

SA's Leading Past Year

Exam Paper Portal



You have Downloaded, yet Another Great
Resource to assist you with your Studies 😊

Thank You for Supporting SA Exam Papers

Your Leading Past Year Exam Paper Resource Portal

Visit us @ www.saexampapers.co.za



SA EXAM
PAPERS



basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

SENIOR CERTIFICATE EXAMINATIONS

INFORMATION TECHNOLOGY P1

2016

MARKS: 150

TIME: 3 hours

This question paper consists of 19 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 that 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 **DelphiDataENGJune2016.exe**.
- Java candidates must use the file **JavaDataENGJune2016.exe**.

Do the following:

- Double click on the password-protected executable file.
- Click on the 'Extract' button.
- Enter the following password: **Hosp9@%\$**

Once extracted, the following list of files will be available in the folder **DelphiDataENGJune2016/JavaDataENGJune2016**:

DELPHI FILES**Question1:**

Question1_P.dpr
Question1_P.res
Question1_U.dfm
Question1_U.pas

Question2:

PrescriptionItem.pas
Question2_P.dpr
Question2_P.res
Question2_U.dfm
Question2_U.pas

Question3:

Question3_P.dpr
Question3_P.res
Question3_U.dfm
Question3_U.pas
DataQ3.txt

JAVA (NETBEANS) FILES**Question1:**

Question1.form
Question1.java

Question2:

PrescriptionItem.java
Question2.form
Question2.java

Question3:

Question3.form
Question3.java
DataQ3.txt

SCENARIO

The Good Health Medical Centre provides highly trained practitioners and medical services to the public.

SECTION A**QUESTION 1: GENERAL PROGRAMMING SKILLS**

The medical centre has a number of different facilities that are available to the community.

Do the following:

- Compile and execute the program found in the **Question1** folder. The interface displays four different sections named Question1_1 to Question1_4. Currently the program has no functionality.
- Complete the code for each section of QUESTION 1, as described in QUESTION 1.1 to QUESTION 1.4 on the next page.

Example of graphical user interface (GUI):

Question1_1

Display the name here

Question1_2

Medical aid options

Option A
Option B
Option C
Option D

Number of dependants

Question1_2

Question1_3

Enter the account balance

Question1_3

Question1_4

Enter the number of patients

Doctors on duty

Number of patients

Doctor 1

Doctor 2

Doctor 3

Question1_4

Minutes per patient

15

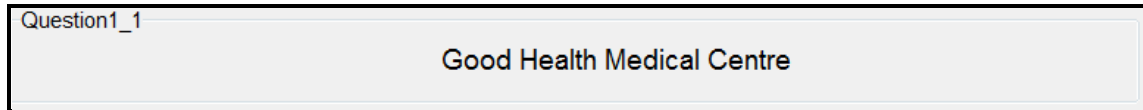
Starting time

☒ 8h00 ☐ 11h00 ☐ 13h00

Time at work

Sign-out time

- 1.1 Write code to set the font size of the text on the label **lblQuestion1_1** to 14 pt and display the text 'Good Health Medical Centre' on the label when the program is executed.



The screenshot shows a form with a label 'Question1_1' in the top left corner. Below the label is a text box containing the text 'Good Health Medical Centre'.

(2)

1.2 **Button [Question1_2]**

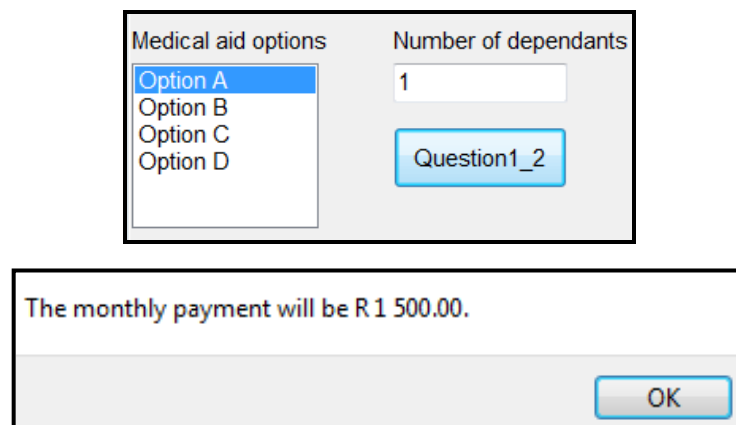
The community is encouraged to contribute towards a medical aid scheme. The scheme offers four options: Option A to Option D. Option A provides the minimum number of benefits and is the least expensive option with an amount of R1 000,00 per month as contribution for the main member of a family. This amount increases by 20% with each option (Option B, Option C and Option D) thereafter.

