basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

NATIONAL SENIOR CERTIFICATE

GRADE 12

INFORMATION TECHNOLOGY P1

NOVEMBER 2013 (1)

MARKS: 120

TIME: 3 hours

This question paper consists of 11 pages and 2 annexures.

INSTRUCTIONS AND INFORMATION

1. 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.

2. You require the files listed below in order to answer the questions. The invigilator/teacher will tell you where to find them.

   QUESTION 1:
   Question1.java
   Question1DB.mdb
   tblCompetitors.txt
   tblResults.txt
   TestQuestion1.java

   QUESTION 2:
   TestQuestion2.java

   QUESTION 3:
   DataQ3.txt
   TestQuestion3

   If you received the files above on an external medium (CD, stiffy or flash drive) write your examination number on the label.

3. Type your examination number as a comment in the first line of each program file that contains your programming code.

4. Your program should always be coded to answer the question in such a way that it will run with different sets of input data.

5. Read ALL the questions carefully. Do not do more than the questions require.

6. Read the entire question before you answer any subquestions.

7. Save your work at regular intervals as a precaution against power failures.

8. During the examination, you may use the manuals originally supplied with the hardware and software. You may also use the HELP functions of the software. Java candidates may use the Java API files. You may NOT use any other resource material.

9. At the end of this examination session, you must hand in the external medium with all your work saved on it OR you must make sure that all your work has been saved on the network as explained to you by the invigilator/teacher.

10. Ensure that all files saved on the external medium or network can be read.

11. If required, make printouts of the programming code for all of the questions you have done.

12. All printing of the questions that you have done will take place within an hour of the completion of this examination.

SCENARIO

The local community organised a talent competition to raise funds for their community centre.

QUESTION 1: PROGRAMMING AND DATABASE

The talent competition was held over two rounds. A panel of three judges each evaluated the items competitors performed and awarded a score out of 10 for each item. The scores were combined to provide a total panel score out of 30. Members of the audience could SMS a score out of 10 for each performance. An average SMS score out of 10 for each performance was calculated as part of the evaluation process. The type of items varied from dance to singing to impersonations, amongst others.

A Microsoft Office Access database named **Question1DB.mdb**, two text files (**tblCompetitors.txt** and **tblResults.txt**) and an incomplete program are given in the folder named **QUESTION 1**.

The design of the tables in the **Question1DB** database and sample data from each table are given in **ANNEXURE A**.

Do the following:

- Make a backup copy of the **Question1DB** database **BEFORE** you start answering the questions. You will need a copy of the original database to be able to test your program thoroughly.
- Rename the folder for QUESTION 1 by adding your examination number to the name of the folder.
- Open the incomplete program for QUESTION 1.
- Enter your examination number as a comment in the first line of the program file.
- Compile and execute the program. The interface displays eight menu options: Option A to Option G, and a Quit option.

  NOTE:

  - An error message will be displayed if any of options A–G are selected because of the incomplete SQL statements.
  - If you experience any problems using the database or connecting to the database, refer to **ANNEXURE B** for troubleshooting hints.
  - If you still experience database problems, you must nevertheless do the SQL code and submit it for marking. **Marks will only be awarded for the programming code that contains the SQL statements.**

- Complete the code for each menu option by formulating an appropriate SQL statement to display the respective query results as described in QUESTIONS 1.1 to 1.7 below.

  NOTE: The code for some input statements and the code to execute the SQL statements and display the results of the queries have already been written as part of the given code.

1.1 **Menu Option A**

Display all the information in the **tblCompetitors** table. Display all the fields. The data must be sorted according to the date the competitor registered to take part in the competition – from the earliest to the latest date.

