**Name of the school**

**September 2019**

**Preliminary Examination**

**Grade 12**

**Information Technology P 1**

**2019**

**Marks: 150**

**Time: 3 hours**

**This paper consists of 21 pages.**

**INSTRUCTIONS AND INFORMATION**

1. This is a **three-hour** examination. 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. Answer ALL the questions.
3. The files you need to complete this question paper have been given to you on the disk/CD/DVD/Flash disk or on the disk space allocated to you.
4. You require the files listed below in order to answer the questions:

**QUESTION 1:**

Question1\_P.dpr

Question1\_P.dfm

Question1\_U.pas

Cansa.png

CDE.png

Choc.jpg

jam.png

Smile.png

symphonia.jpg

noimage.jpg

**QUESTION 2:**

Question2\_P.dpr

Question2\_P.dfm

Question2\_U.pas

NGO.mdb

NGOBackup.mdb

**QUESTION 3:**

Question3\_P.dpr

Question3\_P.dfm

Question3\_U.pas

clsQuestion3.pas

NGO.txt

**QUESTION 4:**

Question4\_P.dpr

Question4\_P.dfm

Question4\_U.pas

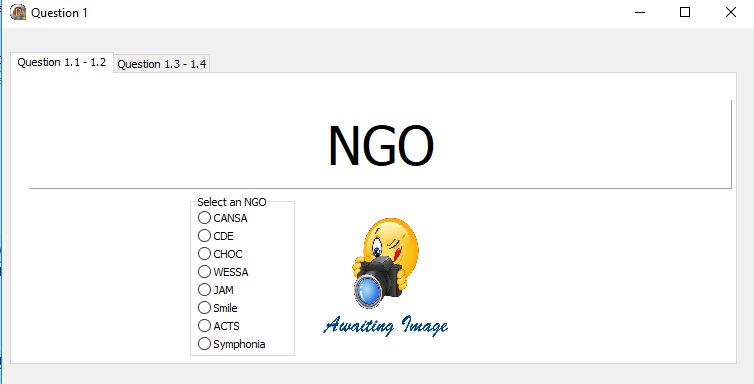
1. Save your work at regular intervals as a precaution against power failures.
2. Rename the folder **12ITPrelimSurnameName** with your own surname and name. E.g. 12ITPrelimSmithJohn
3. Type your name and surname as a comment in the first line of each program.
4. Make sure you answer the questions according 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, 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 or locate, sort and selection or filter must be developed from first principles. You may NOT use the built-in features of Delphi for any of these routines.
8. All data structures must be defined by you, the programmer, unless the data structures are supplied.
9. After the examination you will be given time to print all your units to hand in.

**QUESTION 1: GENERAL PROGRAMMING SKILLS**

Do the following:

* Open the incomplete **Question1\_P.dpr** program in the folder **Question1**.
* Enter your name and surname as a comment on the first line of the **Question1\_U.pas** file.
* Compile and execute the program. The program currently has no functionality. The user interface displays TWO tab sheets.
* Follow the instructions below to complete the code for EACH section of QUESTION 1, as described in QUESTION 1.1. to QUESTION 1.4

**Tab sheet [Question 1.1 – 1.2]**



* 1. **In the [FormActivate] event handler**

Write code to set the properties of the panel **pnlQ1\_1** as follows:

* + - Caption to “Non-Profit Organisations”
    - Font style to bold
    - Font colour to clTeal (5)

Example of output:



* 1. **In the [rgpNGOClick] event handler**
     + Image files have been provided for six of the NGOs (Non-profit organisations) listed in the RadioGroup. The names of these files correspond with the name of the organisations. Some pictures have a .jpg and others a .png file extension as in the table below:

|  |  |
| --- | --- |
| **Name of organisation** | **Name of file** |
| CANSA | Cansa.**png** |
| CDE | CDE.**png** |
| Choc | Choc.**jpg** |
| JAM | jam.**png** |
| Smile | Smile.**png** |
| Symphonia | symphonia.**jpg** |

* + - Write code to create the filename for the picture to be loaded according to the user’s selection of the organisation and the filename extensions provided above.
    - Load the picture file for the selected organisation into the imgNGO component.
    - If no image is available load the **noimage.jpg** file into the imgNGO component.