Family members of the main member are added as dependants to the account of the main member. An amount of 50% of the payment by the main member will be added to his/her account for each dependant.

The user must select an option from the list box and enter the number of dependants in the text box.

Write code to calculate the total monthly payment the member has to make towards the medical aid scheme. Display the monthly payment in a dialog box with a suitable message. If no option was selected, display a message instructing the user to select an option.

Example of output if Option A is selected and the number of dependants is one:



The screenshot shows two windows. The top window is titled 'Medical aid options' and contains a list box with four options: Option A, Option B, Option C, and Option D. Option A is selected. To the right of the list box is a text box labeled 'Number of dependants' containing the value '1'. Below the text box is a button labeled 'Question1_2'. The bottom window is a dialog box with the text 'The monthly payment will be R 1 500.00.' and an 'OK' button.

Example of output if Option B is selected and the number of dependants is three (on the next page):

Medical aid options

- Option A
- Option B
- Option C
- Option D

Number of dependants

3

Question1_2

The monthly payment will be R 3 000.00.

OK

(10)

1.3 Button [Question1_3]

The patient's account can be paid over a period of 12 months.

The first payment is calculated as 15% of the account balance. After the first payment has been deducted the balance must be paid in equal payments over the remainder of the months.

The user is required to enter the account balance in the text box.

Write code to calculate and display the payment number, the amount to be paid monthly and the decreasing balance in neat columns with suitable headings. The amounts must be formatted to currency with two decimal places.

Enter the account balance

4500

Question1_3

Payment	Monthly Payment	Balance
1	R 675.00	R 3 825.00
2	R 347.73	R 3 477.27
3	R 347.73	R 3 129.55
4	R 347.73	R 2 781.82
5	R 347.73	R 2 434.09
6	R 347.73	R 2 086.36
7	R 347.73	R 1 738.64
8	R 347.73	R 1 390.91
9	R 347.73	R 1 043.18
10	R 347.73	R 695.45
11	R 347.73	R 347.73
12	R 347.73	R 0.00

(10)

- 1.4 There are three doctors on duty every day: Doctor 1, Doctor 2 and Doctor 3. The total number of patients must be divided equally amongst the three doctors. If there are any remaining patients, the first patient will be allocated to the first doctor (Doctor 1) and the second patient to the second doctor (Doctor 2).

The algorithm below was compiled to calculate and display the number of patients each doctor will attend to during the day.

Algorithm:

```

1. number ← Enter the total number of patients
2. patients_per_doctor ← number integer division by 3
3. doctor1 ← patients_per_doctor
4. doctor2 ← patients_per_doctor
5. doctor3 ← patients_per_doctor
6. remainder ← number modulus 3
7. if remainder = 1 then
    doctor1 ← doctor1 + 1
    if remainder = 2 then
        doctor2 ← doctor2 + 1
8. display doctor1, doctor2 and doctor3

```

- 1.4.1 (a) Convert the instructions provided in the algorithm above into programming code to calculate and display the number of patients per doctor. (10)
- (b) The incorrect number of patients is allocated to the doctors due to a logical error in one of the IF statements in the algorithm. Identify the incorrect IF statement and correct it in your code so that the correct number of patients is assigned to each doctor.

Example of correct output for a total of five patients:

Enter the number of patients		<input type="text" value="5"/>
Doctors on duty	Number of patients	
Doctor 1	<input type="text" value="2"/>	
Doctor 2	<input type="text" value="2"/>	
Doctor 3	<input type="text" value="1"/>	
<input type="button" value="Question1_4"/>		

Example of correct output for a total of ten patients (on the next page):

Enter the number of patients		<input type="text" value="10"/>
Doctors on duty	Number of patients	
Doctor 1	<input type="text" value="4"/>	
Doctor 2	<input type="text" value="3"/>	
Doctor 3	<input type="text" value="3"/>	
<input type="button" value="Question1_4"/>		

(3)

- 1.4.2 The doctors on duty have to wait until all the patients on the list have been attended to before they can sign out as a team. They have to provide the number of hours and minutes they have worked using the number of hours and minutes that the first doctor has worked. They are only allowed to sign out on the hour. For example, if they worked for 3 hours and 30 minutes, starting at 08:00, they will all sign out at 12:00.