Example of the output:

CompetitorNo	Name	TypeOfItem	RegistrationDate	Local	AmountPaid
2	Dan	Singing	2013-05-12	False	200.00
7	Russell	Magic act	2013-05-17	True	200.00
13	Nancy	Instrument	2013-05-30	True	100.00
3	Lulu	Impersonation	2013-06-13	True	0.00
1	Holly	Dance	2013-07-01	False	0.00
12	Edwina	Singing	2013-07-15	False	300.00
14	Jason	Magic act	2013-07-31	False	400.00
6	Anita	Comedy	2013-07-31	False	0.00
8	Harry	Singing	2013-08-01	False	200.00
9	Rory	Impersonation	2013-08-12	False	0.00
5	Robby	Singing	2013-08-22	True	400.00
4	Audley	Instrumental	2013-08-31	False	100.00
11	Chelsee	Dance	2013-09-01	False	150.00
10	Alex	Comedy	2013-09-30	True	0.00

(3)

1.2 **Menu Option B**

Display the names and registration dates of competitors who registered during the month of August and are NOT members of the local community.

Example of the output:

Name	RegistrationDate
Audley	2013-08-31
Harry	2013-08-01
Rory	2013-08-12

(5)

1.3 **Menu Option C**

Each competitor had to pay a registration fee of R400,00. A query has been received regarding the amounts owed by competitors who performed a 'Magic act' as their type of item. Calculate and display the number of the competitor, the amount paid so far and the amount outstanding of all the competitors who performed a magic act. Use **Amount Outstanding** as the heading for the calculated field.

Example of the output:

CompetitorNo	AmountPaid	Amount Outstanding
7	200.00	200.00
14	400.00	0.00

(5)

1.4 **Menu Option D**

Clothing items were left behind in one of the dressing rooms that was NOT occupied by the competitors who performed dance items. The name of the person these items possibly belongs to either starts with the letter 'H' or ends with the letter 'y'. Display the names and types of items performed of all the possible competitors to whom the clothing items could belong.

NOTE: Each name must only appear on the list ONCE.

Example of the output:

Name	TypeOfItem
Audley	Instrumental
Robby	Singing
Harry	Singing
Rory	Impersonation
Nancy	Instrumental

(7)

1.5 **Menu Option E**

Calculate and display the final score for each of the competitors in a calculated field named **FinalScore**.

The final score for each of the competitors (out of 40) must be calculated using the average score for both rounds from the judging panel (out of 30) (**PanelScore**) and the average score for both rounds from the audience (out of 10) (**SMSScore**) added together. Display the final score correct to ONE decimal place. Display the **CompetitorNo** and **FinalScore** of all the competitors.

Example of the output (on the next page):

CompetitorNo	FinalScore
1	28.6
2	26.4
3	23.4
4	23.6
5	32.0
6	16.3
7	25.0
8	38.7
9	26.5
10	20.3
11	34.5
12	36.4
13	25.4
14	24.3

(5)

1.6 **Menu Option F**

Compare the scores of the judging panel with that of the audience. Convert the score of the judging panel to a score out of 10 and compare it with the score of the audience, which is an average out of 10. Display the number of the item, the number of the competitor, the judging panel's score out of 10 and the **SMSScore** of only those items where the audiences' score is higher than the judges' score.

Example of the output:

ItemNo	CompetitorNo	Panel Score	SMSScore
20	6	2.7	5.1
26	12	8.3	8.9
5	5	7.7	8.9
6	6	4.0	7.4
8	8	9.3	9.9
14	14	5.7	6.7

(4)

1.7 **Menu Option G**

Allow the user to update the account of a competitor by entering the name of a competitor and the amount to be paid. Update the **AmountPaid** field in the **tblCompetitor** table. The total amount to be paid is R400,00 and the payments must only be accepted if the full payment has not been made yet.

Once the records have been updated successfully, a suitable message will be displayed. The code for this message is supplied.

Execute Option A to see whether the relevant **AmountPaid** field has been updated. (5)

- Enter your examination number as a comment in the first line of the file containing the SQL statements.
- Save your program.
- A printout for the code will be required.