(12)

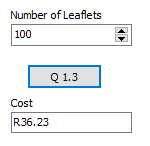
Examples of selected input and corresponding output:



**Tab sheet [Question 1.3 – 1.4]**

* 1. **Button [Q1.3]**
     + Retrieve the number of leaflets the user would like to order for the organisation.
     + 15 Leaflets are packed as a unit. Calculate the number of units the user will need to order. Ensure that the user will rather order too many leaflets rather than too few.
     + Each unit of 15 leaflets cost R4.50. Calculate the cost of the units ordered using the formula: Cost = units X R4.50.
     + Add 15% VAT to this cost.
     + Display the final amount owing in edtQ1\_3, in currency format. (10)

Example of output when 100 flyers are ordered:



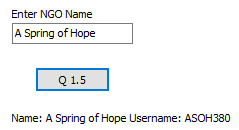
* 1. **Button [Q1.4]**

Each NGO needs to register on our South African government website and needs a username.

* + - Retrieve the name of the organisation.
    - Use this name to build a username for the organisation. The username is made up of:
      * The first letter of each word in the name of the organisation in capital letters.
      * Followed by three digits randomly generated.
    - Display the name of the organisation and the generated username in the lblOutput component. (13)

Example of the output when “A Spring of Hope” was entered:

NOTE: your output will be different due to the random numbers.



* Enter your Name and Surname as a comment in the first line of the program file **Question1\_U**.
* Save all the files ('File/Save All').
* Printouts of the code for the unit (**Question1\_U**) will be required.

**[40]**

**QUESTION 2: DATABASE PROGRAMMING**

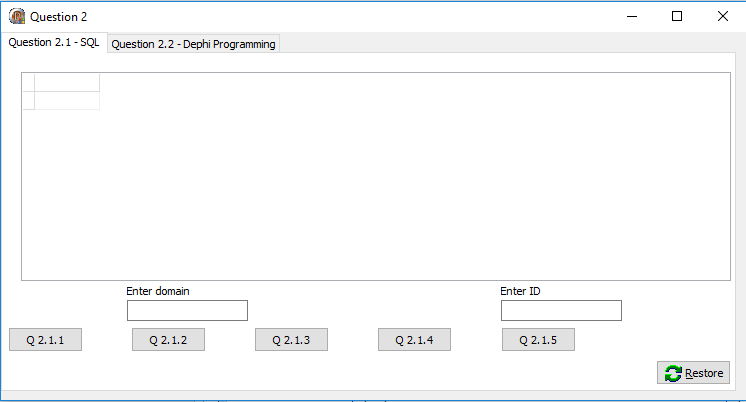
**SCENARIO:** A database NGO.mdb has been created containing a list of South African NGOs and the volunteers that give up their time for these NGOs.

The data pages attached at the end of the question paper (**Annexure A**) provide information on the design of the database and its contents.

Do the following:

* Open the incomplete project file **Question2\_P.dpr** in the folder **Question2**.
* Enter your name and surname as a comment on the first line of the **Question2\_U.pas** file.
* The program has no functionality currently.

The following interface is displayed:



* Follow the instructions to complete the code for each section as described in QUESTION 2.1 and QUESTION 2.2.
* Use SQL code to answer QUESTION 2.1 and Delphi code (DML - Database Management Language) to answer QUESTION 2.2.

**NOTE:**

* The Restore button is provided to restore the data contained in the database to the original content. If you need to test your code with the original data, you may click this button to restore the data.
* Do not change any of the provided code.
* Code is provided to link the GUI components to the database.
* TWO variables are declared as public variables, as described in the table on the next page:

|  |  |  |
| --- | --- | --- |
| **Variable** | **Data type** | **Description** |
| tblNGO | TADOTable | Refers to the table tblNGO |
| tblVolunteers | TADOTable | Refers to the table tblVolunteers |

* 1. **Tab sheet [Question 2.1 – SQL]**

In this section you may use ONLY SQL statements to answer QUESTION 2.1.1 to QUESTION 2.1.5

Code to execute the SQL statements and display the result of the queries is provided. Enter your SQL statement between the provided inverted commas ('') for the **sSQL1**, **sSQL2**, **sSQL3**, **sSQL4** and **sSQL5** variables.

