHighRanking = true ✓ Test if fPoints >= 5000 AND ✓ fPoints < 10000 ✓ Assign Gold to status ✓ Test if (fPoints >= 10000) ✓ Assign Platinum to status ✓ Result = status ✓	**7**	
	Subtotal: Object class	**23**	

QUESTION 3: MARKING GRID (CONT.)

QUESTION	DESCRIPTION	MAX. MARKS	LEARNER'S MARKS
3.2.1	**Button [3.2.1 – Instantiate album object]** Extract album title from combo box ✓ Extract artist name from edit box ✓ *Instantiate the album object:* objAlbum:= ✓ TAlbum.Create ✓ Use two string arguments in correct order ✓	**5**	
3.2.2	**Button [3.2.2 - Calculate points]** Call updatePoints method ✓ Using correct three integer arguments ✓ in correct order ✓ Display points in label ✓ by calling getPoints and converting the value to string ✓	**5**	
3.2.3	**Button [3.2.3 – Set ranking]** Extract number of weeks from input dialogue box ✓ and convert to integer ✓ Call the setRanking method ✓ with the number of weeks as an argument ✓	**4**	
3.2.4	**Button [3.2.4 – Display album details]** Display the information in the rich edit component ✓ using the toString method ✓ and the determineStatus method ✓	**3**	

	Subtotal: Form class	**17**	
	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 – Sort]**
	
	Loop ✓ I from 1 to length(arrPosition) ✓
	Correct nesting ✓
	Loop J from 1 to length(arrPosition) – 1 ✓
	Test if arrPosition[J] > ✓
	arrPosition[J + 1] ✓
	iTemp := arrPosition[J]; ✓
	arrPosition[J] := arrPosition[J + 1]; ✓
	arrPosition[J + 1] := iTemp; ✓
	
	swop arrSongs ✓ at correct index ✓
	
	End J loop
	End I loop
	
	Alternative:
	for A := 1 to length(arrPosition) - 1 do (2)
	for B := A+1 to length(arrPosition) do (2)
	if (arrPosition[A] > arrPosition[B]) then (2)
	begin
	iTemp := arrPosition[A]; (1)
	arrPosition[A] := arrPosition[B]; (1)
	arrPosition[B] := iTemp; (1)
	
	sTemp := arrSongs[A];
	arrSongs[A] := arrSongs[B]; } (2)
	arrSongs[B] := sTemp;
	end;
	
	Also ACCEPT:
	Instead of Length(arrPosition) use 20
	Instead of Length(arrPosition) - 1 use 19
		11	

4.2	**Button [4.2 – New chart]** Display the headings (Song, Position and Movement) in redQ4 ✓ **File handling**: AssignFile(tFile, 'Top20.txt') ✓ Reset(tFile) ✓ Loop through the file / Loop `I` 1 to 20 ✓ Read song from the text file ✓ **Movement changes**: Initialise counter `J` to 0 ✓ Initialise flag to false ✓ Loop while `J` < 20 and Flag = false ✓ Increment `J` ✓ if song from text file = arrSongs[J] ✓ set flag to true ✓ if `J > I` ✓ sMovement = intToStr(`J – I`) + ' UP' ✓ else if `J < I` ✓ sMovement = intToStr(`I – J`) + ' DOWN' ✓ else sMovement = 'SAME POSITION' ✓ if song not found in the text file / flag = false ✓ sMovement = 'NEW' ✓ Display song from text file, position and sMovement in redQ4 ✓ **Concepts:** **Display** Display headings to redQ4 (1) Display song, position and movement to redQ4 (1) **File Handling** AssignFile (1) Reset (1) Loop though file (1) Read from file (1) **Movement** *Use counters/indexes to determine the song position (4 marks):* Initialise counters (1) Loop (1) Increment J (1) Test if the song in the textfile = song in arrSongs (1) Determine movement - up (2) Determine movement - down (2) Determine movement - same position (1) Use a flag (2) to determine new songs (2)	**19**	
	TOTAL SECTION D:	**30**	

SUMMARY OF LEARNER'S MARKS:

CENTER 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

```
//=============================================================================
// 1.1 - Gender                                                        3 marks
// ============================================================================

procedure TfrmQuestion1.btn1_1Click(Sender: TObject);
begin
rgpQ1_1.Caption:='Gender';
rgpQ1_1.Items.Add('Male');
rgpQ1_1.Columns:=2;
end;

//=============================================================================
// 1.2 - Random day                                                   13 marks
// ============================================================================

procedure TfrmQuestion1.btnQ1_2Click(Sender: TObject);
var
  iRand: integer;
begin
  iRand := RandomRange(1,8);
  lstQ1_2.ItemIndex:= iRand;
  lblQ1_2.Caption := 'Day: '+IntToStr(iRand);
  if iRand IN [1, 7] then
    begin
      edtQ1_2.Text:= 'Weekend';
    end
  else
    begin
      edtQ1_2.Text:= 'Weekday';
    end;
end;

//=============================================================================
// 1.3 - Calculate                                                    11 marks
// ============================================================================

procedure TfrmQuestion1.btnQ1_3Click(Sender: TObject);
const
  P = 8;
var
  rBonus:Real;
  iYears:Integer;
begin
  iYears := spnQ1_3.Value;
  rBonus := Power(P,iYears) * SQRT(SQR(P) / 7 * 20);
  if chkQ1_3.Checked then
  begin
   rBonus:= rBonus * 1.1;
  end;
  ShowMessage(FloatToStrF(rBonus,ffCurrency,10,2));
end;
```

```
//============================================================================
// 1.4 - Title case                                                13 marks
// ============================================================================

procedure TfrmQuestion1.btnQ1_4Click(Sender: TObject);
var K : integer ;
    sSentence, sWord, sTitleCase : String;
begin
// Provided code
  redQ1_4.Clear;
  sSentence := InputBox('','Enter sentence','Unlock the power of
technology and ignite innovation');
  // sSentence := InputBox('','Enter sentence','Let innovation be your
guiding star as you navigate the realms of cyberspace');

// 1.4 - Title case
  // for loop - Solution 1
   sSentence := sSentence+' ';
  sWord := '';
  for K := 1 to Length(sSentence) do
    begin
      if sSentence[K] <> ' ' then
        sWord := sWord + sSentence[K]
      else
        begin
           sWord[1]:= Upcase(sWord[1]);
           redQ1_4.Lines.Add(sWord);
           sWord := '';
        end;
    end;

  { while loop - Solution 2
  while sSentence <> '' do
    begin
        iPosWord:= Pos(' ',sSentence);
        if iPosWord > 0 then
          begin
             sWord:= Copy(sSentence,1,iPosWord-1);
             sWord:= UPPERCASE(sWord[1]) +
                              Copy(sWord,2,Length(sWord));
             sSentence:= Copy(sSentence,iPosWord+1,
                              Length(sSentence)-iPosWord);
           end
        else
        if iPosWord = 0 then
          begin
              sWord:= UPPERCASE(sSentence[1]) +
                           Copy(sSentence,2,Length(sSentence));
              sSentence:= '';
          end;
       redQ1_4.Lines.Add(sWord);
    end;}
```

```
{ repeat..until loop – Solution 3

   i := 0;
   repeat
    inc(i);
      if sSentence[i] = ' ' then
    begin
      sTemp := Copy(sSentence, 1, i);
      Delete(sSentence, 1, i);
      sTemp := UpperCase(sTemp[1]) + Copy(sTemp,2,length(sTemp));
      redQ1_4.Lines.Add(sTemp);
      i := 0;
    end;
   until sSentence = '';}

end.
```

ANNEXURE F: SOLUTION FOR QUESTION 2

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

// ============================================================================
//  2.1.1 - Free videos                                                3 marks
// ============================================================================

  sSQL1 := 'SELECT Title, Duration, UploadDate, CreatorID ' +
           'FROM tblVideos ' +
           'WHERE FreeVideo = True';