[34]

QUESTION 2: OBJECT-ORIENTED PROGRAMMING

A program is required to provide information about ticket sales for the talent competition. Tickets become available three months before the first event. The sales staff sell tickets and earn commission based on the number of tickets sold. The program has to determine the number of tickets sold as well as the commission each salesperson has earned.

The files required for this question can be found in the folder named **QUESTION 2**. It contains an incomplete program named **TestQuestion2**.

Do the following:

- Rename the folder for **QUESTION 2** by adding your examination number to the name of the folder.
- Open the incomplete **TestQuestion2.java** test class file.
- Add your examination number as a comment in the first line of the **TestQuestion2.java** test class.
- Compile and execute the program. The program displays three menu options: Option A, Option B, and a Quit option.

2.1 Create an object class named **Employee** and save the file in the **QUESTION 2** - **XXXX** folder (XXXX represents your examination number).

Add your examination number as a comment in the first line of the **Employee** class. The class must contain the following private fields:

Name of the field	Description
name	Name of the employee
s_month1	The number of tickets the employee sold during the first month of sales
s_month2	The number of tickets the employee sold during the second month of sales
s_month3	The number of tickets the employee sold during the third month of sales

Ensure that you choose appropriate data types for these fields.

Do the following to complete the code in the object class named **Employee**:

2.1.1 Create a parameterised constructor that will pass values for the private instance fields of the class. These parameters should be used to initialise the instance fields of the class. (5)

2.1.2 Write a **toString** method to return information about an employee in one string, formatted as follows:

Name of employee<space>Tickets sold month 1<tab>Tickets sold month 2<tab>Tickets sold month 3

Example of a return string for an employee:

Vuyo 40 20 34 (4)

2.1.3 Write code for an accessor method named **getName** for the name of the object. (2)

2.1.4 Write code for a method named **calcTotal** to calculate the total number of tickets the employee sold. (3)

2.1.5 Write code for a method named **calcCommission** that will receive the selling price of a ticket and calculate and return the commission the employee has earned based on the total number of tickets he/she sold during the first three months of sales.

Commission on ticket sales is calculated as follows:

Number of tickets sold	Percentage commission
Less than 30 tickets	No commission
30 up to 100 tickets	10% of the sales amount
More than 100 tickets	15% of the sales amount

Examples:
(Assume the price per ticket is R100,00.)

Tickets sold	Sales amount	Commission
25	R2 500,00	R0,00
75	R7 500,00	R750,00
120	R12 000,00	R1 800,00

(10)

2.2 Do the following to complete the code in the test class.

NOTE: The code for QUESTION 2.2.1 must be executed before the menu is displayed when the program is executed.

2.2.1 Declare an array of Employee objects. Assume that there could be up to 20 employees who are selling tickets.

Write code to enter the information of an unknown number of employees from the keyboard and assign the Employee objects to the array according to the following steps:

- Allow the user to enter the name of an employee. Use 'XXX' to end input.
- Allow the user to enter THREE values indicating the number of tickets the employee sold during month 1, 2 and 3.

- Create an instance of an employee object using the user input values. Assign the object to the array.

- Repeat the steps listed above for a number of employees and keep track of the number of employees entered. (11)

2.2.2 Write code to complete **Menu Option A** to display the information of all the Employee objects using the **toString** method.

Example of the output:

```
List of employees
==============
Vuyo      40   20   34
Russell   35   115   50
```
(4)

2.2.3 Write code to complete **Menu Option B** to do the following:

Allow the user to enter the price of the ticket. Thereafter, display a list of the names of the employees, the total number of tickets each employee sold and the commission they earned.

Determine and display the name of the employee who sold the most tickets.

Example of the output:

```
List of Employees
==============
Name              Tickets sold      Commission
Vuyo                    94          R   940.00
Russell                 200         R 3000.00
.....
Best salesperson: Russell
```
(12)

- Make sure that your examination number is entered as a comment in the first line of the class as well as the test class.
- Save all the files.
- A printout of the code will be required. Print the object class (Java) and the test class. **[51]**

QUESTION 3: PROBLEM-SOLVING PROGRAMMING

A local university wants to award bursaries to some of the best competitors. Competitors who performed items during the competition were requested to supply their names, the type of item they performed and their final score as a percentage to a recruitment agency. The recruitment agency will process the information using a set of prescribed criteria and supply the local university with the names of the competitors who met the criteria.