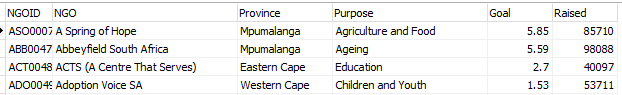
The following user interface is displayed:



* + 1. **Button [Q 2.1.1]**

Display all the records in the tblNGO table in alphabetical order according to the names (fieldname NGO) of the NGOs.

Example of output of the first four records:

 (3)

* + 1. **Button [Q 2.1.2]**

The domain name is the part of an e-mail address that follows the @ symbol. Use the domain name input by the user in edtDomain to perform a query. Display the FirstName, surname and e-mail (eMail) address of full-time volunteers in the tblVolunteers table whose e-mail addresses (eMail field) end with the domain name entered in edtDomain, by the user.

Example of output when gmail.com was entered:

 (5)

* + 1. **Button [Q 2.1.3]**

Display all the NGOs in Gauteng (province field) and the number of volunteers for each NGO in Gauteng. The number of volunteers must be displayed in a column with the heading **Number volunteers**.

An extract of the first few records in the output:

 (8)

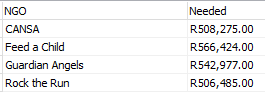
* + 1. **Button [Q 2.1.4]**

The following formula calculates the amount that each NGO still needs to raise to meet their goal amount:

Goal X 100 000 – Raised

Display the NGO names (NGO field) and amounts, in currency format, that need to be raised (using the formula) with the heading **Needed**, of those NGOs that still need to raise more than R 500 000 to reach their goal amount from the tblNGO table.

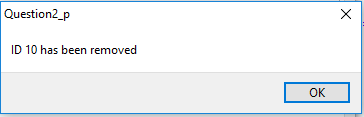
Example of the output:

 (5)

* + 1. **Button [Q 2.1.5]**

Delete the volunteer with ID number 10 from the table tblVolunteers.

Example of the output:

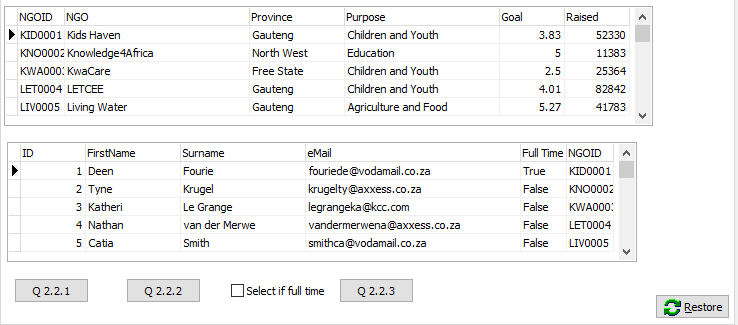
 (2)

* 1. **Tab sheet [Question 2.2 – Delphi code (DML)]**

In this section only Delphi programming code may be used to answer QUESTION 2.2.1 to QUESTION 2.2.3

NO marks will be awarded for SQL statements in QUESTION 2.2.

The following user interface for QUESTION 2.2 is displayed when the program is executed:



* + 1. **Button [Q 2.2.1]**

Calculate and display, in a dialog box, the total amount raised, by all NGOs whose purpose is Education.

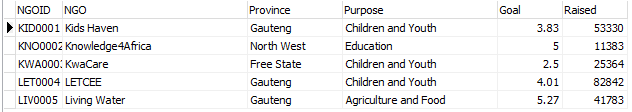
Example of output:

 (9)

* + 1. **Button [Q 2.2.2]**

Add R1 000 to the amount raised by the NGO selected (current active record) in tblNGO – the record in the example is already set to active.