// ============================================================================
//  2.1.2 - Check domain                                               5 marks
// ============================================================================

  sSQL2 := 'SELECT CreatorName, Email, Country ' +
           'FROM tblCreators ' +
           'WHERE Email NOT LIKE "%@gmail.com" AND ' +
           'Country = "South Africa"';

// ============================================================================
//  2.1.3 - Latest videos                                              4 marks
//
============================================================================

  sSQL3 := 'SELECT Top 3 UploadDate, VideoID, Title ' +
           'FROM tblVideos ' +
           'ORDER BY UploadDate DESC';

// ============================================================================
//  2.1.4 - Videos per creator                                         8 marks
// ============================================================================

  sSQL4 := 'SELECT CreatorID, ' +
           'Count(*) AS NumberUploaded ' +
           'FROM tblVideos ' +
           'GROUP BY CreatorID ' +
           'HAVING Count(*)  > 5';

// ============================================================================
//  2.1.5 - Add new creator                                            4 marks
// ============================================================================

  sSQL5 := 'INSERT INTO tblCreators (CreatorID, CreatorName,
            Email, Country) ' +
           'VALUES ("C011", "TRISHKALOM", "trish@rsmarketing.co.za",
                     "South Africa")';
```

```
// ============================================================================
// 2.2 - Section Delphi code
// ============================================================================

// ============================================================================
// 2.2.1 - Remove creator                                          12 marks
// ============================================================================

procedure TfrmQuestion2.btnQ2_2_1Click(Sender: TObject);
var
  sCreatorName : String;
begin
// 2.2.1 - Remove creator
  sCreatorName := cmbQ2_2_1.Text;
  tblCreators.First;
  while NOT tblCreators.Eof do
    begin
      if tblCreators['CreatorName'] = sCreatorName then
        begin
          tblVideos.First;
          while NOT tblVideos.Eof do
            begin
              if tblCreators['CreatorID'] = tblVideos['CreatorID'] then
                tblVideos.Delete
              else
                tblVideos.Next;
            end;
        tblCreators.Delete;
        end;
    tblCreators.Next;
    end;

// Provided code
  ShowMessage('Records deleted successfully');
end;

// ============================================================================
// 2.2.2 - Change upload date                                       4 marks
// ============================================================================

procedure TfrmQuestion2.btnQ2_2_2Click(Sender: TObject);
begin
  tblVideos.Edit;
  tblVideos['UploadDate'] := Date;
  tblVideos.Post;
end;
```

```
// ============================================================================
// {$ENDREGION}
// ============================================================================
// {$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.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, dbgrdSQL);
  pgcDBAdmin.ActivePageIndex := 0;
end;

// ============================================================================
// {$ENDREGION}
// ============================================================================

end.
```

ANNEXURE G: SOLUTION FOR QUESTION 3

Object class

```
unit Album_U;

interface

uses
  SysUtils, StdCtrls, Dialogs, Math;

type
  TAlbum = class(TObject)
  private
    fAlbumTitle: String;
    fArtist: String;
    fHighRanking: Boolean;
    fPoints: Integer;

  public
// Provided code
    function toString: String;
    function determineStatus: String;
    //
=============================================================================

    constructor Create(sAlbumTitle, sArtist: String);
    procedure updatePoints(iAlbumSales, iSongsDownload, iSongsStream:
                                  Integer);
    procedure setRanking(iNumWeeks: Integer);
    function getPoints: Integer;
end;

implementation

// ============================================================================
// Provided code
// ============================================================================
function TAlbum.toString: String;
begin

  Result := 'Title: ' + fAlbumTitle + #13 + 'Artist: ' + fArtist + #13 +
'High ranking: ' +
    BoolToStr(fHighRanking, true) + #13 + 'Number of points: ' + IntToStr
    (fPoints);
end;
// ============================================================================
```

```
// ============================================================================
// 3.1.1 Constructor Create                                          5 marks
// ============================================================================

constructor TAlbum.Create(sAlbumTitle, sArtist: String);
begin
  fAlbumTitle := sAlbumTitle;
  fArtist := sArtist;
  fHighRanking := false;
  fPoints := 0;
end;

