basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

NATIONAL SENIOR CERTIFICATE

GRADE 12

INFORMATION TECHNOLOGY P1

NOVEMBER 2015

MARKS: 150

TIME: 3 hours

This question paper consists of 22 pages.

INSTRUCTIONS AND INFORMATION

1. This question 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 question 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.

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. Data structures which are not supplied must be defined by you, the programmer.

9. 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.

10. Make sure that your examination number appears as a comment in every program that you code, as well as on every event indicated.

11. 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.

12. 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.

13. The files that you need to complete this question paper have been given to you on a 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.

NOTE:

- Delphi candidates must use the file **DelphiDataENGNov2015.exe**.
- Java candidates must use the file **JavaDataENGNov2015.exe**.

Do the following:

- Double click on the password-protected executable file.
- Click on the extract button.
- Enter the following password: **GymCentre54#$**

Once extracted, the following list of files will be available in the folder **DelphiDataENGNov2015/JavaDataENGNov2015**:

Delphi Files	**Java (Netbeans) Files**
Question1:	**Question1:**
Question1P.dpr	Question1.form
Question1P.res	Question1.java
Question1U.dfm	
Question1U.pas	
Question2:	**Question2:**
DataQ2.txt	DataQ2.txt
Question2P.dpr	Question2.form
Question2P.res	Question2.java
Question2U.dfm	Student.java
Question2U.pas	
StudentU.pas	
Question3:	**Question3:**
Question3P.dpr	Question3.form
Question3P.res	Question3.java
Question3U.dfm	
Question3U.pas	

SECTION A

QUESTION 1: GENERAL PROGRAMMING SKILLS

The Eagle Star Gym at the local shopping mall is used by the community to keep fit.

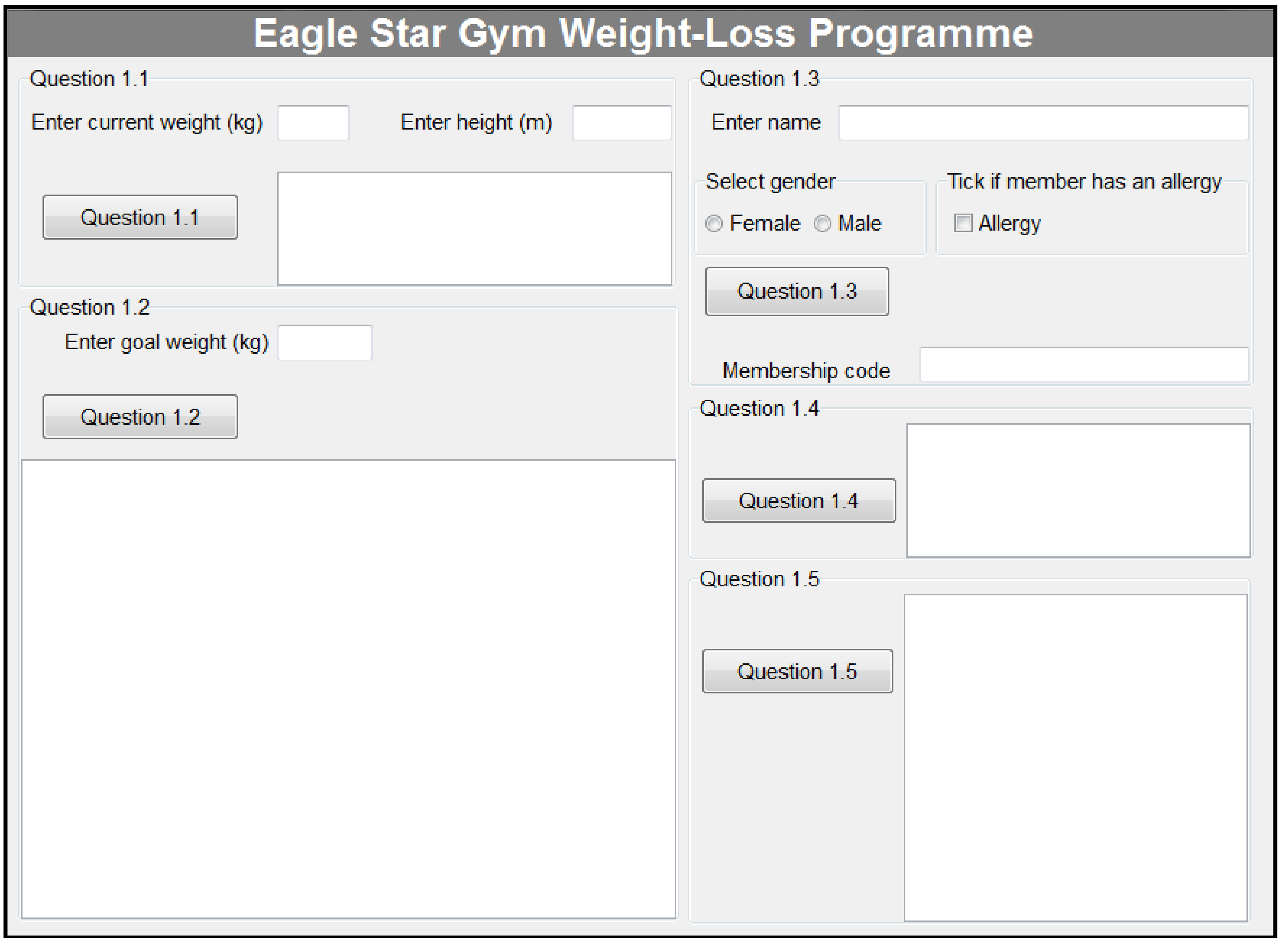
This gymnasium has recently piloted a special weight-loss programme for a group of twenty people. A computer program is being designed to assist in the administration of this special weight-loss programme.