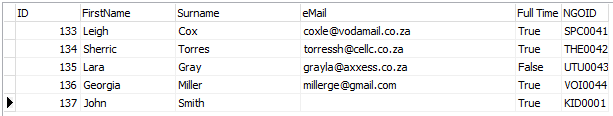
Example of output when Kids Haven was the active record:

 (4)

* + 1. **Button [Q 2.2.3]**

Insert a new volunteer, John (FirstName field) Smith (Surname field), into **tblVolunteers**. John Smith will volunteer at the selected NGO in **tblNGO**. Read the selected NGOs (active record) NGOID and assign it to the NGOID of John Smith in **tblVolunteers**. The user will indicate whether the volunteer is available full time (Full Time field) or not, by clicking on **chkFullTime** on the form.

Example of the output when **Kids Haven** was the active record in **tblNGO** and the user is available full time:

 (4)

* Enter your Name and Surname as a comment in the first line of the program file **Question2\_U**.
* Save all the files ('File/Save All').
* Printouts of the code for the unit (**Question2\_U**) will be required.

**[40]**

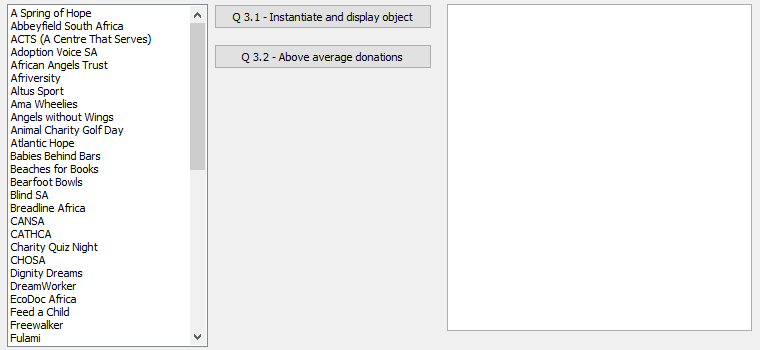
**QUESTION 3: OBJECT ORIENTED PROGRAMMING**

**SCENARIO:** NGOs rely on donations to support their cause. Software is needed to assist NGOs to keep track of their donations.

Do the following:

* Open the incomplete program **Question3\_P.dpr** in the folder **Question3**.
* Open the incomplete object class **clsNGO\_U.pas**.
* Enter your name and surname as a comment on the first line of the **Question3\_U.pas** file and the **clsNGO\_U.pas** file.
* Compile and execute the program. The program currently has no functionality.

The following interface is displayed:



Complete the code as specified in QUESTION 3.1 for the **clsNGO.pas** object class and in QUESTION 3.2. for the **Question3.pas** form class.

* 1. The incomplete object class (TNGO) provided contains the following:
* Declarations of three attributes that define an **objNGO** object.

|  |  |
| --- | --- |
| **Name of attribute** | **Description** |
| fName | Name of the NGO |
| fFunds | The total income from donations |
| fNumber | The number of donations |
| fCode | The registration code for the NGO |

* Declarations (Signatures) and skeleton code for the methods:
  + **Constructor Create**
  + **IncreaseNumber**
  + **CalculateAve**
  + **GetFunds**
  + **ToString**
    1. Complete the code for the given **Constructor Create** method to assign the parameter received to the name of the NGO attribute. Set the attributes for the total income from donations and the number of donations to zero. (3)
    2. Create a mutator method **SetFunds** to receive an amount for a donation as a parameter and add this amount to the total income from donations. (2)
    3. Complete the code for the given **IncreaseNumber** method to increase the number of donations by one. (1)
    4. Complete the code for the given **CalculateAve** method to calculate and return the average amount of money donated per donation. (1)
    5. Complete the code for the given accessor method **GetFunds** to return the total income from donations. (1)
    6. Complete the code for the given **ToString** method to return a string with attributes in the following format:

NGO: <Name of the NGO>

Number of donations: <Number of donations>

Total raised: <Total income from donations>

Average raised: <Average amount donated> (5)

* 1. The incomplete unit **Question3\_U** provided contains code for the object class to be accessible and the declaration of an object variable **objNGO**.