Add code to the QUESTION1_4 button to do the following:

Obtain the duration of one consultation in minutes. The default time is 15 minutes, but the user is allowed to enter any other value. Obtain the time of day when consulting starts. Calculate and display the time the doctors have worked and the time of day when they will sign out.

Example of output if the total number of patients is five, the duration of one consultation is 15 minutes and consulting starts at 08:00:

Enter the number of patients		<input type="text" value="5"/>
Doctors on duty	Number of patients	
Doctor 1	<input type="text" value="2"/>	
Doctor 2	<input type="text" value="2"/>	
Doctor 3	<input type="text" value="1"/>	
<input type="button" value="Question1_4"/>		
Minutes per patient		<input type="text" value="15"/>
Starting time		
<input checked="" type="radio"/> 8h00 <input type="radio"/> 11h00 <input type="radio"/> 13h00		
Time at work	<input type="text" value="0 hour(s) 30 minutes"/>	
Sign-out time	<input type="text" value="9h00"/>	

Example of output if the total number of patients is ten, the duration of one consultation is 20 minutes and consulting starts at 13:00:

Enter the number of patients	<input type="text" value="10"/>
Doctors on duty	Number of patients
Doctor 1	<input type="text" value="4"/>
Doctor 2	<input type="text" value="3"/>
Doctor 3	<input type="text" value="3"/>
<input type="button" value="Question1_4"/>	
Minutes per patient	<input type="text" value="20"/>
Starting time	
<input type="radio"/> 8h00 <input type="radio"/> 11h00 <input checked="" type="radio"/> 13h00	
Time at work	<input type="text" value="1 hour(s) 20 minutes"/>
Sign-out time	<input type="text" value="15h00"/>

(10)

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

TOTAL SECTION A: 45

SECTION B**QUESTION 2: OBJECT-ORIENTATED PROGRAMMING**

The Good Health Pharmacy allows pharmacists to enter the prescription the patient has received for tablets electronically. The labels for the tablet containers and a patient account will then have to be created. The total cost of all prescriptions required must then be calculated.

Do the following:

DELPHI PROGRAMMERS	JAVA PROGRAMMERS
<ul style="list-style-type: none"> Open the incomplete program found in the Question2 folder. Open the incomplete object class PrescriptionItem.pas. Enter your examination number as a comment in the first line of both files Question2_U.pas and PrescriptionItem.pas. 	<ul style="list-style-type: none"> Open the incomplete program found in the Question2 folder. Open the incomplete object class PrescriptionItem.java. Enter your examination number as a comment in the first line of both classes Question2.java and PrescriptionItem.java.

- Compile and execute the program. The program currently has no functionality.

Example of the interface:

NOTE: The **pnlGeneric** panel is set not to be visible when the program is executed.

- Complete the code for this program as specified in QUESTION 2.1 and QUESTION 2.2 below.

2.1 The given incomplete object class (**TPrescriptionItem/PrescriptionItem**) contains the declaration of four attributes that describes the **PrescriptionItem** object.

The given attributes of the **PrescriptionItem** object are as follows:

NAMES OF ATTRIBUTES		DESCRIPTION
Delphi	Java	
fCode	code	A unique six-character code allocated to each prescribed tablet
fDosage	dosage	A string consisting of four characters to indicate the number of tablets to take (See explanation in note below.)
fDays	days	The prescribed number of days the tablets must be taken
fPrice	price	The price of a single tablet

Complete the code in the given **PrescriptionItem** class (**TPrescriptionItem/PrescriptionItem**) as described in QUESTIONS 2.1.1 to 2.1.6 below.

NOTE: The dosage attribute consists of four characters in the format XX#X, where the # is used as a symbol to separate information.

Example 1: D1#B

Example 2: H2#8

First character	Explanation
D	Tablets must be taken once daily. Second character: Number of tablets to take at a time Fourth character: B: After breakfast S: After supper Example: D1#B means: Take one tablet daily after breakfast.
H	Tablets must be taken hourly. Second character: Number of tablets to take at a time Fourth character: Refers to the hourly period that the tablet must be taken. The hourly period can either be 2, 3, 4, 6 or 8. Example: H2#8 means: Take two tablets every 8 hours.

2.1.1 Write code for a **constructor** method to receive the tablet code, dosage code, number of days and the price per tablet as parameter values. Assign these values to the relevant attributes. (4)

2.1.2 Write a method called **calcTabletsPerDay** that will use the dosage code to determine and return the number of tablets that is prescribed per day. A day is regarded as a period of 24 hours.

