basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

SENIOR CERTIFICATE EXAMINATION

INFORMATION TECHNOLOGY P1

2015

MARKS: 150

TIME: 3 hours

This question paper consists of 23 pages.

INSTRUCTIONS AND INFORMATION

1. This paper is divided into THREE sections. Candidates must answer ALL THREE sections.

2. 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.

3. This paper is set in programming terms that are not specific to any particular programming language (Delphi/Java (using the Netbeans IDE)).

4. 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 only.

5. Answer only what is asked in each question. For example, if the question does not ask for data validation, then no marks will be awarded for data validation.

6. 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.

7. Routines such as search, sort and selection must be developed from first principles. You may not use the built-in features of a programming language for any of these routines.

8. You must save your work regularly on the disk you have been given, or the disk space allocated to you for this examination session.

9. Make sure that your examination number appears as a comment in every program that you code.

10. If required, print the programming code of all the programs/classes that you completed. You will be given half an hour printing time after the examination session.

11. At the end of this examination session you must hand in a disk/CD/DVD/flash disc 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.

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

NOTE:

- Delphi programmers must use the file **DelphiDataENG.exe**.
- Java programmers must use the file **JavaDataENG.exe**.

Do the following:

- Double click on the password-protected executable file.
- Click on the extract button.
- Enter the following password: **School@?$@2**

List of files provided in the folder DelphiDataENG/JavaDataENG (once extracted):

Delphi files	**Java (Netbeans) files**
Question1:	**Question1:**
Question1P.dpr	Question1.form
Question1P.res	Question1.java
Question1U.dfm	
Question1U.pas	
Question2:	**Question2:**
OrderU.pas	Order.java
Photographs.txt	Photographs.txt
Question2P.dpr	Question2.form
Question2P.res	Question2.java
Question2U.dfm	
Question2U.pas	
Question3:	**Question3:**
Question3P.dpr	Question3.form
Question3P.res	Question3.java
Question3U.dfm	
Question3U.pas	

SCENARIO:

Your school is celebrating its 40th anniversary. As part of its year long celebrations the school has decided to host a variety of events. The events include a quiz event, a dinner and dance and a variety of sports and cultural events.

SECTION A

QUESTION 1: GENERAL PROGRAMMING SKILLS

INSTRUCTIONS:

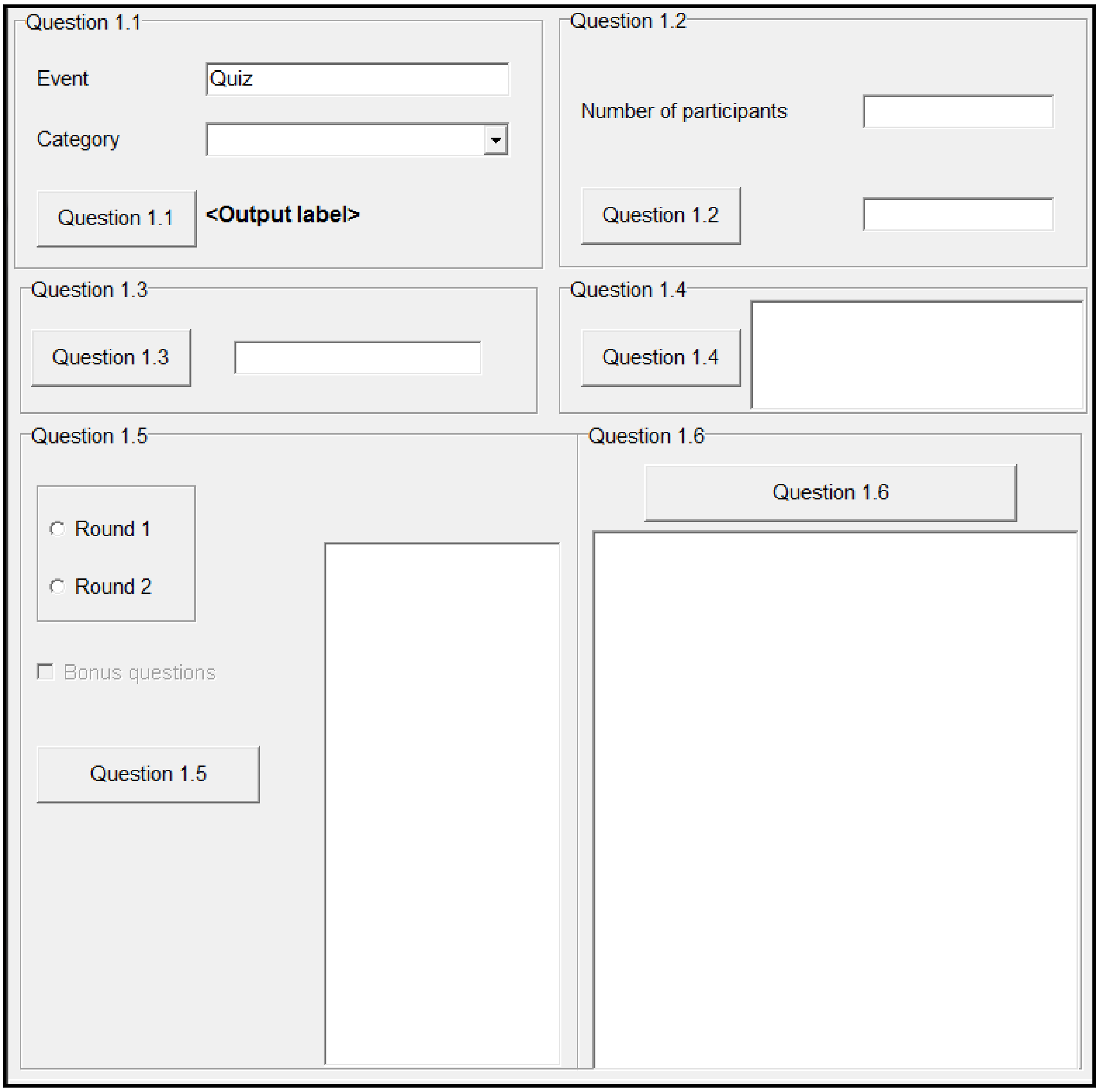
Delphi programmers	Java programmers
• The project **Question1** is provided in the **DelphiDataENG** folder.	• The project **Question1** is provided in the **JavaDataENG** folder.
• Open the incomplete project file **Question1P.dpr** in the **Question1** folder.	• Open the incomplete class called **Question1.java** contained in the folder **Source Packages (src)**, **Question1Package**.
• Add your examination number as a comment in the first line of the main form unit **(Question1U)** file.	• Add your examination number as a comment in the first line of the class **(Question1)**.