INSTRUCTIONS:

Delphi programmers	Java programmers
• The project **Question1** is provided in the **DelphiDataENGNov2015** folder.	• The project **Question1** is provided in the **JavaDataENGNov2015** folder.
• Open the incomplete project file called **Question1P.dpr** in the **Question1** folder.	• Open the incomplete class called **Question1.java** in the folder **Source Packages (src)**, **Question1Package**.
• Add your examination number as a comment in the first line of the main form unit called **Question1U.pas**.	• Add your examination number as a comment in the first line of the class called **Question1.java**.

Do the following:

- Compile and execute the program. The GUI displays five sections labelled QUESTION 1.1 to QUESTION 1.5. The program has no functionality currently. An example of the GUI is given on the next page.



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

1.1 When a group member enrols for the weight-loss programme, his/her body mass index (BMI) is calculated using the following formula:

$$\text{BMI} = \frac{\text{weight}}{\text{height}^2}$$

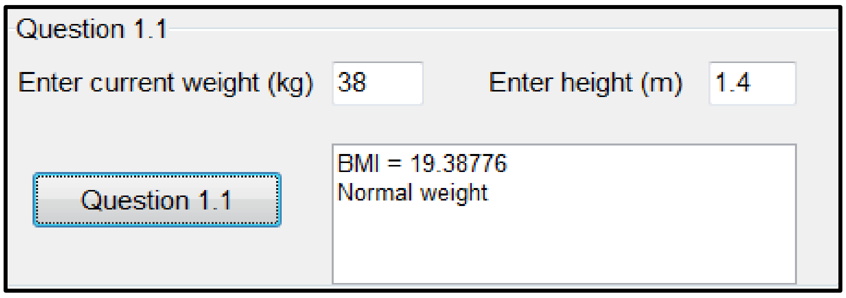
A group member is classified as being overweight, of normal weight or underweight based on his/her BMI according to the criteria in the table below.

BMI	WEIGHT DESCRIPTION
< 18,5	Underweight
18,5 to 25	Normal weight
> 25	Overweight

The user is required to enter the weight and height of the group member in kilograms and metres respectively in the text boxes provided.

Write code to obtain the weight and height from the text boxes provided and calculate the BMI of the group member. Use the output area to display the BMI and a message indicating whether the member is underweight, of normal weight or overweight according to the classification in the given table. Format values to FIVE decimal places.

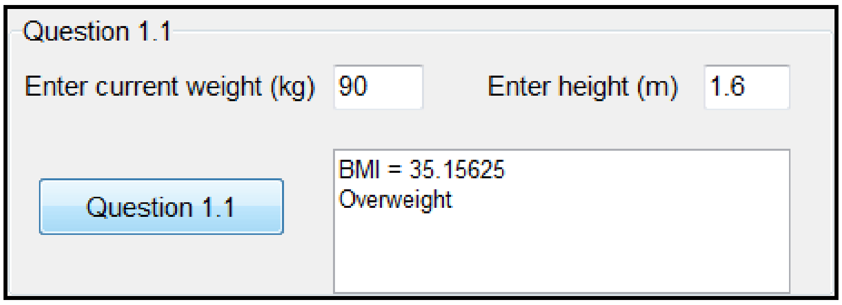
Example of output if the weight is 38 kg and the height is 1,4 m:



Enter current weight (kg): 38
Enter height (m): 1.4

```
BMI = 19.38776
Normal weight
```

Example of output if the weight is 90 kg and the height is 1,6 m:



Enter current weight (kg): 90
Enter height (m): 1.6

```
BMI = 35.15625
Overweight
```

(8)

1.2 A member who has enrolled for the weight-loss programme wants to know how many days it would take to reach his goal weight. The user is required to enter his goal weight in the text box provided.

Write code to do the following:

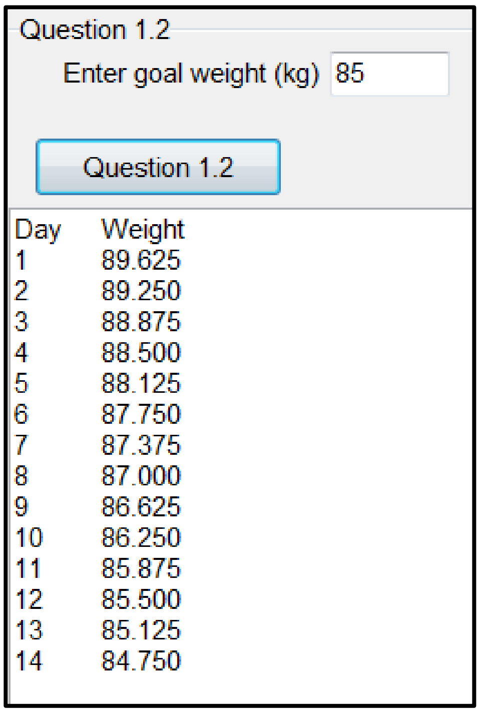
Obtain the goal weight from the text box and use the current weight of the member obtained in QUESTION 1.1 to calculate and display the output, as shown in the example of output on the next page.

The following applies:

- Assume that the weight loss is 375 grams (0,375 kg) per day, starting on the first day.
- If the goal weight is less than the current weight, display the day number and the weight of the member from the first day until he reaches his goal weight.
- Display the message 'Invalid value entered' if the goal weight is greater than or equal to the member's current weight.