Example 1: If the dosage code is D1#S, it means that one tablet must be taken daily after supper.

Example 2: If the dosage code is H2#4, it means that two tablets must be taken every four hours, which adds up to a total of 12 tablets per day. (8)

2.1.3 Write a method called **compileFrequencyOfUseMessage** that will use the dosage code to compile and return a message indicating when the tablets should be taken.

Example 1: If the dosage code is D1#S, the message to return is:
Take 1 tablet(s) daily after supper.

Example 2: If the dosage code is H4#8, the message to return is:
Take 4 tablet(s) every 8 hours. (8)

2.1.4 Write a method called **calcTotalTablets** to calculate and return the total number of tablets for this prescription item. (3)

2.1.5 Write **accessor** methods for the tablet code and price attributes. (2)

2.1.6 Write a **compileLabel** method to return the necessary information to be printed on the label of the tablet container in the format shown below.

Tablet code: <code>

<Frequency of use message> for <number of days>
days

(<Total number of tablets included> tablets)

Example of output if the tablet code is HPCMCN, the dosage code is H2#8 and the number of days is four:

Tablet code: HPCMCN

Take 2 tablet(s) every 8 hours for 4 days.

(24 tablets)

(5)

2.2 Two parallel arrays are provided in the incomplete class **Question2_U.pas/Question2.java**.

The array called **arrCodes** contains the codes of all available tablets. The array called **arrPrices** contains the prices per single tablet for each corresponding code.

Example of data in the supplied arrays:

The first four codes in the **arrCodes** array:

CBLTAD, HPCMCN, TRVKMG, BGJKPT

The first four prices in the **arrPrices** array:

17.89, 14.60, 23.50, 22.75

The data for the first tablet can be interpreted as follows:

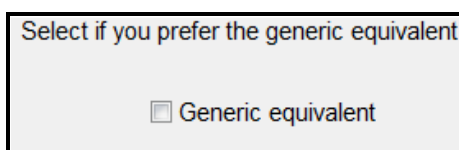
- The code of the tablet is **CBLTAD**.
- The price of a single tablet is **17.89**.

Do the following to complete the code for each button in the main form unit (Delphi)/GUI class (Java):

NOTE: The object **objPrescriptionItem** has been declared in the program.

2.2.1 Code for the combo box

The user is required to select the code of the tablet from the relevant combo box. If the code selected ends with the letter 'G', it means that a generic equivalent for the tablet is available. The panel **pnlGeneric** must be displayed, which allows the user to choose the generic equivalent of the tablet.



The image shows a rectangular panel with a light gray background and a black border. At the top, the text "Select if you prefer the generic equivalent" is displayed in a blue font. Below this text, there is a checkbox with a small square icon to its left, followed by the text "Generic equivalent" in a blue font.

A generic tablet is defined as a tablet that is used to treat the same condition as the original tablet but is usually sold at a lower price.

In this program the cost of the generic equivalent of a tablet is 40% less than the cost of the original tablet. The **arrPrices** array contains the prices of the original tablets.

The **pnlGeneric** panel must not be visible if there is no generic equivalent available for the tablet.

HINT: The indices of the codes in the **arrCodes** array are the same as those of the codes in the combo box.

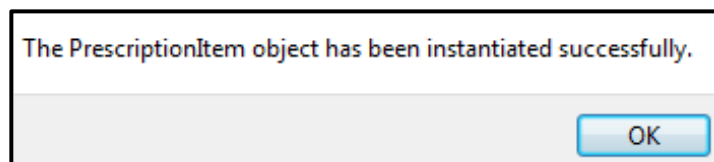
(9)

2.2.2 Button [2.2.2 – Instantiate Prescription Item]

Use the information supplied by the user and instantiate a new **PrescriptionItem** object. The tablet code, dosage code, number of days and the price per tablet must be used as arguments to instantiate the object.

If the **pnlGeneric** panel appears and the generic check box is selected, the price of the tablet must be reduced by 40% before the object is instantiated.

Display a message in a dialog box to indicate that the object was instantiated successfully.