Do the following:

- Compile and execute the program. The interface displays six different sections named QUESTION 1.1 to QUESTION 1.6. The program currently has no functionality.
- Complete the code for each section of QUESTION 1 as described in QUESTION 1.1 to QUESTION 1.6 that follow.

Participants in the quiz event can take part in one of several categories. Each category is presented in a different venue. Participants will pay a fee that will be used to fund the event. Tickets will be available to members of the public.

An example of the graphical user interface (GUI) is given on the next page.

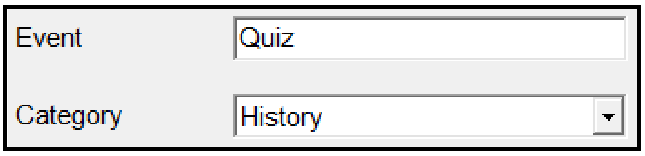


1.1 **Button [Question 1.1]**

The type of event has been supplied in the text box as a default value. The user needs to select a category from the combo box.

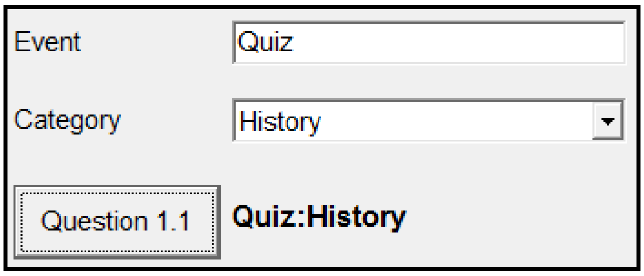
Obtain the type of event and the selected category from the provided components. Create and display a line of text as output that indicates the type of event and the selected category, as shown in the example below.

Example of possible input:



Event	Quiz
Category	History

Example of output:



Event: Quiz

Category: History

Question 1.1 **Quiz:History**

(3)

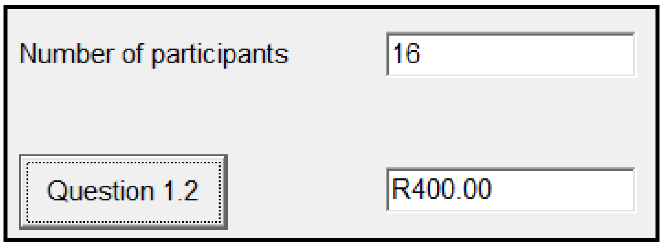
1.2 **Button [Question 1.2]**

Enter the number of participants that will take part in the quiz event. Calculate the amount the participants will contribute towards the fundraising event as follows:

- The first 20 participants will pay an amount of R25,00.
- All other participants will pay an amount of R30,00.

Display the calculated amount in the text box provided.

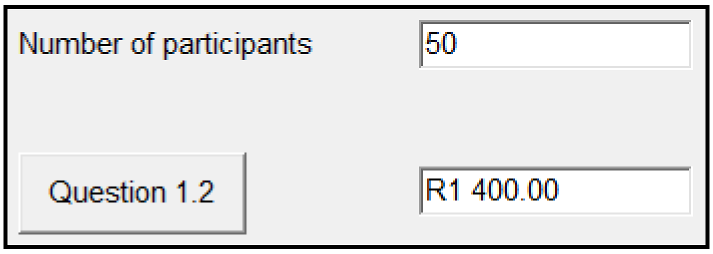
Example 1: Number of participants is 16.



Number of participants: 16

Question 1.2 R400.00

Example 2: Number of participants is 50.



Number of participants: 50

Question 1.2 R1 400.00

NOTE: The currency format of the output may vary. (7)

1.3 **Button [Question 1.3]**

The following formula is used in one of the questions in the science category of the quiz event:

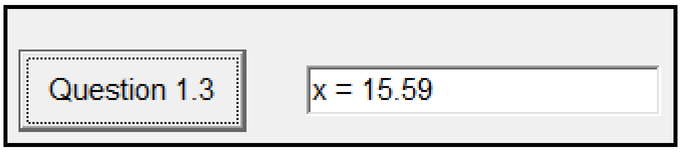
$$x = \sqrt{\text{base}^{\text{exponent}}}$$

Write code for the button to calculate the value of x using the algorithm given below:

Step 1	Input the base value from the keyboard.
Step 2	Input the exponent value from the keyboard.
Step 3	Calculate the power by raising the base to the exponent.
Step 4	Determine the square root of the power calculated in Step 3.
Step 5	Format the value calculated in Step 4 to TWO decimal places.
Step 6	Display the final answer in the text box provided.

NOTE: No marks will be allocated if the given algorithm was not used.