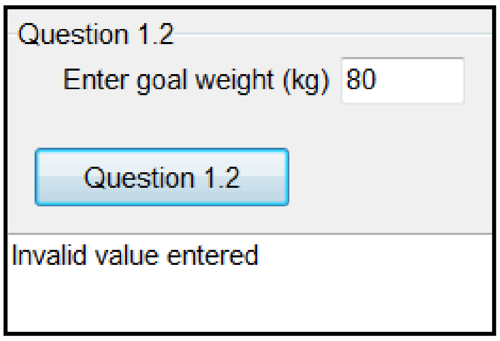
Round off all weight values that are displayed to THREE decimal places.

Example of output if the current weight is 90 kg and the goal weight is 85 kg:



```
Day    Weight
1      89.625
2      89.250
3      88.875
4      88.500
5      88.125
6      87.750
7      87.375
8      87.000
9      86.625
10     86.250
11     85.875
12     85.500
13     85.125
14     84.750
```

Example of output if the current weight is 70 kg and the goal weight is 80 kg:



```
Invalid value entered
```

(7)

1.3 A membership code must be compiled for each person joining the weight-loss programme.

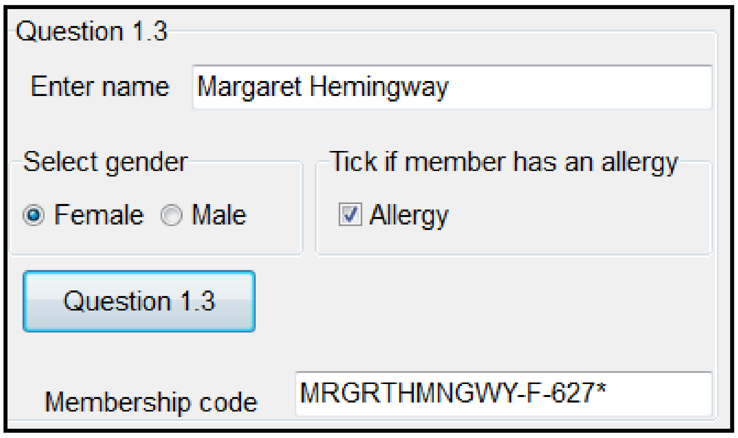
Write code to do the following:

- Obtain the full name of the member from the text box provided.
- Obtain the gender of the member from the radio buttons provided.
- Determine whether the check box for an allergy has been ticked.
- The membership code for the member consists of three parts that are compiled as follows:
  - Part 1
    - Convert the full name to upper case.
    - Remove all vowels and spaces.
  - Part 2
    - Add the characters '-M-' or '-F-' to Part 1 of the membership code, depending on whether the member is male or female.

- Part 3
  - Add a check number to the membership code as a three-digit number. The check number is constructed as follows:
    - Generate a random number in the range 1 to 9 (1 and 9 included). This will be the first digit.
    - Add the value of 10 to the random number.
    - Determine the number of letters that the full name consists of without the vowels (first part of the member code). Add the value to the current value. The total value will be the last two digits of the check number.

    Example: In Part 1, the full name Peter Smith will become PTRSMTH (7 letters).
    If the random number is 4, then the last two digits will be 21 (4 + 10 + 7 = 21).
    The check number will then be '421'.
  - Add the asterisk character (*) to the membership code if the member has an allergy.

Example of output for a female member named Margaret Hemingway, who has an allergy:



Enter name: Margaret Hemingway
Gender: [✓ Female] Male
[✓ Allergy]
Membership code: MRGRTHMNGWY-F-627*

NOTE: Due to the random selection of membership codes, the membership codes displayed in the screenshot above may differ from the membership codes displayed by your program. (14)

1.4 An array called **arrMemberCodes** containing 20 membership codes is provided.

The first ten membership codes in the array are shown below.

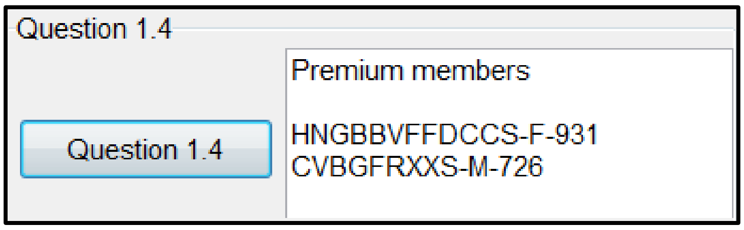
```
PRTHNMM-M-421
LYYHNBB-F-623*
DFGQWJJK-M-220*
NBVGTYY-F-926
NBGTRFSSD-F-322*
NJKYTRRTG-M-928
JBHGTYGFTR-F-121
HGTYRJJ-F-522*
KJHYTGFDDRWQ-M-830
NHYTRFDDD-M-221*
```

Two members from the group contained in the array must be randomly selected for Premium membership. Premium membership allows members access to any Eagle Star Gym in South Africa.

Write code to randomly select TWO membership codes. One of the selected members must be male and the other selected member must be female.

Display the membership codes of the two randomly selected members in the output area provided.

Example of the contents of the output area:



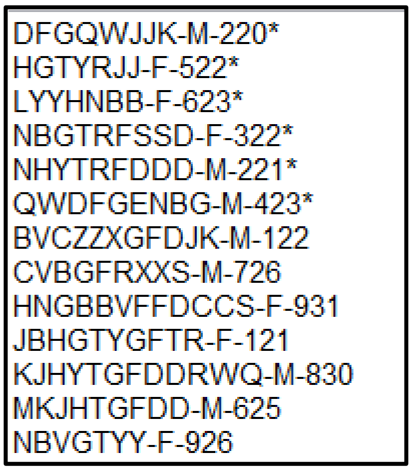
```
Premium members

HNGBBVFFDCCS-F-931
CVBGFRXXS-M-726
```

NOTE: Due to the random selection of membership codes, the membership codes displayed in the screenshot above may differ from the membership codes displayed by your program. (9)

1.5 The organisers want the 20 membership codes in the **arrMemberCodes** array to be sorted in alphabetical order. They requested that all the members with an allergy must be displayed at the top of the list.