// ============================================================================
// 3.1.2 Function getPoints                                          2 marks
// ============================================================================

function TAlbum.getPoints: integer;
begin
  Result := fPoints;
end;

// ============================================================================
// 3.1.3 Procedure updatePoints                                      6 marks
// ============================================================================

procedure TAlbum.updatePoints(iAlbumSales, iSongsDownload,
  iSongsStream: Integer);
begin
  fPoints := iAlbumSales * 100 + iSongsDownload * 10 + iSongsStream;
end;
// ============================================================================
// 3.1.4 Procedure setRanking                                        4 marks
// ============================================================================

procedure TAlbum.setRanking(iNumWeeks: integer);
begin
  if iNumweeks > 4 then
    fHighRanking := true;
end;
```

```
// ============================================================================
// 3.1.5 Function determineStatus                                  7 marks
// ============================================================================
function TAlbum.determineStatus: String;
var
  sStatus: String;
begin
  // Provided code
  sStatus := 'None';

  // 3.1.5
  if (fHighRanking) then
    if (fPoints >= 5000) AND (fPoints < 10000) then
      sStatus := 'Gold';
    if (fPoints >= 10000) then
      sStatus := 'Platinum';
  Result := sStatus;
end;

end.
```

Main Form Unit

```
unit Question3_U;

interface

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

type
  TfrmQuestion3 = class(TForm)
    gbxQ3_2_1: TGroupBox;
    gbxQ3_2_3: TGroupBox;
    redQ3: TRichEdit;
    btnQ3_2_1: TButton;
    gbxQ3_2_2: TGroupBox;
    btnQ3_2_2: TButton;
    Panel1: TPanel;
    Panel2: TPanel;
    btnQ3_2_3: TButton;
    Image1: TImage;
    Label6: TLabel;
    edtQ3_2_1: TEdit;
    Label2: TLabel;
    spnQ3_2_1: TSpinEdit;
    chbQ3_2_1: TCheckBox;
    Label1: TLabel;c
    sedQ3_2_2: TSpinEdit;
    procedure btnQ3_2_1Click(Sender: TObject);
    procedure btnQ3_2_2Click(Sender: TObject);
    procedure btnQ3_2_3Click(Sender: TObject);
  private
    const
    arrArtist: array [1 .. 3] of string = ('SZA', 'Morgan Wallen',
                                           'People');
  public

  end;

var
  frmQuestion3: TfrmQuestion3;
  objAlbum: TAlbum;
  iSold, iDownloaded, iStreamed : integer;
implementation

{$R *.dfm}
```

```
// ============================================================================
// 3.2.1 Instantiate album object                                    5 marks
// ============================================================================

procedure TForm1.btnQ3_2_1Click(Sender: TObject);
var
  sAlbum, sArtist: String;
begin
  sAlbum := cmbQ3_2_1.Text;
  sArtist := edtQ3_2_1.Text;
  objAlbum := TAlbum.Create(sAlbum, sArtist);

  // Provided code
  ShowMessage('Album object has been instantiated successfully.');
end;

// ============================================================================
// 3.2.2 Calculate points                                            5 marks
// ============================================================================

procedure TfrmQuestion3.btnQ3_2_2Click(Sender: TObject);
begin
  objAlbum.updatePoints(iSold, iDownloaded, iStreamed);
  lblQ3_2_2.Caption := IntToStr(objAlbum.getPoints);
end;

// ============================================================================
// 3.2.3 Set ranking                                                 4 marks
// ============================================================================

procedure TfrmQuestion3.btnQ3_2_3Click(Sender: TObject);
var
  iNumWeeks : integer;
begin
  iNumWeeks := StrToInt(InputBox('Number of weeks ranked 1', 'Enter number
of weeks', ''));
  objAlbum.setRanking(iNumWeeks);
end;

// ============================================================================
// 3.2.4 Display album details                                       3 marks
// ============================================================================

procedure TForm1.btnQ3_2_4Click(Sender: TObject);
begin
 redQ3.Clear;
 redQ3.Lines.Add(objAlbum.toString);
 redQ3.Lines.Add('Status of album: ' +
 objAlbum.determineStatus);end;
```