Example of output if the value for the base equals 3 and the value for the exponent equals 5:



x = 15.59

(7)

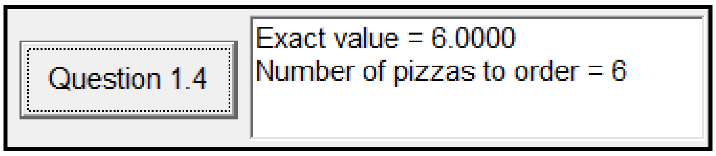
1.4 **Button [Question 1.4]**

The school wants to serve pizza to all participants. It is estimated that each participant will have three slices of pizza. Each pizza has eight slices.

Calculate and display the number of pizzas that should be ordered using the number of participants entered in QUESTION 1.2. The exact value must be formatted to FOUR decimal places. Determine and display the number of whole pizzas to be ordered.

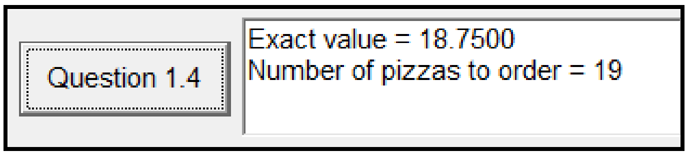
NOTE: If QUESTION 1.2 has not been completed, use the default value set for the number of participants. The default value has been assigned to a global variable as part of the given code.

Example 1: Output if the number of participants entered is 16:



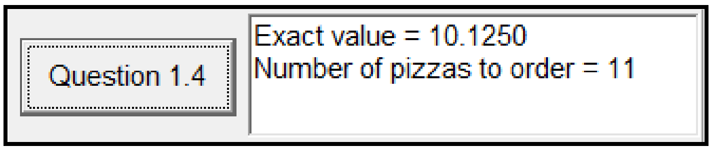
```
Exact value = 6.0000
Number of pizzas to order = 6
```

Example 2: Output if the number of participants entered is 50:



```
Exact value = 18.7500
Number of pizzas to order = 19
```

Example 3: Output if the number of participants entered is 27:



```
Exact value = 10.1250
Number of pizzas to order = 11
```

(9)

1.5 Button [Question 1.5]

1.5.1 Participants in the quiz event can choose to participate in one of two rounds. The number of questions that will be asked will depend on the selected round. The following applies:

- Round 1 consists of a total of five questions.
- Round 2 consists of a total of eight questions.

If the participant selects Round 2, a check box should provide the option to add two bonus questions to the total number of questions to be answered.

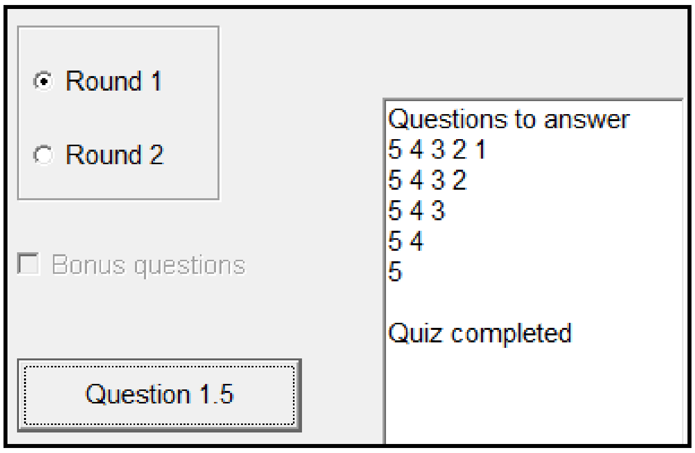
Write code to determine the number of questions that will be asked according to input from the participant. (6)

1.5.2 Code has been provided to display the number of questions still to be answered while the quiz is running. The output must be displayed in the form of a pattern, as shown in the examples that follow.

When the given code is executed an incorrect pattern is displayed due to logic errors in the two loop statements. Correct only the two loop statements in order to display the required output correctly.

Examples of the correct pattern that must be displayed according to the input of the participant are shown on the next page.

Example of the correct output if Round 1 has been selected:



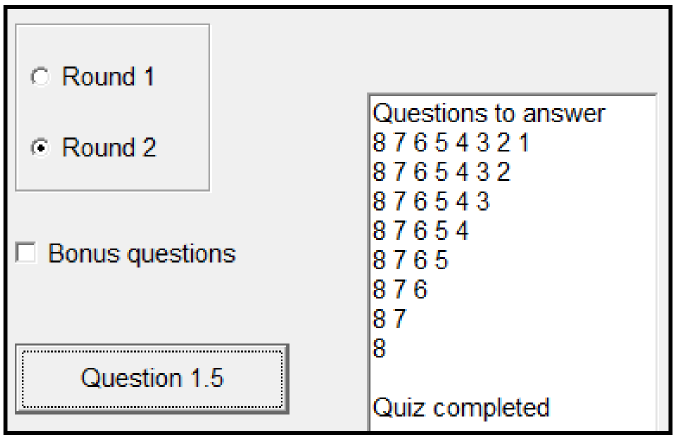
[✓ Round 1] Round 2

Bonus questions

```
Questions to answer
5 4 3 2 1
5 4 3 2
5 4 3
5 4
5

Quiz completed
```

Example of the correct output if Round 2 is selected with no bonus questions:



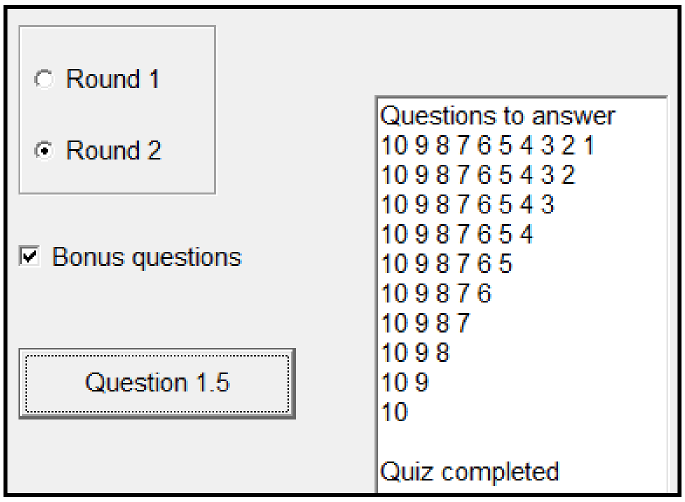
Round 1 [✓ Round 2]

Bonus questions

```
Questions to answer
8 7 6 5 4 3 2 1
8 7 6 5 4 3 2
8 7 6 5 4 3
8 7 6 5 4
8 7 6 5
8 7 6
8 7
8

Quiz completed
```

Example of the correct output if Round 2 was selected with bonus questions added:



Round 1 [✓ Round 2]

[✓ Bonus questions]

```
Questions to answer
10 9 8 7 6 5 4 3 2 1
10 9 8 7 6 5 4 3 2
10 9 8 7 6 5 4 3
10 9 8 7 6 5 4
10 9 8 7 6 5
10 9 8 7 6
10 9 8 7
10 9 8
10 9
10

Quiz completed
```

NOTE: No additional instructions must be added to the provided code.

NOTE: No marks will be allocated if you code your own solution. (4)

1.6 **Button [Question 1.6]**

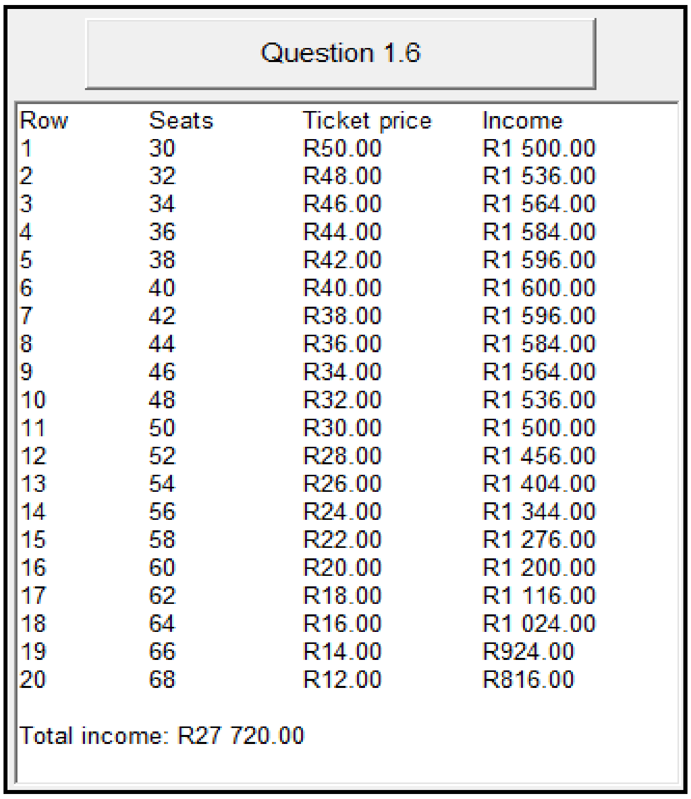
The sports category of the quiz event will be hosted in the sports centre. The seats in the sports centre are arranged as follows:

There are 20 rows of seats numbered from the front of the sports centre in ascending order. The front row (row 1) has 30 seats. Thereafter each row has two more seats than the previous row. That means row 2 has 32 seats, row 3 has 34 seats et cetera.

The price of a ticket in the front row (row 1) is R50,00. The price of tickets decreases by R2,00 per row towards the back of the sport centre. This means the price of the tickets in row 2 is R48,00, in row 3 R46,00, et cetera. All the tickets for the quiz event to be hosted for the sport category have been sold.

Write code to display the row number, the number of seats in the row, the ticket price and the income per row for the entire sport centre. Calculate and display the total income from tickets sales. All monetary values must be formatted using currency.

Example of output:



```
Row        Seats      Ticket price   Income
1          30         R50.00         R1 500.00
2          32         R48.00         R1 536.00
3          34         R46.00         R1 564.00
4          36         R44.00         R1 584.00
5          38         R42.00         R1 596.00
6          40         R40.00         R1 600.00
7          42         R38.00         R1 596.00
8          44         R36.00         R1 584.00
9          46         R34.00         R1 564.00
10         48         R32.00         R1 536.00
11         50         R30.00         R1 500.00
12         52         R28.00         R1 456.00
13         54         R26.00         R1 404.00
14         56         R24.00         R1 344.00
15         58         R22.00         R1 276.00
16         60         R20.00         R1 200.00
17         62         R18.00         R1 116.00
18         64         R16.00         R1 024.00
19         66         R14.00         R924.00
20         68         R12.00         R816.00

Total income: R27 720.00
```

(14)

- Enter your examination number as a comment in the first line of the program file.
- Save the program.
- A printout of the code may be required.

TOTAL SECTION A: 50

SECTION B

QUESTION 2: OBJECT-ORIENTATED PROGRAMMING

A special photo session was organised for the dinner and dance event as part of the celebrations. All attendees are allowed to have group as well as individual photos taken. Afterwards they can place an order for a number of copies of the photos taken.