A text file called **NGO.txt** has been provided which contains the amounts donated, the NGO names and which company donated to them in the following format:

<Amount>#<Name of NGO>#<Company that donated>

Example of the first four lines of the text file:

2838.21#Kids Haven#Adcock Ingram

9101.31#Knowledge4Africa#Afrihost

8410.24#KwaCare#Airports Company South Africa (ACSA)

7799.56#LETCEE#Anglo American

The purpose of the program is to:

* Search for all the instances of the NGO selected from the ListBox, in the text file.
* Calculate the total income and number of donations for this NGO.
* Display a summary of the funds.
* Provide a list of companies that donated more than the average amount for this NGO.

Follow the instructions below to code the solution.

* + 1. **Button [3.2.1 –** **Instantiate and display object]**

Code has been provided to test if the file exists. The user will select the name of the NGO from the ListBox provided and the program must find all the donations for this NGO from the text file. Two arrays arrCompany and arrFunds are given to store donors and their donations to the selected NGO. A global counter (iCount) variable, has been declared to keep track of the number of items in the array.

Write code to do the following:

* Retrieve the name of the NGO the user selected from the **lstNGO** component.
* Instantiate the **objNGO** object.
* Search for the name of the NGO in the text file.
* When the NGO name is found:
  + Increase the counter iCount.
  + Extract the amount donated and the name of the donor company.
  + Store the name of the company in arrCompany and the amount donated in arrFunds.
  + Call the relevant methods to:
    - Add this amount to the funds attribute.
    - Increase the number of donations attribute by one.
* Display the summary of the donations for this NGO in the **redOutput** component using the **ToString** method.

Example of output if **A Spring of Hope** was selected:

 (19)

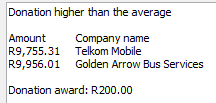
NOTE: Currency format may be omitted.

* + 1. **Button [3.2.2 – Above Average Donations]**

Code has been provided for the headings and the tab stops to display in the **redOutput** component.

* Use the **CalculateAve** method written in QUESTION 3.1.4 to display the amount and the company name, of those companies that donated more than the average amount raised for this NGO.
* For every full R10 000 raised for this NGO, a donation award of R100 will be received.
* Calculate and display the donation award for this NGO by using the **GetFunds** method.

Example of output if **Abbeyfield South Africa** was selected:



NOTE: Currency format may be omitted.

Example of output if **ACTS (A Centre That Serves)** was selected:



NOTE: Currency format may be omitted. (8)

* Enter your Name and Surname as a comment in the first line of the object class **clsNGO.pas** and the form class **Question3\_U**.
* Save all the files ('File/Save All').
* Printouts of the code for both the units (**Question3\_U.pas** and **clsNGO.pas**) will be required.

**[40]**

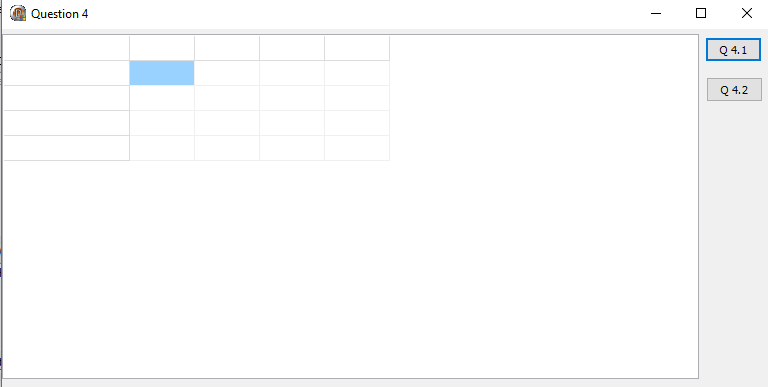
**QUESTION 4: PROBLEM-SOLVING PROGRAMMING**

**SCENARIO:** All non-profit organisations rely on donations to fund their cause. Data is stored in a two-dimensional array that needs to be processed and displayed.