(8)

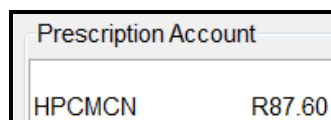
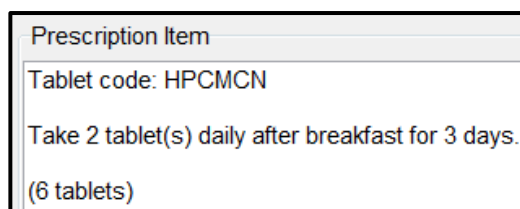
2.2.3 Button [2.2.3 – Display]

Use the **compileLabel** method to display the label information in the output area on the **Prescription Item** panel.

Also display the tablet code and the cost of this prescription item in the output area of the **Prescription Account** panel. The cost must be formatted to currency with two decimal places.

NOTE: The total cost of all prescription items must be calculated to be displayed when the **2.2.4 – Total of all Prescription Items** button is clicked.

Example of output for the prescription item and account details if the tablet code is HPCMCN, the dosage code is D2#B and the number of days entered is 3:



(8)

2.2.4 Button [2.2.4 – Total of all Prescription Items]

Display the total amount owed for this specific prescription, formatted to currency with two decimal places.

Example of output for the **Prescription Account** output area after all prescription items have been entered:

Prescription Account	
HPCMCN	R52.56
PLOMNG	R575.06
BGHYTR	R211.41
Total:	R839.03

(2)

2.2.5 Button [2.2.5 – Clear All]

Clear the output area for the prescription item and the prescription account. Also clear all the input components and set the relevant variables to 0.

(3)

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

TOTAL SECTION B: 60

SECTION C**QUESTION 3: PROBLEM-SOLVING PROGRAMMING****SCENARIO**

There are three sections in the wing for female patients at the Good Health Medical Centre. Section 1 is for emergency patients, Section 2 for children and Section 3 for adult patients. Each section has five wards and each ward has ten beds.

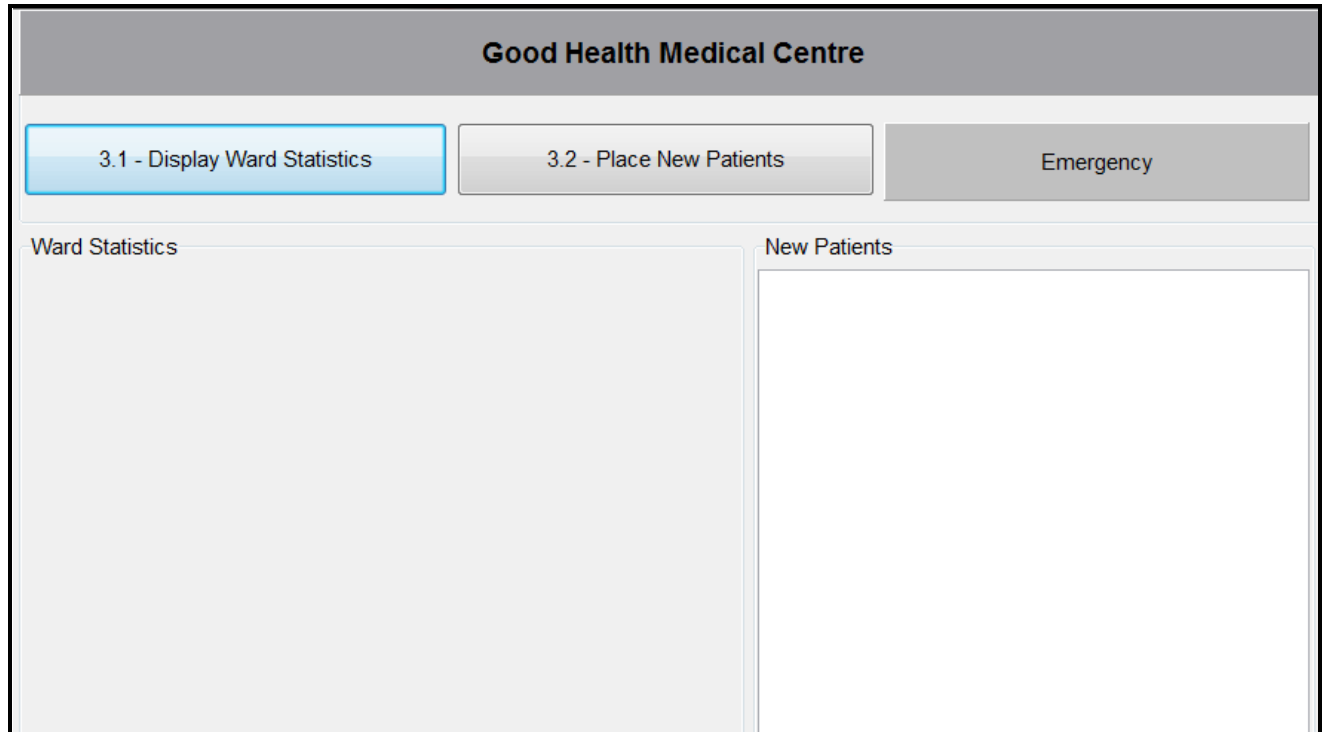
Do the following:

- Compile and execute the program found in the **Question3** folder. Currently the program has no functionality.
- Complete the code for each question as described in QUESTION 3.1 and QUESTION 3.2 below.