INSTRUCTIONS:

Delphi programmers	**Java programmers**
• The project **Question2** is provided in the **DelphiDataENG** folder, which contains: o A main form unit called **Question2U.pas** o An incomplete unit file called **OrderU.pas** o A text file **(Photographs.txt)** that contains information on photograph packages ordered	• The project **Question2** is provided in the **JavaDataENG** folder, which contains: o A GUI class file called **Question2.java** o An incomplete object class file called **Order.java** o A text file **(Photographs.txt)** that contains information on photograph packages ordered
• Open the incomplete project file called **Question2P.dpr** in the **Question2** folder.	• Open the incomplete classes called **Question2.java** and **Order.java** in the folder **Source Packages (src)**, **Question2Package**.
• View (Ctrl + F12) the unit file **OrderU.pas** and add your examination number as a comment in the first line of both files **Question2U.pas** and **OrderU.pas**.	• Add your examination number as a comment in the first line of both classes **Question2.java** and **Order.java**.

Do the following:

- Complete the code for each section of QUESTION 2 as described in QUESTION 2.1 and QUESTION 2.2 below.

2.1 An object class called **(TOrder/Order)** which represents an order with some methods, has been provided.

Complete the code in the given order class **(TOrder/Order)** as described in QUESTION 2.1.1 to QUESTION 2.1.6 that follow.

2.1.1 Declare the attributes of the class using the information in the table given below.

Names of attributes		Description
Delphi	**Java**	
fName	name	The name of the person who placed the order
fNumGroup	numGroup	The number of group photographs ordered
fNumSingle	numSingle	The number of individual photographs ordered
fPaid	paid	The character Y or N to indicate whether a payment has been made or not

(3)

2.1.2 (a) Remove the comment symbols from the accessor methods. (1)

(b) Modify the supplied **getPaid** method to return the word 'Yes' or 'No' instead of the character Y or N. (3)

2.1.3 Write code for a constructor that receives the name of the person who placed the order, the number of group and individual photographs ordered and whether a payment has been made or not, as parameters.

Initialise the attributes of the class using the parameters. (3)

2.1.4 Write a method called **calcAmount** to calculate and return the total amount owed. The cost for each photograph is as follows:

- Group photograph: R15,70
- Individual photograph: R12,50 (4)

2.1.5 Discount of five per cent is allowed when the following criteria are met:

- An order of at least four of either individual or group photographs were placed
- A payment has been made

Write a method called **calcDiscount** to calculate and return the discount allowed. (5)

2.1.6 Write a **toString** method to construct a string which contains the information on an order in the format shown below.

```
Name:<name of the person who placed the order>
Group:<number of group photographs ordered>
Individual:<number   of   individual   photographs
ordered>
Paid:<Yes or No>
```

(4)

2.2 The text file **Photographs.txt** contains data needed to compile a report for a person who placed an order.

The name of the school is stored in the first line of the text file. Each line of text thereafter contains the following information:

```
<Initial and surname of the person who placed the order>#<number of group photographs>;<number of individual photographs>;<whether or not a payment has been made (Y – paid, N – not paid)>
```

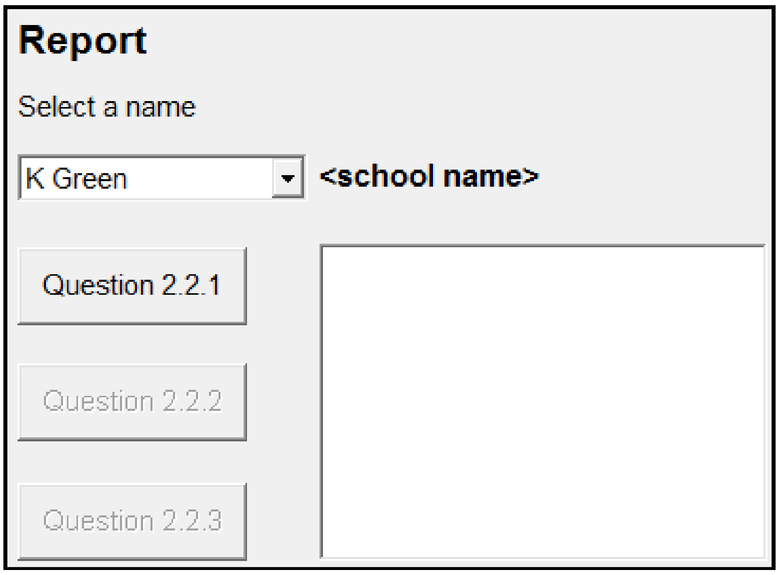
Example of the first six lines of data stored in the text file:

```
School of Excellence
K Green#3;2;N
K Pillay#5;4;N
D Smith#5;2;Y
P Ally#3;4;Y
S Moonsamy#2;0;N
L Ndhlovu#3;4;N
P Smith#2;1;N
L Mahlangu#0;1;N
:
```

The data of the first order saved in the given text file can be interpreted as follows:

- K Green placed the order.
- Three group photographs and two individual photographs were ordered.
- A payment has not been made yet.

An example of the GUI for QUESTION 2:



Do the following to complete the code for the buttons in the main form unit (Delphi)/GUI class (Java) as described below:

2.2.1 **Button [Question 2.2.1]**

Obtain the name of the person who placed the order from the combo box provided.