```
// Provided code - do not change
// ================================================================

procedure TfrmQuestion3.cmbQ3_2_1Change(Sender: TObject);
begin
  edtQ3_2_1.Text := arrArtist[cmbQ3_2_1.ItemIndex + 1];
  iSold := Random(100);
  iDownloaded := Random(500);
  iStreamed := Random(500);

  edtSold.Text := IntToStr(iSold);
  edtDownloaded.Text := IntToStr(iDownloaded);
  edtStreamed.Text := IntToStr(iStreamed);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, jpeg, math;
type
  TfrmQuestion4 = class(TForm)
    Panel1: TPanel;
    Panel2: TPanel;
    btnQ4_2: TButton;
    redQ4: TRichEdit;
    btnDisplay: TButton;
    GroupBox1: TGroupBox;
    btnQ4_1: TButton;
    Image1: TImage;
    GroupBox2: TGroupBox;
    GroupBox3: TGroupBox;
    procedure btnQ4_2Click(Sender: TObject);
    procedure btnDisplayClick(Sender: TObject);
    procedure btnQ4_1Click(Sender: TObject);
    procedure FormShow(Sender: TObject);
  private
    procedure DisplayArrays;
    { Private declarations }
  public
    { Public declarations }
  end;
var
  frmQuestion4: TfrmQuestion4;

  arrSongs: array [1 .. 20] of String = (
    'Castle of Hope', 'Deep Green Hills', 'Backseat Kiss', 'Earning
Nocturno', 'Edges of Dawing', 'Free Future', 'Heart Hymn', 'Heroic Flavor',
'Me and You', 'New York Dirt', 'Not Night', 'Adagio', 'Running Study', 'So
Hard Spring', 'Sound of Illusion', 'The Celebration', 'Unexpected Skies',
'Wait for Friends', 'Warm Heart', 'Winter Friends');

  arrPosition: array [1 .. 20] of integer = (
    4, 6, 11, 20, 2, 12, 19, 5, 1, 10, 14, 13, 17, 9, 3, 18, 16, 7, 8, 15);

implementation

{$R *.dfm}
```

```
// ============================================================================
// 4.1 - Sort                                                        11 marks
// ============================================================================

procedure TfrmQuestion4.btnQ4_1Click(Sender: TObject);
var
  I, iTemp: integer;
  J: integer;
  sTemp: String;
begin
// Provided code
  redQ4.Clear;

// Question 4.1

  for I := 1 to length(arrPosition) do
  begin

    for J := 1 to length(arrPosition) - 1 do
    begin

      if arrPosition[J] > arrPosition[J + 1] then
      begin

        iTemp := arrPosition[J];
        arrPosition[J] := arrPosition[J + 1];
        arrPosition[J + 1] := iTemp;

        sTemp := arrSongs[J];
        arrSongs[J] := arrSongs[J + 1];
        arrSongs[J + 1] := sTemp;

      end;

    end;

  end;

  DisplayArrays;

end;
```

```
// ============================================================================
// 4.2 New chart                                                    19 marks
// ============================================================================

procedure TfrmQuestion4.btnQ4_2Click(Sender: TObject);
var
  tFile: TextFile;
  J, iNew: integer;
  bFound: boolean;
  sNewSong, sMsg: String;

// Variables for Alternative 2 and 3
//  tFile: TextFile;
//  arrNewSongs: array [1 .. 20] of String;
//  I: integer;
//  J, iDiff, iPos: integer;
//  bFlag, bFound: boolean;
//  sLine, sOutput: String;
begin
  // Question 4.2

  redQ4.Clear;
  AssignFile(tFile, 'Top20.txt');
  Reset(tFile);

  redQ4.lines.add('Song' + #9 + 'Position' + #9 + 'Movement');

  for iNew := 1 to 20 do
  begin
    readln(tFile, sNewSong);
    J := 0;
    bFound := false;
    while (J < 20) AND (NOT bFound) do
    begin
      inc(J);
      if sNewSong = arrSongs[J] then
      begin
        bFound := true;
        if J > iNew then
          sMsg := IntToStr(J - iNew) + ' UP'
        else if iNew > J  then
          sMsg := IntToStr(iNew - J) + ' DOWN'
        else
          sMsg := 'SAME POSITION';
        end;
      end;
      if NOT bFound then
        sMsg := 'NEW';

      redQ4.lines.add(sNewSong + #9 + IntToStr(iNew) + #9 + sMsg);

    end;
  end;
```

```
{ //Alternative 2
redQ4.Clear;