Supplied GUI:

The supplied GUI contains components for events that have to take place.

An example of the GUI is given below.



Use the program requirements in the questions that follow to decide on a suitable output component to be placed in the output area labelled **Ward Statistics** on the GUI.

Supplied data:**NOTE:**

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

You are provided with a two-dimensional array called **arrWardStats**, which contains the number of beds currently occupied by patients in each of the five wards.

The values that appear in the **arrWardStats** array are the following:

NOTE: The labels are not part of the content of the array.

	Ward 1	Ward 2	Ward 3	Ward 4	Ward 5
Emergency	10	9	10	8	9
Children	9	10	10	9	10
Adults	9	3	8	2	1

3.1 Button [3.1 – Display Ward Statistics]

The program must display the different sections, ward numbers and the number of patients in each ward. The information must be neatly displayed in columns with suitable headings and labels.

Example of output:

Ward Statistics					
	Ward 1	Ward 2	Ward 3	Ward 4	Ward 5
Emergency	10	9	10	8	9
Children	9	10	10	9	10
Adults	9	3	8	2	1

(8)

3.2 Button [3.2 – Place New Patients]

A text file called **DataQ3.txt** contains information on new patients that need to be admitted. Each line of text in the file contains the full name and date of birth (YYYY-MM-DD) of the patient separated by a hash character. If the patient must be admitted to the emergency ward, a semicolon, followed by the character 'E', has been added to the line of text.

Example of the first three lines of text in the file:

```
Debra Jenkins#2005-10-12
Geraldine Mathews#1992-01-09
Nosipho Mbele#2010-09-23;E
```

Write code to do the following:

Check whether the text file **DataQ3.txt** exists.

If the text file does not exist, display a suitable message and close the program.

If the text file exists, do the following for each new patient:

- Determine the section: Emergency, Children or Adults. A patient is considered to be a child if he/she is younger than eighteen years.
- Find a ward with an available bed in the section. The search process must start at Ward 1. If there are no available beds in Ward 1, then Ward 2 must be checked, and so forth. Each ward has a maximum of ten beds.
- Compile a placement code indicating the section and number of the ward where the patient will be placed.

Example:

A3 means that the patient will be placed in the section for adults in Ward 3.

If all wards in the specific section are full, the placement code for the patient must be the words 'Waiting list'.

If an emergency patient is placed on a waiting list, the colour of the 'Emergency' panel must be changed to red.

- Display the name, age and placement code in the output area for new patients.
- If a patient is placed in a ward, the values in the **arrWardStats** array must be updated accordingly.

Display the updated **arrWardStats** array after processing all the patients in the text file.

Example of **New Patients** output area (on the last page):

New Patients		
Name of patient	Age	Ward
Debra Jenkins	11	C1
Geraldine Mathews	24	A1
Nosipho Mbele	6	E2
Samantha Myers	9	C4
Brenda Hofmeyer	32	A2
Annamarie James	15	Waiting List
Elizabeth Morake	17	E4
Lee-Ann Johnson	14	E4
Ashley Brown	8	E5
Diane Malinga	16	Waiting List

Example of **Ward Statistics** output area after new patient details have been processed:

Ward Statistics					
	Ward 1	Ward 2	Ward 3	Ward 4	Ward 5
Emergency	10	10	10	10	10
Children	10	10	10	10	10
Adults	10	4	8	2	1

MARK ALLOCATION FOR QUESTION 3.2	
Read and obtain information from the text file.	10
Determine the section to which the patient will be admitted.	8
Determine the ward that is available in the specific section.	7
Construct the placement code or the words 'Waiting list'.	6
Update the arrWardStats array.	2
Display the name, age and placement code or words of the new patient.	3
Display the updated arrWardStats array.	1
TOTAL	37

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

TOTAL SECTION C: 45
GRAND TOTAL: 150