Write code to sort the membership codes in the array in alphabetical order. From the sorted array, extract all members with an allergy and display these at the top of the list, followed by all the other members.

Example of output of the first 13 membership codes after they have been sorted alphabetically:



```
DFGQWJJK-M-220*
HGTYRJJ-F-522*
LYYHNBB-F-623*
NBGTRFSSD-F-322*
NHYTRFDDD-M-221*
QWDFGENBG-M-423*
BVCZZXGFDJK-M-122
CVBGFRXXS-M-726
HNGBBVFFDCCS-F-931
JBHGTYGFTR-F-121
KJHYTGFDDRWQ-M-830
MKJHTGFDD-M-625
NBVGTYY-F-926
```

(12)

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

TOTAL SECTION A: 50

SECTION B

QUESTION 2: OBJECT-ORIENTATED PROGRAMMING

Eagle Star Gym offers a variety of instructor courses to its members. A member can enrol for one instructor course at a time. Members who are enrolled for an instructor's course are referred to as students.

INSTRUCTIONS:

Delphi programmers	Java programmers
• The project **Question2** is provided in the **DelphiDataENGNov2015** folder, which contains: o A main form unit called **Question2U.pas** o An incomplete unit file called **StudentU.pas** o A text file called **DataQ2.txt** that contains information on a member enrolled for an instructor course • Open the incomplete project file called **Question2P.dpr** and the class called **StudentU.pas** in the **Question2** folder. • Add your examination number as a comment in the first line of both files **Question2U.pas** and **StudentU.pas**.	• The project **Question2** is provided in the **JavaDataENGNov2015** folder, which contains: o A GUI class file called **Question2.java** o An incomplete object class file called **Student.java** o A text file called **DataQ2.txt** that contains information on a member enrolled for an instructor course • Open the incomplete classes called **Question2.java** and **Student.java** in the folder **Source Packages (src)**, **Question2Package**. • Add your examination number as a comment in the first line of both classes **Question2.java** and **Student.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 A partially completed object class called **TStudent/Student** has been provided.

Complete the code in the given student class called **TStudent/Student**, as described in QUESTION 2.1.1 to QUESTION 2.1.6.

The table below contains descriptions of the attributes of a member enrolled as a student for an instructor course.

Names of attributes		Description
Delphi	**Java**	
fName	name	Surname and first name of the student
fRegCode	regCode	Unique six-character registration code assigned to each student enrolled for an instructor course
fExpiryDate	expiryDate	Date on which the registration for the instructor course will expire, in the format YYYY/MM/DD
fSessionsCompleted	sessionsCompleted	Number of training sessions that the student has completed

2.1.1 Remove the comment signs from the statements contained in the **private** method called **determineExpDate**. The method receives the date on which the student registered for the instructor course as a parameter. The student has to complete the course in a period of two years from the date of registration. Therefore, registration for the course expires two years from the date of registration. Use the parameter value to determine and return the date on which the registration expires. (4)

2.1.2 Write code for a constructor method to do the following:

- Receive the name and the registration code of the student as well as the date of registration as parameters.
- Assign the relevant parameter values to the **fName/name** and **fRegCode/regCode** attributes.
- Use the date of registration as an argument and call the private method **determineExpDate** to set the **fExpiryDate/expiryDate** attribute.
- Set the **fSessionsCompleted/sessionsCompleted** attribute to zero. (5)

2.1.3 Write a mutator method called **setSessionsCompleted** to receive a value as a parameter value and set the **fSessionsCompleted/ sessionsCompleted** attribute to the value. (2)

2.1.4 Write a method called **increaseSessionsCompleted** to increase the number of sessions completed by 1. (2)

2.1.5 Write a method called **evaluateProgress** that receives the total number of sessions that the instructor course consists of as a parameter and returns a string indicating the progress of a student.

The string must be compiled as follows:

- If the percentage of sessions completed by a student is greater than 75% of the total number of sessions, return a string formatted as follows:

```
<Name of student> qualifies as an instructor
```

Example of the compiled string:

```
Peter Drake qualifies as an instructor
```

- If the percentage of sessions completed by the student is less than or equal to 75% of the total number of sessions, return a string formatted as follows:

```
Percentage completed:<percentage formatted to
two decimal places>%
```

Example of the compiled string:

```
Percentage completed: 67.24%
```

(8)

2.1.6 Write a **toString** method to display the details of a student in the following format:

```
<name of student> [<registration code>]
Expiry date: <expiry date>
Completed sessions: <number of sessions completed>
```

Example of output:

```
Peter Drake [PD1203]
Expiry date: 2015/11/12
Completed sessions: 0
```

(4)

2.2 The text file **DataQ2.txt** contains an unknown number of lines of information on students who attended the various instructor courses daily. Each line of information contains data for a single training session attended by a student in the following format:

```
<registration code> trained on <date the student attended the training
session>#<session completed or not completed>
```

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

```
PD1203 trained on 2013/12/07#Not completed
CS3011 trained on 2015/10/19#Completed
SM2102 trained on 2015/09/28#Not completed
PD1203 trained on 2014/01/09#Completed
SM2102 trained on 2015/10/03#Not completed
CJ1506 trained on 2015/10/19#Completed
```

The data in the first two lines of text can be interpreted as follows:

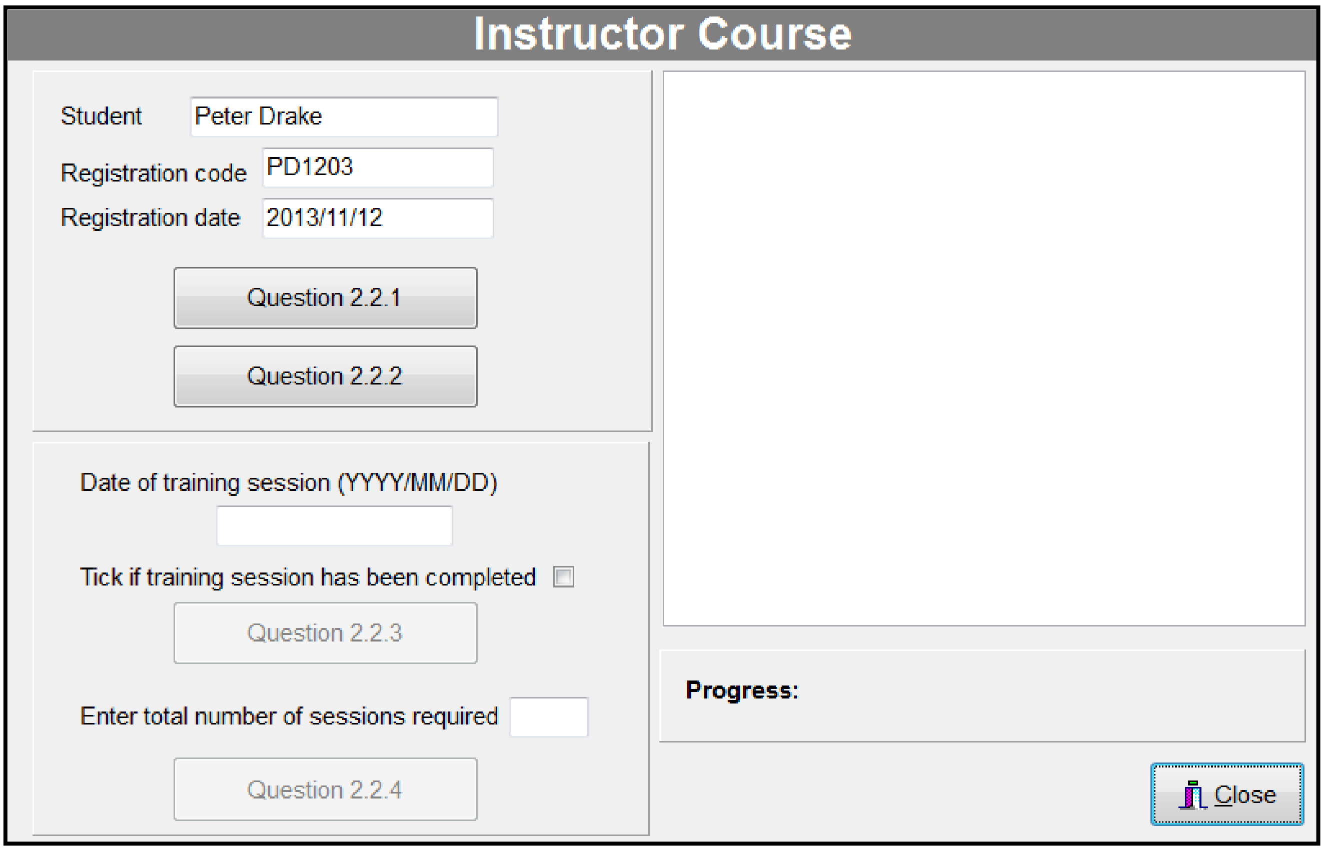
Line 1: `PD1203 trained on 2013/12/07#Not completed`

- The student with code PD1203 trained on 7 December 2013 and did not complete the session.

Line 2: `CS3011 trained on 2015/10/19#Completed`

- The student with code CS3011 trained on 19 October 2015 and completed the session.

Example of the GUI for QUESTION 2:



Student: Peter Drake
Registration code: PD1203
Registration date: 2013/11/12

Date of training session (YYYY/MM/DD)

Tick if training session has been completed

Enter total number of sessions required

Progress:

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]**

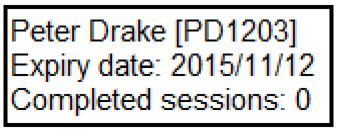
Peter Drake has registered for the 'Cardio and Muscle Building' instructor course. A registration card that contains the following information was issued to him:

Course:	Cardio and Muscle Building
Student:	Peter Drake
Registration code:	PD1203
Registration date:	2013/11/12

Write code to do the following:

- Use the values provided in the relevant text boxes to instantiate the object **objStudent**.
- Display the details of the student using the **toString** method.

Example of output for the student called Peter Drake:



```
Peter Drake [PD1203]
Expiry date: 2015/11/12
Completed sessions: 0
```

(3)

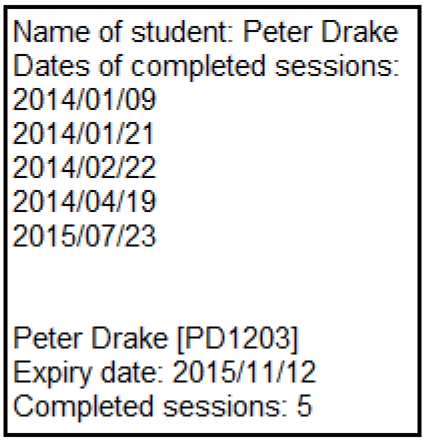
2.2.2 **Button [Question 2.2.2]**

Write code to do the following:

- Check whether or not the **DataQ2.txt** text file exists.
- If the text file does NOT exist, display a suitable message and close the program.
- If the text file exists, do the following:
  - o Display the name of the student.
  - o Use the registration code of the student object to search and count the number of completed sessions of the student in the text file. Display the date of each training session completed by the student.