  AssignFile(tFile, 'Top20.txt');

  try
    reset(tFile);

    for I := 1 to 20 do
    begin

      readln(tFile, arrNewSongs[I]);

    end;
    redQ4.lines.add('Song' + #9 + 'Position' + #9 + 'Movement');

    for I := 1 to length(arrNewSongs) do
    begin

      J := 0;
      bFlag := true;
      while (J < 20) AND (bFlag) do
      begin
        inc(J);
        if arrNewSongs[I] = arrSongs[J] then
        begin
          if J - I > 0 then
          begin
            sLine := arrNewSongs[I] + #9 + intTostr(I) + #9 + '(' +
                     intTostr(abs(J - I)) + ' UP)';
          end
          else if J - I < 0 then
          Begin
            sLine := arrNewSongs[I] + #9 + intTostr(I) + #9
                     + '(' + intTostr(abs(J - I)) + ' DOWN)';
          End
          else
          begin
            sLine := arrNewSongs[I] + #9 + intTostr(I) + #9
                     + '(SAME POSITION)';
          end;
          bFlag := false;
        end
        else
        begin
          sLine := arrNewSongs[I] + #9 + intTostr(I) + #9 + '(NEW)';
        end;
      end;
      redQ4.lines.add(sLine);
    end;
  except
    ShowMessage('File not found');
    Application.Terminate;
  end;
}
```

```
{ //Alternative 3
AssignFile(tFile,'Top20.txt');
 Reset(tFile);
 j:=0;
 while not eof(tFile) do
 Begin
  Readln(tFile,sLine);
  bFlag:= False;
  Inc(j);
  i:=0;
  sOutput:= sLine+#9+IntToStr(j);
   while(bFlag = false) AND (i<20) do
  begin
     Inc(i);
     if arrSongs[i] = sLine then
     begin
        bFlag := True;

    iDiff :=  i - j;
    if iDiff < 0 then
    begin
      sOutput := sOutput+ #9+'('+IntToStr(ABS(iDiff))+' DOWN)';
    end
    else  if  iDiff > 0 then
    begin
      sOutput := sOutput+ #9+'('+IntToStr(iDiff)+' UP)';
    end
    else   if  iDiff = 0 then
    begin
       sOutput := sOutput+ #9+'SAME POSITION';
    end ;

  end; //if bflag - true
  end; //while for array

 if bFlag = False then
  begin
     sOutput:= sOutput+#9+'NEW';
  end;

  redQ4.Lines.Add(sOutput);
 End;//while  for file
 CloseFile(tFile);
end; }
```

```
// ============================================================================
// Provided code
// ============================================================================
procedure TfrmQuestion4.FormShow(Sender: TObject);
begin
  redQ4.Paragraph.TabCount := 3;
  redQ4.Paragraph.Tab[0] := 0;
  redQ4.Paragraph.Tab[1] := 120;
  redQ4.Paragraph.Tab[2] := 180;
end;

procedure TfrmQuestion4.DisplayArrays;
var
  I: Integer;
begin
  // Provided code
  redQ4.lines.add('TOP CHARTS');
  redQ4.lines.add(format('%-20s%-15s%-5s', ['Songs', 'Artist',
'Position']));
  for I := 1 to length(arrSongs) do
    redQ4.lines.add(format('%-20s%-15s%-5d', [arrSongs[I], arrArtists[I],
arrPosition[I]]));
end;

end.
// ============================================================================
  End of provided code
// ============================================================================
```