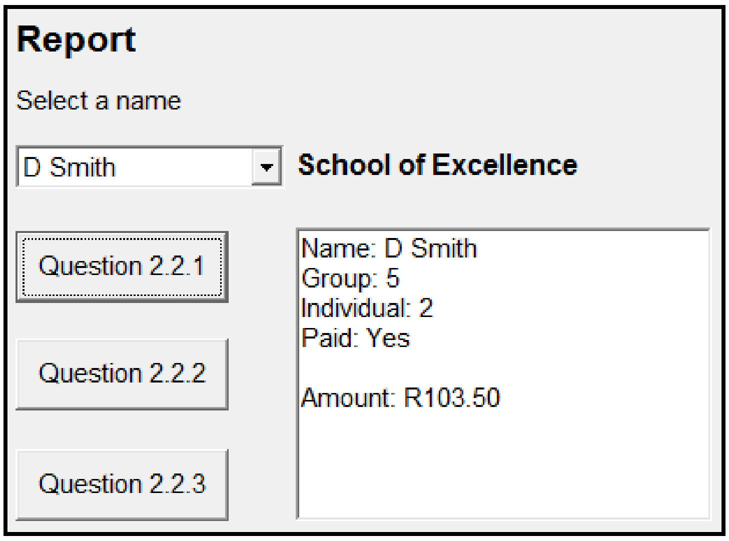
The program must check whether the text file **Photographs.txt** exists.

Display a suitable message if the text file does not exist and close the program.

Do the following if the text file exists:

- Read the name of the school from the file and display the name in the label component provided.
- Use a conditional loop and search the text file for the name of the person that was obtained from the combo box.
  - o If the name is found:
    - – Use the data from the line of text to instantiate a new **Order/TOrder** object. Use the **photoPackage** object variable that has been declared globally as part of the given code to store the object.
    - – Display the details of the order using the **toString** method.
    - – Display the total amount.
    - – Enable the buttons for QUESTION 2.2.2 and QUESTION 2.2.3.
  - o If the name is NOT found, display a suitable message and disable the buttons for QUESTION 2.2.2 and QUESTION 2.2.3.

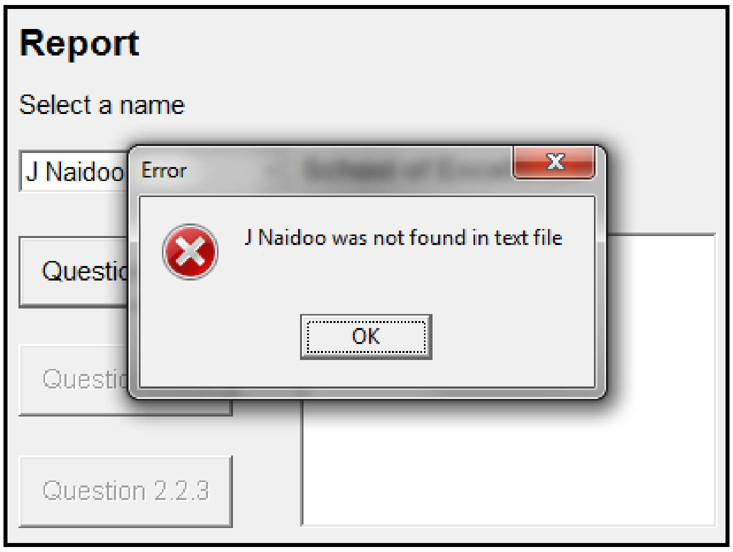
Example of output if the name is found in the text file:



```
Name: D Smith
Group: 5
Individual: 2
Paid: Yes

Amount: R103.50
```

Example of output if the name is not found in the text file:



```
J Naidoo was not found in text file
```

(25)

2.2.2 **Button [Question 2.2.2]**

If a payment has not been made, do the following:

- Create a text file using the name of the person who placed the order as the file name.
- Write the number of group photographs, number of single photographs and total amount outstanding to the text file in the following format:

```
Group: <number of group photographs ordered>
Individual:<number of individual photographs
ordered>
Amount: <the total amount owed>
```

- Use the output area to display a message to indicate that the file has been created.

If a payment has been made, use the output area to display the message 'Account paid'.

Example of output if a payment was not made:

```
Name: S Moonsamy
Group: 2
Individual: 0
Paid: No

Amount: R31.40
Text file created
```

Example of output if a payment was made:

```
Name: D Smith
Group: 5
Individual: 2
Paid: Yes

Amount: R103.50
Account paid
```

(10)

2.2.3 **Button [Question 2.2.3]**

Use the output area to display the amount of discount awarded, formatted to currency.

Example of output for P Ally who qualifies for discount:

```
Name: P Ally
Group: 3
Individual: 4
Paid: Yes

Amount: R97.10
Account paid
Discount: R4.85
```

Example of output for L Ndhlovu who does not qualify for discount:

```
Name: L Ndhlovu
Group: 3
Individual: 4
Paid: No

Amount: R97.10
Text file created
Discount: R0.00
```

(2)

- Enter your examination number as a comment in the first line of the class and the form.
- Save the program.
- Print out the code contained in both the class and the main form if printouts are required.

TOTAL SECTION B: 60

SECTION C

QUESTION 3: PROBLEM-SOLVING

The calendar for the festive year shows that the school will host 45 different events for the various sports and cultural activities at the school.

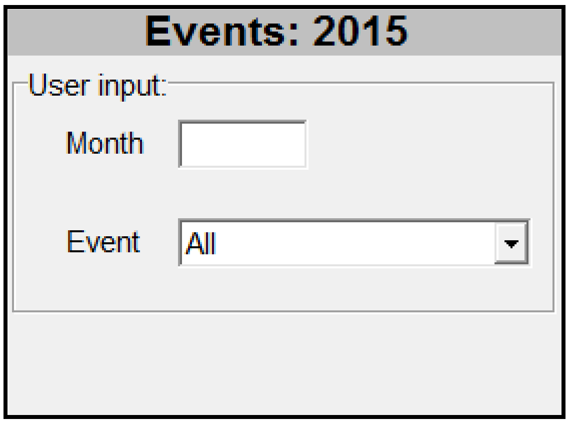
NOTE: Read the following sections carefully before attempting to answer this question:

- **GUI AND DATA SUPPLIED**
- **INSTRUCTIONS**
- **PROGRAM REQUIREMENTS**
- **MARK ALLOCATION**

GUI AND DATA SUPPLIED:

The GUI that is supplied contains components for user input only. You have to add suitable components as you see fit.