The folder for QUESTION 3 contains a text file named **DataQ3.txt** and an incomplete program named **TestQuestion3**.

Each line of text in the text file (**DataQ3.txt**) contains information about the performance of the competitor in the following format:

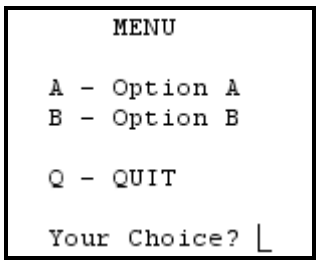
<name of the competitor>,<the type of item performed>:<final score at the competition as a percentage>

Example of the data for the first three competitors in the text file named **DataQ3.txt**:

```
Holly Thompson,Dance:77
Dan Ntumba,Singing:89
Lulu Franklin,Impersonation:71
```

Do the following:

- Rename the folder for **QUESTION 3** by adding your examination number at the end of the name of the folder.
- Open the incomplete program.
- Enter your examination number as a comment in the first line of the program.
.
- Execute the program. A menu with the following options will be displayed:



- Complete the code for each menu option as follows:

NOTE: Run the menu options in sequence when you test the program, that is first Option A and then Option B.

3.1 Competitors who performed dance, comedy or impersonation items and who have a final score of 70 or more for their performance are the successful candidates who will be invited for an interview at the university. Write code to be executed before the menu is displayed to select the names of the successful candidates from the given text file and place these names in an array called 'Interviewees'. (14)

3.2 Write a method to alphabetically sort the array ('Interviewees') containing the names of the selected candidates. (6)

3.3 **Menu Option A**

Write code for **Menu Option A** that will display an alphabetically sorted list with the names of the selected candidates. Use the method written in QUESTION 3.2 to sort the names alphabetically.

Example of the output:

```
Sorted List of Candidates
=========================
Alex Jay
Holly Thompson
Lulu Franklin
Rory Duma
```

(4)

3.4 **Menu Option B**

Temporary student numbers must be generated for the candidates who were selected to be interviewed. Write code for **Menu Option B** to do the following:

- Use the method written in QUESTION 3.2 to sort the array of selected candidates.
- Generate a student number for each candidate as follows:
  - Take the first three consonants from the name (first name and surname) of the learner. The student number may contain only capital letters.
    NOTE: All the letters from the alphabet except the vowels (A, E, I, O, U) are consonants.
  - Generate a three-digit random number.
  - Join the three consonants and the randomly generated number in a string to form a student number consisting of SIX characters.
- Display the names of the selected candidates and their temporary student numbers. Display a suitable heading and subheadings.

Example of the output:

```
Name                    Student Number
=====================================
Alex Jay                LXJ515
Holly Thompson          HLL312
Lulu Franklin           LLF693
Rory Duma               RRY273
```

The student numbers will be different every time the program is executed due to the numbers being randomly generated. (11)

- Make sure your examination number is entered as a comment in the first line of the program.
- Save all the files.
- A printout for the code will be required. **[35]**

TOTAL: 120

ANNEXURE A: DATABASE STRUCTURE AND SAMPLE DATA (Page 1 of 2)

This annexure shows the database structure and sample data for the tables contained in the **Question1DB.mdb** database used in **QUESTION 1**.

tblCompetitors: This table contains the data of the 14 competitors who took part in the talent competition.