- Use an appropriate method to set the attribute for the number of completed sessions to the latest number counted.
- Enable the buttons for QUESTION 2.2.3 and QUESTION 2.2.4.

Example of output for Peter Drake:

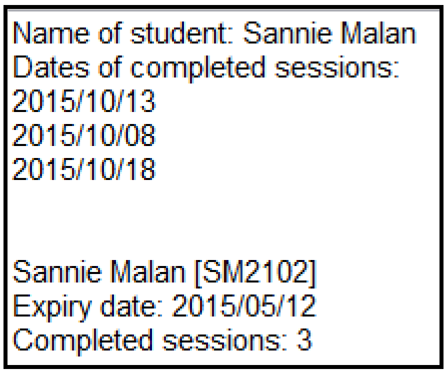


```
Name of student: Peter Drake
Dates of completed sessions:
2014/01/09
2014/01/21
2014/02/22
2014/04/19
2015/07/23


Peter Drake [PD1203]
Expiry date: 2015/11/12
Completed sessions: 5
```

NOTE: The object can also be tested with the following student data:

Name: Sannie Malan
Registration code: SM2102
Registration date: 2013/05/12

Example of output for Sannie Malan:



```
Name of student: Sannie Malan
Dates of completed sessions:
2015/10/13
2015/10/08
2015/10/18


Sannie Malan [SM2102]
Expiry date: 2015/05/12
Completed sessions: 3
```

(16)

2.2.3 **Button [Question 2.2.3]**

The student Peter Drake attended another training session. Enter the date of the training session in the text box provided. Use the check box provided to indicate whether he completed the training session or not.

The information of the new training session must be added to the text file called **DataQ2.txt** in the same format as the lines of text already saved in the text file.

The format of each line in the text file is as follows:

```
<Registration code> trained on <date when the
student attended the training session>#<session
completed or not completed>
```

Write code to do the following:

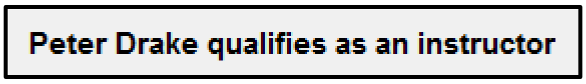
- Compile a string to be written to the text file according to the required format. Add the string to the **DataQ2.txt** text file.
- Display a message indicating that the information has been added to the text file.
- Update the number of training sessions completed by the student if the session was completed.
- Use the **toString** method to display the updated information of the student. (10)

2.2.4 **Button [Question 2.2.4]**

A student qualifies as an instructor if he/she has completed more than 75% of the total number of training sessions. The user is required to enter the total number of training sessions for the instructor course in the text box provided. Use the **evaluateProgress** method to determine the student's progress and display the message returned by the method.

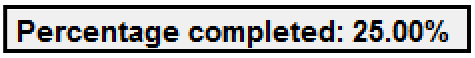
Display the progress of the student in the 'Progress' label provided, as shown in the examples of output below.

Example of 'Progress' label if the student has met the criteria and completed more than 75% of the total number of training sessions:



```
Peter Drake qualifies as an instructor
```

Example of 'Progress' label if the student only completed six out of a total of 20 training sessions:



```
Percentage completed: 25.00%
```

(3)

- Enter your examination number as a comment in the first line of both files containing your code.
- Save all the files.
- A printout of the code for both files may be required.

TOTAL SECTION B: 57

SECTION C

QUESTION 3: PROBLEM-SOLVING PROGRAMMING

Eagle Star Gym is hosting a health and fitness seminar over a period of four days. Six workshops on different topics will be conducted daily.

The topics for the six workshops are Aerobics, Bodybuilding, Cardio, Dancing, Energy Supplements and First Aid.

A maximum of 20 members can book for any workshop on any given day.

INSTRUCTIONS:

Delphi programmers	Java programmers
• The project **Question3** is provided in the **DelphiDataENGNov2015** folder.	• The project **Question3** is provided in the **JavaDataENGNov2015** folder.
• Open the incomplete project file called **Question3P.dpr** in the **Question3** folder. The main form unit is called **Question3U.pas**.	• Open the incomplete class called **Question3.java** in the folder **Source Packages (src)**, **Question3Package**.
• Add your examination number as a comment in the first line of the main form unit called **Question3U.pas**.	• Add your examination number as a comment in the first line of the class called **Question3.java**.

You are required to design a program that must be able to do the following:

- Display the bookings in neatly formatted rows and columns.
- Allow the user to book for a workshop.
- Determine the number of cases of bottled water required. (Members who attend workshops each receive a bottle of water per day.)
- Cancel a workshop and the bookings made for the workshop.

Read the following sections before attempting the solution:

- GUI and data supplied
- Program requirements
- Mark allocation

NOTE:

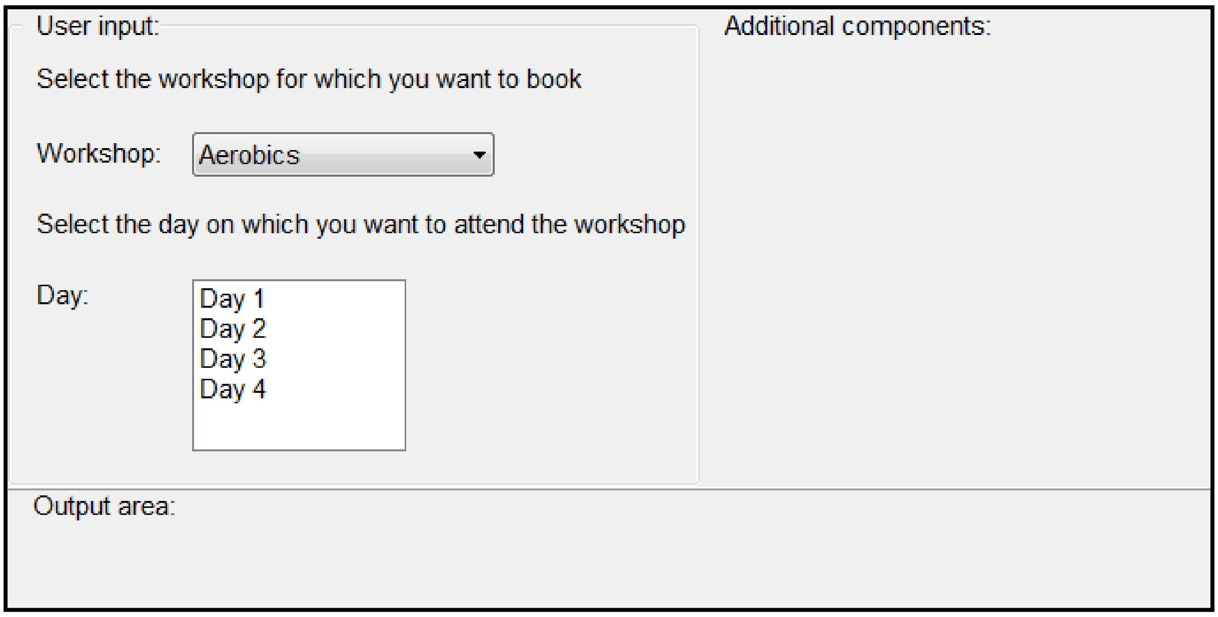
- You may NOT modify data supplied manually. Code must be added to manipulate the data supplied according to the requirements.
- The use of good programming techniques and modular design must be applied in the design and coding of your solution.

GUI AND DATA SUPPLIED

GUI

The GUI contains components for user input only. Use the program requirements on the next page to decide on suitable additional components required. Add these components to the section on the GUI labelled **Additional components**. Add output components to the section on the GUI labelled **Output area**.

Example of the GUI provided:


User input:

Select the workshop for which you want to book

Workshop: Aerobics

Select the day on which you want to attend the workshop

Day:

Day 1
Day 2
Day 3
Day 4

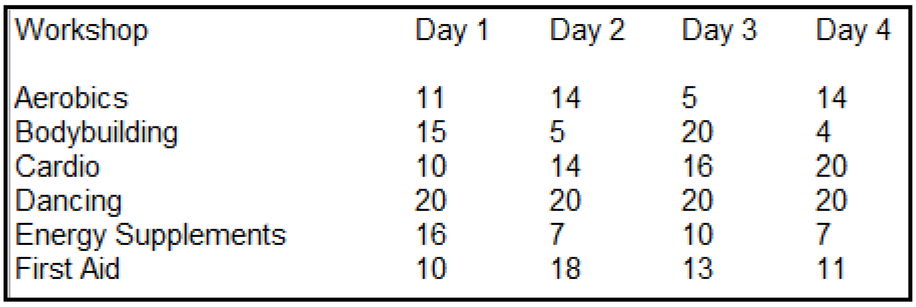
Additional components:

Output area:

DATA

The workshop topics are stored in a one-dimensional array called **arrWorkshops**.

The number of bookings made for the various workshops is stored in a two-dimensional array called **arrBookings**. The table below is a representation of the number of members who have booked for the various workshops already:



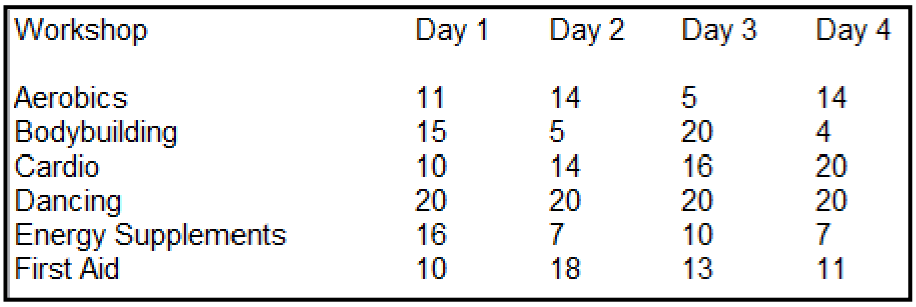
Workshop	Day 1	Day 2	Day 3	Day 4
Aerobics	11	14	5	14
Bodybuilding	15	5	20	4
Cardio	10	14	16	20
Dancing	20	20	20	20
Energy Supplements	16	7	10	7
First Aid	10	18	13	11

PROGRAM REQUIREMENTS

Display bookings:

- The program must provide a suitable method to display the contents of the arrays **arrWorkshops** and **arrBookings** neatly in rows and columns.
- Provide suitable headings and display the data neatly in rows and columns.