Example of GUI supplied:



Data for the events during the year and the scheduled date when each event will take place is supplied in **two parallel arrays**, called **arrEvents** and **arrDates**. Each array contains 45 text strings as explained below:

- The **arrEvents** array stores the names of the 45 events to be hosted during the year. A specific sport or cultural event may take place more than once during the year. Some events are scheduled more than once on the same day.

- The **arrDates** array stores the dates when the corresponding events in the **arrEvents** array will take place. The elements of the **arrDates** array are formatted as text strings using the following date format: YYYY/MM/DD.

An example of the first four elements of the **arrEvents** array:

```
Hockey
Chess
Rugby
Chess
```

An example of the first four elements of the **arrDates** array:

```
2015/10/04
2015/03/20
2015/09/17
2015/03/20
```

The first two elements of corresponding data from the two arrays can be interpreted as follows:

- **`Hockey`** from **arrEvents** and **`2015/10/04`** from **arrDates** → a hockey game is scheduled for 4 October 2015.
- **`Chess`** from **arrEvents** and **`2015/03/20`** from **arrDates** → a chess game is scheduled for 20 March 2015.

INSTRUCTIONS:

Delphi programmers	Java programmers
The project **Question3** is provided in the **DelphiDataEng** folder. • Open the incomplete project file **Question3P.dpr** in the **Question3** folder. The main form unit file is named **Question3U.pas**. • Insert your examination number as a comment in the first line of the unit file **Question3U.pas**.	The project **Question3** is provided in the **JavaDataEng** folder. • Open the incomplete class named **Question3.java** contained in the **Source Packages**, **Question3Package**. • Insert your examination number as a comment in the first line of the class **Question3.java**.

- Add suitable components to answer the question according to the requirements stated in the next section. Supply the components you have added with suitable descriptive names.
- Write code to answer the question according to the requirements stated in the section that follows.

PROGRAM REQUIREMENTS:

The GUI allows the user to enter a month value and/or select an event from the provided combo box. Data from the supplied arrays must be displayed based on the user input.

NOTE:

- The data must be sorted to ensure that all displayed lists of data are displayed in ascending order according to the dates when the events are scheduled to take place.

- If a month value is entered by the user, the value must be validated to be in the range of 1 to 12. If the entered month value is not in the required range, an error message must be displayed and the user must be allowed to enter the month value again until a valid value is entered.

  NOTE: The user is allowed to NOT enter a month value, as explained in the examples that follow.

- All displayed lists must show a heading which contains the name of the event and the month value that was entered, as shown in the examples that follow. The lists must be neatly displayed in columns.

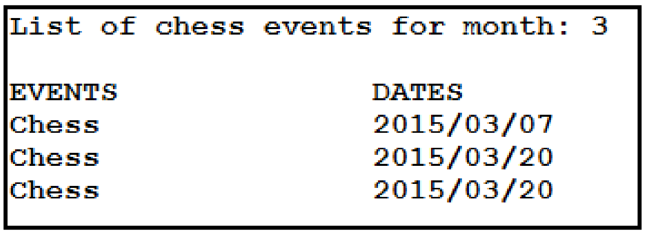
- You are not allowed to modify the content of the supplied arrays manually. Code must be used to manipulate the supplied data according to the requirements during execution of the program.

- The use of good programming techniques and modular design must be applied in the design and coding of your solution.

The following applies to input from the user:

- If the user enters a valid month value and an event from the combo box, the event and dates when that event will take place in that month must be displayed.

  Example of output if the user enters a month value of 3 and selects 'Chess' from the combo box:

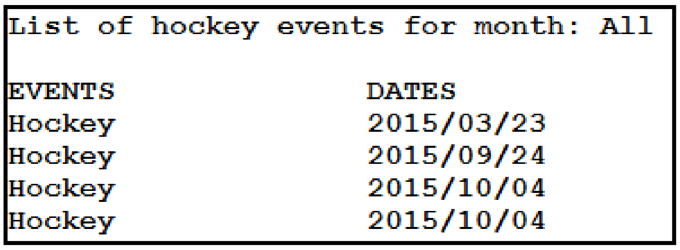


```
List of chess events for month: 3

EVENTS                    DATES
Chess                     2015/03/07
Chess                     2015/03/20
Chess                     2015/03/20
```

- If the user does not enter a value in the month text box and only selects an event from the combo box, the event and all the dates when that event will take place during the year must be displayed. The heading must indicate the month value as 'All'.

  Example of output when no month value was entered and the 'Hockey' event was selected:

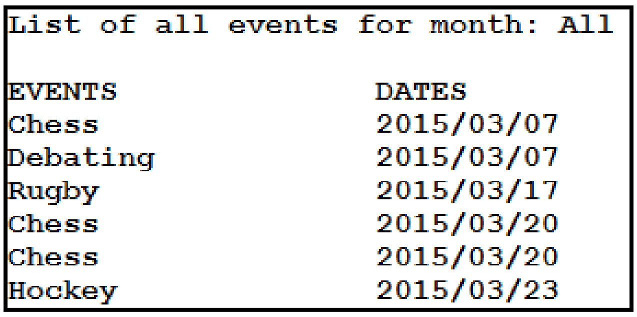


```
List of hockey events for month: All

EVENTS                  DATES
Hockey                  2015/03/23
Hockey                  2015/09/24
Hockey                  2015/10/04
Hockey                  2015/10/04
```

- If the user does not enter a month value and selects the 'All' event from the combo box, all the events and the dates when each event will take place must be displayed.