Field Name	Type	Size	Description
CompetitorNo	Number	Integer	A unique number assigned to each competitor
Name	Text	15	The first name of the competitor
TypeOfItem	Text	30	The type of item the competitor performed, e.g. dance or singing or impersonation, et cetera
RegistrationDate	Date/Time	Short Date	The date when the competitor registered to take part in the competition
Local	Yes/No		Whether the competitor is a member of the local community or not
AmountPaid	Currency		The amount that was paid as part of the registration fee

Example data:

CompetitorNo	Name	TypeOfItem	RegistrationDate	Local	AmountPaid
1	Holly	Dance	2013/07/01	FALSE	R 0.00
2	Dan	Singing	2013/05/12	FALSE	R 100.00
3	Lulu	Impersonation	2013/06/13	TRUE	R 0.00
4	Audley	Instrumental	2013/08/31	FALSE	R 100.00
5	Robby	Singing	2013/08/22	TRUE	R 400.00
6	Anita	Comedy	2013/07/31	TRUE	R 0.00
7	Russell	Magic act	2013/05/17	TRUE	R 200.00
8	Harry	Singing	2013/08/01	FALSE	R 200.00
9	Rory	Impersonation	2013/08/12	FALSE	R 0.00
10	Alex	Comedy	2013/09/30	TRUE	R 0.00
11	Chelsee	Dance	2013/09/01	FALSE	R 150.00
12	Edwina	Singing	2013/07/15	FALSE	R 300.00
13	Nancy	Instrumental	2013/05/30	TRUE	R 100.00
14	Jason	Magic act	2013/07/31	FALSE	R 400.00

ANNEXURE A (continued) **(Page 2 of 2)**

tblResults: This table contains data on the results of all the competitors over two rounds of the competition.

Field Name	Type	Size	Description
ItemNo	Number	Integer	A unique number assigned to each item that was performed
CompetitorNo	Number	Integer	A unique number to identify the competitor
Round	Number	Integer	A number indicating the round during which the item was performed
PanelScore	Number	Integer	The judging panel's total score out of 30
SMSScore	Number	Double	The average score from the audience via SMS scores out of 10

Example data:

ItemNo	CompetitorNo	Round	PanelScore	SMSScore
1	1	1	25	6.3
2	2	1	20	4.5
3	3	1	15	3
4	4	1	18	3.1
5	5	1	23	8.9
6	6	1	12	7.4
7	7	1	15	4.5
8	8	1	28	9.9
9	9	1	24	6.5
10	10	1	15	4.4
11	11	1	26	7.7
12	12	1	30	8.8
13	13	1	18	5.6
14	14	1	17	6.7
15	1	2	20	5.9
16	2	2	22	6.3
17	3	2	23	5.8
18	4	2	21	5.1
19	5	2	25	7
20	6	2	8	5.1
21	7	2	23	7.4
22	8	2	30	9.5
23	9	2	17	5.5
24	10	2	18	3.2
25	11	2	27	8.2
26	12	2	25	8.9
27	13	2	21	6.1
28	14	2	19	5.9

ANNEXURE B: JAVA – TROUBLESHOOTING DATABASE PROBLEMS

B.1 If you cannot use the given database:

- Create your own database with the name **Question1DB** that includes two tables, named **tblCompetitors** and **tblResults** in the same folder as your program for QUESTION 1.
- Import the two text files (**tblCompetitors.txt** and **tblResults.txt**) to use as data for the different tables.
- The first line in the text files contains the field names to be used.

B.2 If your program cannot establish connectivity with the database make sure that the database **Question1DB** is in the same folder as your program for QUESTION 1. If this is not the case, copy the database file into the same folder as your program.

B.3 If you cannot establish connectivity with the database with the given files, use the following source code to ensure database connectivity:

```
try
  {
     Class.forName ("sun.jdbc.odbc.JdbcOdbcDriver");
     String filename = "Question1DB.mdb";
     String database = "jdbc:odbc:Driver={Microsoft Access Driver (*.mdb)};DBQ=";
            database += filename.trim () + ";DriverID=22;READONLY=true}";
     Connection conn = DriverManager.getConnection (database, "", "");
  }
  catch (Exception e)
  {
     System.out.println ("Unable to connect to the database");
  }
```