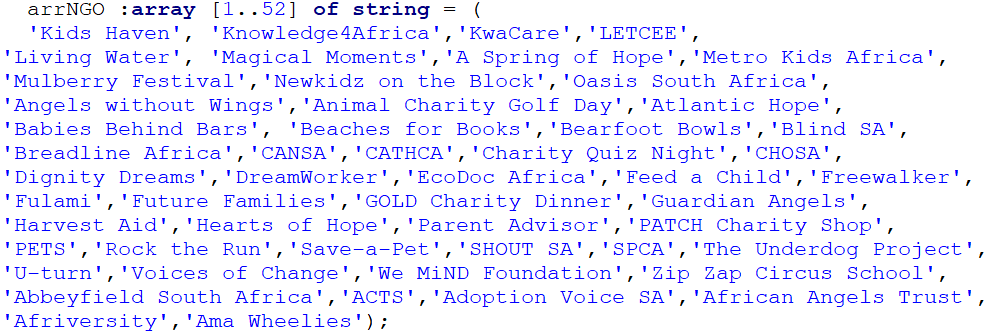
Do the following:

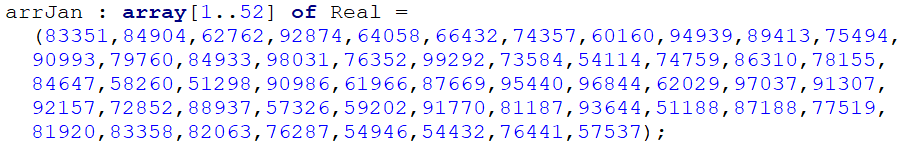
* Open the incomplete program**Question4\_P.dpr** in the folder **Question4**.
* Enter your name and surname as a comment on the first line of the **Question4\_U.pas** file.
* Compile and execute the program. The program has no functionality currently.

The following interface is displayed:

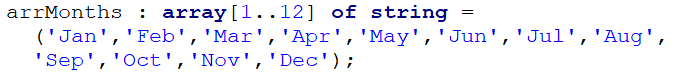


The program contains the declarations of four arrays called arrNGO, arrJan, arrMonths and arrDonations as well as an integer variable iMonth that keeps track of how many months of data has been added to the arrays.

* arrDonations: array [1..52,1..13] of real, keeps the donations for the possible 12 months and the total amount raised for the months entered.
* arrNGO contains the names of NGOs:   
  
* arrJan contains the funds raised (donations) in January:



* arrMonths contains the month names:



* iMonth has been given a value of 1 in Form OnActivate, which is used to hold the value of how many months’ donations have been stored in the two-dimensional array arrDonations.

**NOTE:**

* Do not change the code provided.

Complete the code as described in QUESTION 4.1 to QUESTION 4.3 that follow:

* 1. **In the [FormActivate] event handler**

When the program executes the first time, the total amount raised will be the same as the funds raised (donations) for January stored in arrJan.

Assign the values in the arrJan array variable to the arrDonations array variable for both January (column 1) and the total (column 13). (3)

* 1. **Button [Q 4.2]**

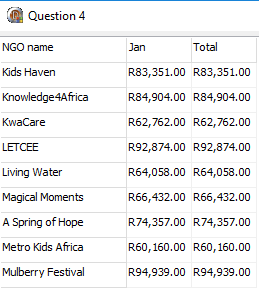
Add code to ensure the correct number of columns are available to display in **stgOutput,** depending on how much data there is to output, e.g. the first time this button executes only 3 columns are needed for the NGO name, January and Total amount raised.



In the first row of **stgOutput**, display “NGO names” in the first column, the names of the months stored in arrMonths, as well as the label Total, in the last column after all the names of the months.

Add code to display all data, currently in arrDonations, in the **stgOutput** component.

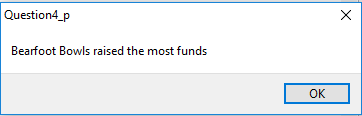
An extract of output:

 (13)

Write and call a user defined method to find the NGO that has raised the most funds, from the data currently stored in the two-dimensional array, using the total column.