  Example of the output of the first six events when the user does not enter a month value and does not select an event from the combo box:



```
List of all events for month: All

EVENTS                  DATES
Chess                   2015/03/07
Debating                2015/03/07
Rugby                   2015/03/17
Chess                   2015/03/20
Chess                   2015/03/20
Hockey                  2015/03/23
```

The supplied data provide for some of the events to take place several times on one day. For example two chess events will take place on 2015/03/20, three soccer events will take place on 2015/06/03, et cetera.

The following additional option must be added to the program:

Remove duplicate dates (for a particular event)

The following must happen when this option is executed:

- You must add/use a suitable component to allow the user to enter the name of an event.

- No validation of the type of event that was entered is required. If the user enters an event that is not contained in **arrEvents** a blank list must simply be displayed.

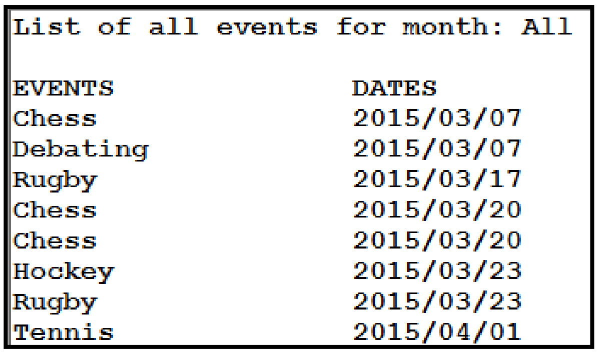
- The program must execute correctly when entering the type of event using lower and/or upper case characters.

- Programming code must ensure that the event that was entered does not appear more than once on the same date on the displayed list.

Examples of output before and after dates of the specified event that was entered has been removed:

Example 1:

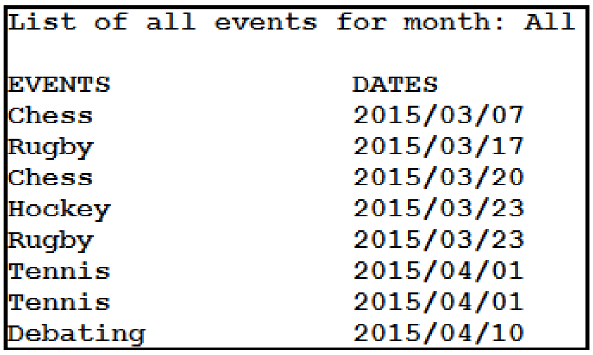
Example of output of the first eight events and corresponding dates where all events and dates have been selected and before any dates have been removed:



```
List of all events for month: All

EVENTS                  DATES
Chess                   2015/03/07
Debating                2015/03/07
Rugby                   2015/03/17
Chess                   2015/03/20
Chess                   2015/03/20
Hockey                  2015/03/23
Rugby                   2015/03/23
Tennis                  2015/04/01
```

Example of output of the first eight events when the user enters 'Chess' as the event and activate the **Remove duplicate dates** option to remove duplicate dates for the chess event:

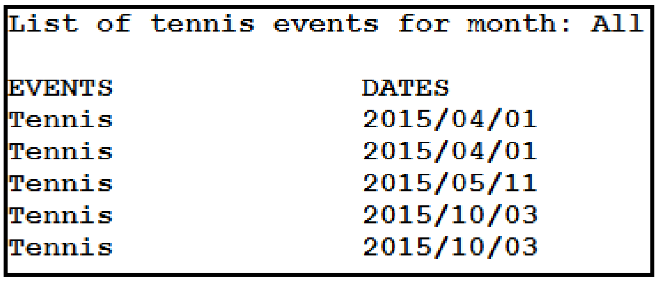


```
List of all events for month: All

EVENTS                  DATES
Chess                   2015/03/07
Rugby                   2015/03/17
Chess                   2015/03/20
Hockey                  2015/03/23
Rugby                   2015/03/23
Tennis                  2015/04/01
Tennis                  2015/04/01
Debating                2015/04/10
```

Example 2:

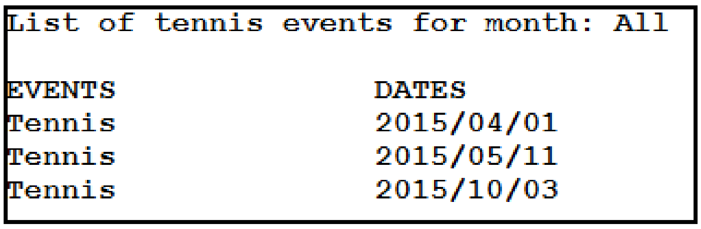
Example of output when all the 'Tennis' events and corresponding dates are displayed before any dates have been removed:



```
List of tennis events for month: All

EVENTS                  DATES
Tennis                  2015/04/01
Tennis                  2015/04/01
Tennis                  2015/05/11
Tennis                  2015/10/03
Tennis                  2015/10/03
```

Output of the list when the user enters 'Tennis' as the event and activates the **Remove duplicate dates** option:



```
List of tennis events for month: All

EVENTS                  DATES
Tennis                  2015/04/01
Tennis                  2015/05/11
Tennis                  2015/10/03
```

MARK ALLOCATION:

Requirements	**Maximum marks**
Application of good programming techniques and modular design	5
User input validation	8
Sorting of arrays	8
Display of output	9
Remove duplicate elements	10

- Enter your examination number as a comment in the first line of the program file.
- Save the program.
- A printout of the code may be required.

TOTAL SECTION C: 40
GRAND TOTAL: 150