  Example of output that the method must provide to display the workshop topics and the number of bookings for each day:

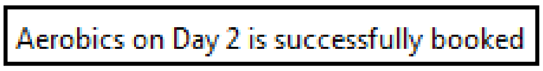


```
Workshop              Day 1   Day 2   Day 3   Day 4

Aerobics              11      14      5       14
Bodybuilding          15      5       20      4
Cardio                10      14      16      20
Dancing               20      20      20      20
Energy Supplements    16      7       10      7
First Aid             10      18      13      11
```

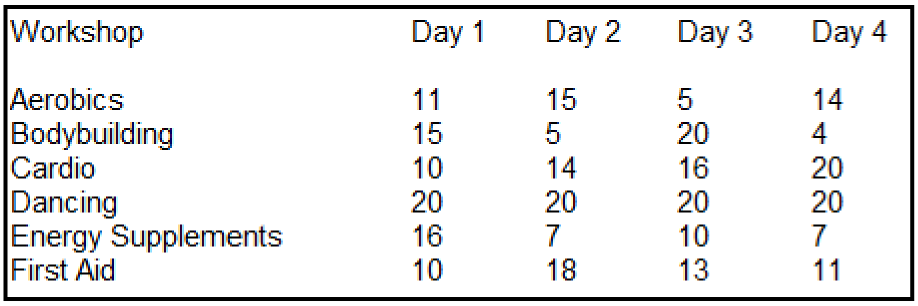
Book a workshop:

- A workshop can accommodate a maximum of 20 bookings.
- The user has to select a workshop and a day for attending the workshop in order to make a booking.
- If the workshop is not fully booked, update the booking for the selected workshop. Display a suitable message indicating that the booking was made and display the updated booking.
- If the selected workshop is fully booked, display a suitable message to inform the user that the selected workshop is fully booked.

  Example of output if a member selects to book for the Aerobics workshop on Day 2:



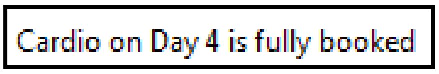
```
Aerobics on Day 2 is successfully booked
```



```
Workshop              Day 1   Day 2   Day 3   Day 4

Aerobics              11      15      5       14
Bodybuilding          15      5       20      4
Cardio                10      14      16      20
Dancing               20      20      20      20
Energy Supplements    16      7       10      7
First Aid             10      18      13      11
```

NOTE: The number of bookings for Aerobics on Day 2 increased from 14 to 15.

Example of output if a member selects to book for the Cardio workshop on Day 4, which is fully booked already:

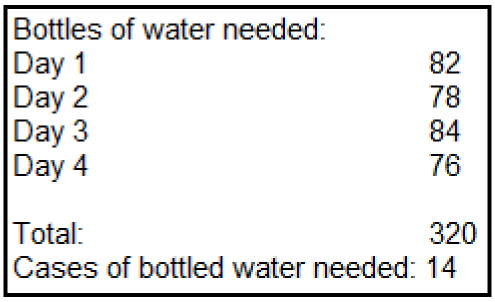


```
Cardio on Day 4 is fully booked
```

Determine the number of cases of bottled water to be ordered:

- All members attending the workshops receive a free bottle of water every day. Calculate the number of bottles of water required for each day of the four-day period. Display the day numbers and the total number of bottles of water required each day.
- The management also wants to know how many full cases of bottled water in total must be purchased for the four-day period. Each full case contains 24 bottles of water.

  Example of output of the number of bottles of water and the full cases of bottled water needed, using the original data supplied:

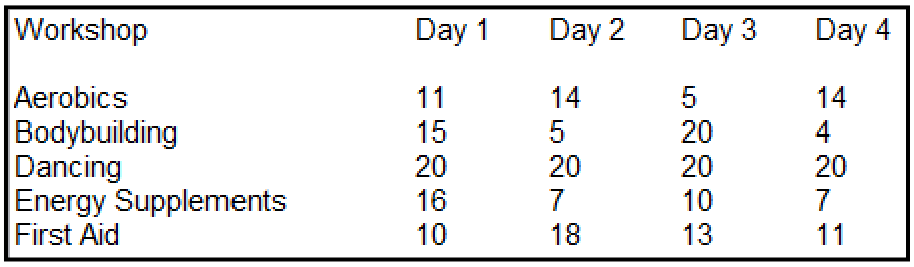


```
Bottles of water needed:
Day 1                        82
Day 2                        78
Day 3                        84
Day 4                        76

Total:                       320
Cases of bottled water needed: 14
```

Cancel a workshop:

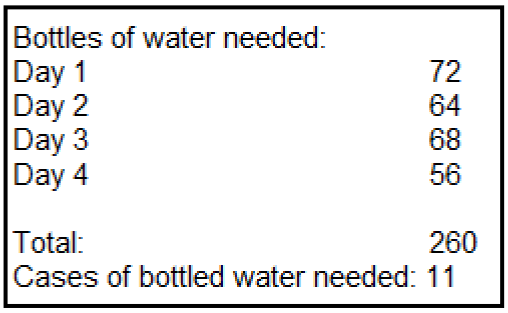
- Management would sometimes need to cancel a workshop due to too few bookings or the instructor not being available. The user must select a workshop to be cancelled. The selected workshop must be removed from the respective arrays and from the combo box.
- All the bookings for the selected workshop for the four days must be removed so that all output and calculated values exclude the cancelled workshop.

  Example of output if the Cardio workshop was removed (for original data supplied):



```
Workshop                  Day 1    Day 2    Day 3    Day 4

Aerobics                  11       14       5        14
Bodybuilding              15       5        20       4
Dancing                   20       20       20       20
Energy Supplements        16       7        10       7
First Aid                 10       18       13       11
```

Example of output of the number of bottles of water needed if the Cardio workshop was removed (for original data supplied):



```
Bottles of water needed:
Day 1                        72
Day 2                        64
Day 3                        68
Day 4                        56

Total:                       260
Cases of bottled water needed: 11
```

MARK ALLOCATION

REQUIREMENTS	MAXIMUM MARKS
Use appropriate processing and output components. Rename components using suitable descriptive names.	2
Apply good programming techniques and modular design.	2
Display content of arrays with headings.	8
Book for a workshop.	11
Calculate and display the number of bottles of water and cases of bottled water required.	10
Cancel a workshop and remove relevant data.	10

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

TOTAL SECTION C: 43
GRAND TOTAL: 150