A message has been provided to display the name of the NGO, stored in sNGO, that raised the most funds. (6)

An example of output:

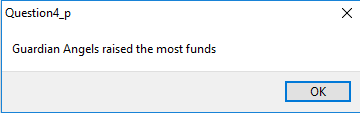


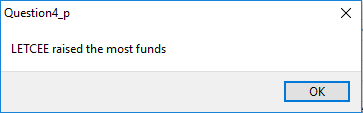
* 1. **Button [Q 4.3]**

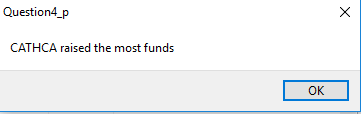
Another month’s donations need to be added. The new month’s donations will be added to column one of **arrDonations**.

Move all the current donations one column to the right in **arrDonations**. Add a random number from 50 000 to 100 000 to each NGOs donations for a new month in the first column of **arrDonations**. Update the total column in arrDonations to include the new value. Ensure that the array will display correctly when **Q 4.2** is clicked.

An extract of output after **Q 4.3** and then **Q 4.2** buttons were clicked three times each:

  (8)

**NOTE:** Due to random numbers being selected, your February and March column will not have the same values, but January should be the same as in the output extract.

You do NOT need to add code to prevent the user from adding more than 12 months.

* Enter your Name and Surname as a comment in the first line of the program file **Question4\_U**.
* Save all the files ('File/Save All').
* Printouts of the code for the unit (**Question4\_U**) will be required.

**[30]**

**GRAND TOTAL: 150**

**ANNEXURE A**

**QUESTION 2: DATABASE INFORMATION**

**DESIGN OF DATABASE TABLES:**

Table: **tblNGO**

This table contains the information of the NGOs.

|  |  |  |
| --- | --- | --- |
| **Field name** | **Data type** | **Description** |
| NGOID | Text (7) | A unique ID number assigned to each NGO |
| NGO | Text (30) | The name of the NGO |
| Province | Text (15) | The province in which the NGO is situated |
| Purpose | Text (25) | The main purpose of the NGO |
| Goal | Currency | The goal amount to be raised in multiples of R100 000 |
| Raised | Currency | The actual amount of money raised so far |

Example of data of the first ten records:

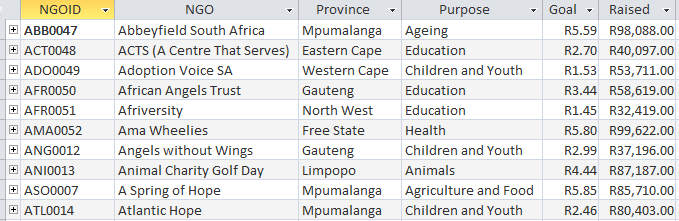
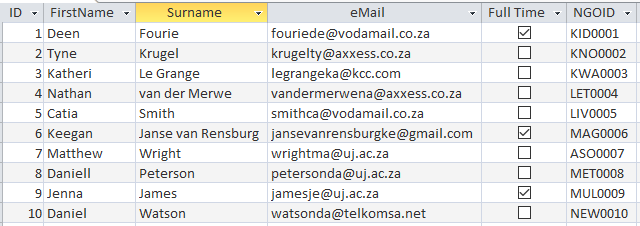


Table: **tblVolunteers**

This table contains the personal information of volunteers that work for the NGOs.

|  |  |  |
| --- | --- | --- |
| **Field name** | **Data type** | **Description** |
| ID | AutoNumber | A unique number assigned to each volunteer |
| FirstName | Text (15) | The first name of the volunteer |
| Surname | Text (20) | The surname of the volunteer |
| eMail | Text (35) | The e-mail address of the volunteer |
| Full Time | Boolean | Boolean field indicating whether the volunteer is available full time or not |
| NGOID | Text (7) | A number that identifies for which NGO the volunteer is giving up their time |

Example of data of the first ten records:



The following one-to-many relationship with referential integrity exists between the two tables in the database:

