education

Department:
Education
PROVINCE OF KWAZULU-NATAL

INFORMATION TECHNOLOGY P1

GRADE 11

JUNE EXAMINATION 2019

Date of examination: 28 MAY 2019

MARKS: 150

TIME: 3 hours

EXAMINER: V B RAMKILAWAN

MODERATORS: MRS R PILLAY
MRS S NUNKUMAR

This question paper consists of 15 pages.

INSTRUCTIONS AND INFORMATION

1. This paper is divided into FOUR sections. Candidates must answer ALL the questions from all FOUR 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 with programming terms that are specific to Delphi programming language. The Delphi programming language must be used to answer the questions.

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 the Delphi programming language for any of these routines.

8. All data structures must be defined by you, the programmer, unless the data structures are supplied.

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 name 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. Your examination number must appear on all the printouts. 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 provided to you on the 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.

Do the following:

- Open the following folder:
  IT_JUNE2019
- Verify that all files listed below are present.

Question1:
Question1P.dpr
Question1P.dproj
Question1P.res
Question1U.dfm
Question1U.pas
valid.jpg
invalid.jpg
q1_3.jpg

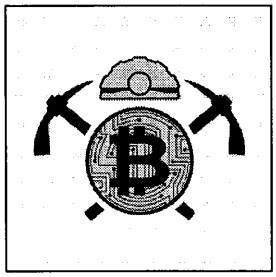
Question2:
Question2P.dpr
Question2P.dproj
Question2P.res
Question2U.dfm
Question2U.pas

Question3:
Miners.TXT
Question3P.dpr
Question3P.dproj
Question3P.res
Question3U.dfm
Question3U.pas

Question4:
Question4P.dpr
Question4P.dproj
Question4P.res
Question4U.dfm
Question4U.pas

SCENARIO

CryptoCurrency is a decentralised method to perform financial transactions. Rather than using a central bank to process all transactions, cryptocurrencies rely on a "blockchain" system where a user is connected to a network and allows their computer to process transactions anonymously. A South African group of programmers have decided to create a new brand of cryptocurrency and you, as part of their team, have to lend your programming skills to get the system running.



SECTION A

QUESTION 1: GENERAL PROGRAMMING SKILLS

The blockchain requires software to ensure efficient and effective processing.

Do the following:

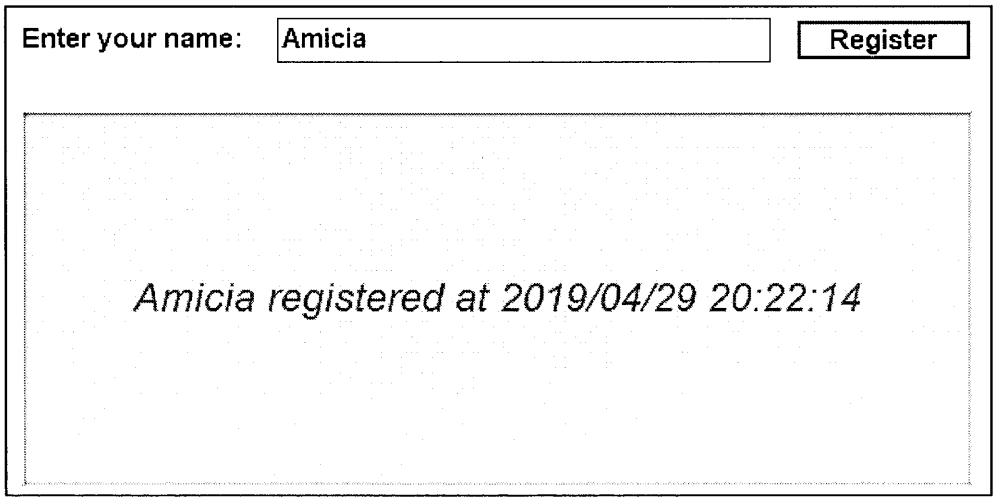
- Open the incomplete program in **Question1** folder.
- Compile and execute the program. The user interface displays FOUR tab sheets labelled Q1.1 to Q1.4. The program has no functionality currently.
- Complete the code for each section of QUESTION 1 as described in QUESTION 1.1 to QUESTION 1.4

1.1 Tab sheet [Q1.1]

Button [Register]

An employee is expected to register using a portal before starting work. The employee's registration data is then displayed boldly for confirmation purposes. Write code to:

- Extract the user's name from Edit Box *edtName1_1*.
- Get the system date and time.
- Combine the user's name with the words "registered at" and the system date and time.
- Display the combined data in the *pnlQ1_1*.
- Set the Panel Font Colour to Red.
- Set the Panel Colour to Black.
- Set the Panel Font Size to 18.
- Set the Panel Font Style to Italic.



(10)

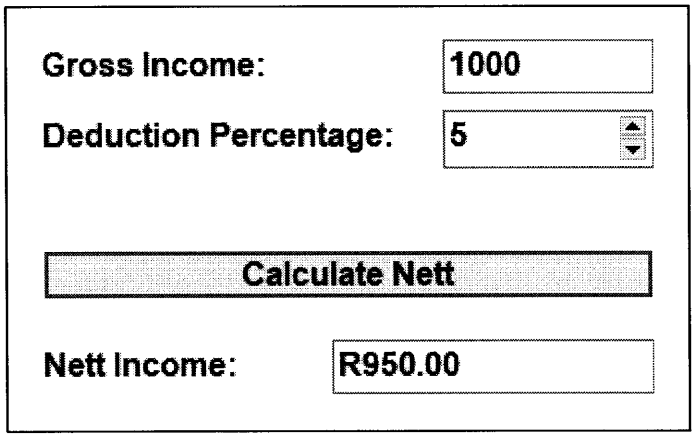
1.2 **Tab sheet [Q1.2]**

Button [Calculate Nett]

Employees are paid weekly. A specific percentage (%) is deducted from an employee's gross salary to cover expenses such as unemployment insurance fund (UIF) and income tax.

Write code to perform the following tasks:

- Obtain the gross income from Edit Box *edtIncome*.
- Obtain the deduction percentage from SpinEdit *sedDeduction*.
- Calculate the deduction amount.
- Calculate the Nett Income by subtracting the deduction amount from the Gross Income.
- Display the Nett Income in Edit Box *edtNett1_2* formatted as currency.



(8)

1.3 **Tab sheet [Q1.3]**

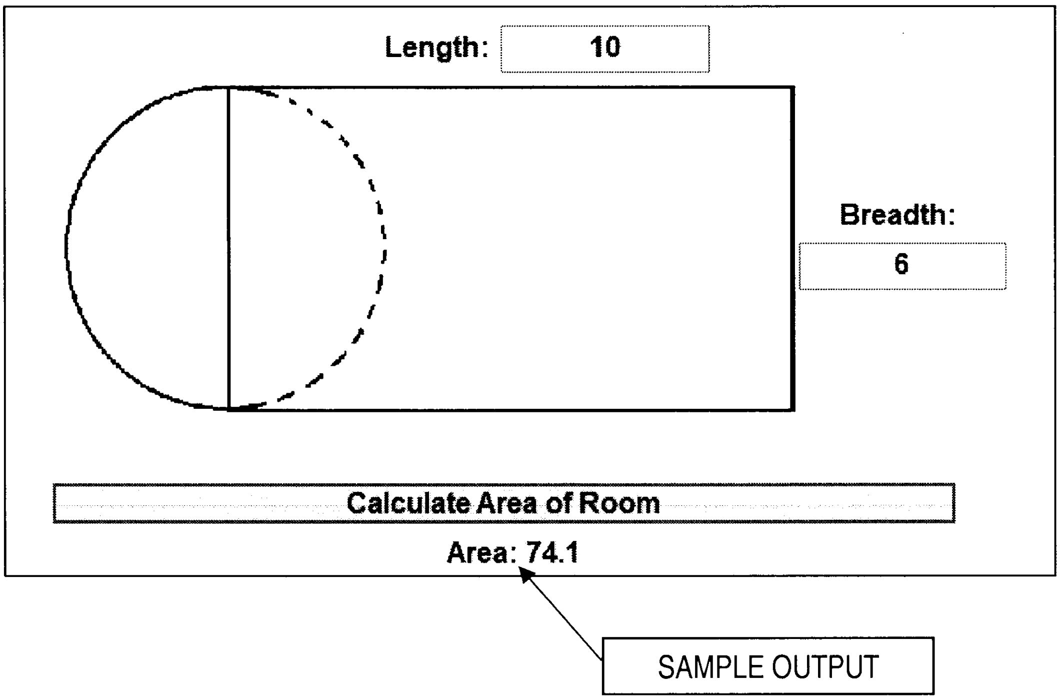
Button [Calculate Area of Room]

The server room where the CryptoCurrency blockchains will be maintained have a very special design. The room is rectangular in shape with a semicircle design on one end. The floor area of this server room is required.

Write code to perform the following tasks:

- Extract data for the length and breadth of the room from Edit Boxes *edtLength* and *edtBreadth* respectively.
- Use the values to determine the area of the room.
- Display the area in *lblArea1_3* with a suitable caption. The area must be displayed with 1 decimal place.

NOTE:
Area of a Rectangle = Length x Breadth
Area of a Circle = πr^2



(16)

60 + 113.10
6 + 113.2
173.2

1.4 **Tab sheet [Q1.4]**

Button [Check Transaction Code]

Each CryptoCurrency transaction is given a unique transaction code which is used to securely track the status of the transaction.

A transaction code:

- is eight (8) characters in length
- has the following format:
  - 1 letter; followed by
  - 2 digit number; followed by
  - 3 letters; followed by
  - 1 digit number; followed by
  - 1 check digit
- The last character in a transaction code is the check digit
- The **check digit** for the transaction code: H74VZM51 is 1 and the check digit for the transaction code: J32OIL85 is 5

The check digit is used to verify whether the transaction code is valid or invalid using the following steps:

- **STEP 1**
  Determine the sum of the second and third characters.
  Example: For transaction code H74VZM51: 7+4 = 11

- **STEP 2**
  Multiply the seventh character by 2.
  Example: For transaction code H74VZM51: 5 x 2 = 10

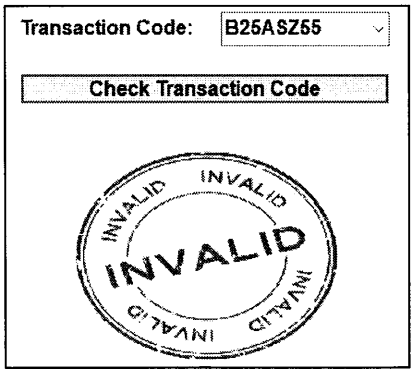
- **STEP 3**
  Calculate the absolute difference between the answers from Step 1 and Step 2.
  HINT: Use the ABS function from the Math class.
  Example: For transaction code H74VZM51: 11 – 10 = 1

- **STEP 4**
  If the answer in Step 3 is equal to the check digit, then the transaction code is valid; otherwise the transaction code is invalid.
  Example: The transaction code H74VZM51 is valid because the answer in step 3 equals (=) to check digit

Write the **code to:**

- Test
  - Extract the transaction code from ComboBox *cmbTransaction1_4*.

for the validity of the transaction code using the steps described above



(16)

• Enter your name as a comment in the first line of the program file • Save your program • Print the code if required

TOTAL SECTION A: 50

SECTION B

QUESTION 2: ARRAYS

Before the CrypoCurrency goes live, various employees participate in simulated tests using the blockchain software to ensure everything is functioning correctly.

Two arrays have been declared and populated:

arrNames: A String array containing 10 elements with the names of various staff members who tested the software with sample transactions

arrTransactions: An integer array containing 10 elements where each element represents the number of CryptoCoins that were transacted by the corresponding individual in arrNames

NOTE: ***arrNames and arrTransactions are parallel arrays***

Do the following:

- Compile and execute the program in the **Question2** folder. The program currently has limited functionality

2.1 **BUTTON [Display Transactions]**

Write code to:

- Display arrays arrNames and arrTransactions in neat columns in the Output Area (redOut).
- Include a 3rd column which will display the calculated Rand value of the specified number of coins formatted currency.

  NOTE: 1 coin = R5.25

All columns should include headings as shown in the screenshot below.

Name	Coins	Amount
Jack	8	R42.00
Kate	4	R21.00
Sawyer	16	R84.00
Locke	2	R10.50
Jack	3	R15.75
Ben	15	R78.75
Kate	42	R220.50
Claire	4	R21.00
Locke	10	R52.50
Locke	8	R42.00

(12)

2.2 **BUTTON [Determine Average]**

Write code to:

- Determine the average number of coins transacted.
- Display the average (formatted to 2 decimal places) in the Output Area with an appropriate message, as shown in the sample data below.

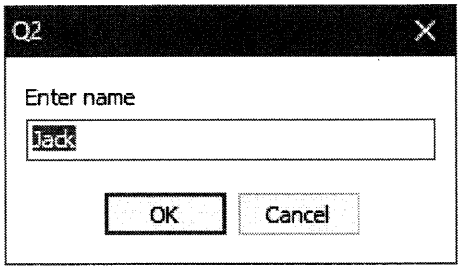
Name	Coins	Amount
Jack	8	R42.00
Kate	4	R21.00
Sawyer	16	R84.00
Locke	2	R10.50
Jack	3	R15.75
Ben	15	R78.75
Kate	42	R220.50
Claire	4	R21.00
Locke	10	R52.50
Locke	8	R42.00
Average coins: 11.20		

(8)

2.3 **BUTTON [Total per User]**

Write code to:

- Prompt the user to enter the employee's name using an Input Dialogue Box.
- Loop through arrNames, counting the number of times the employee's name appears in arrNames.
- Determine the total number of coins transacted by the employee name provided.
- Display the number of transactions and total coins transacted in the output area as shown in the sample data below. (15)



Name	Coins	Amount
Jack	8	R42.00
Kate	4	R21.00
Sawyer	16	R84.00
Locke	2	R10.50
Jack	3	R15.75
Ben	15	R78.75
Kate	42	R220.50
Claire	4	R21.00
Locke	10	R52.50
Locke	8	R42.00
Average coins: 11.20		
Jack has 2 transactions.		
Total coins: 11		

- Enter your name as a comment in the first line of the program file
- Save your program
- Print the code if required

TOTAL SECTION B: 35

SECTION C

QUESTION 3: STRING HANDLING AND TEXT FILES

Transactions are tracked in an electronic ledger which uses encryption techniques to prevent outsiders from eavesdropping on transactions. Users who enable mining on their systems are logged in a text file for reward purposes.

Do the following:

- Open the incomplete program in the **Question3** folder.
- Compile and execute the program

3.1 A user who wants to join the CryptoMining programme, is required to register with the company using a username

3.1.1 **BUTTON [Validate]**

The username is constructed as follows:

- The username must begin with an @ symbol;
- Have a minimum of 9 characters in total

Examples:

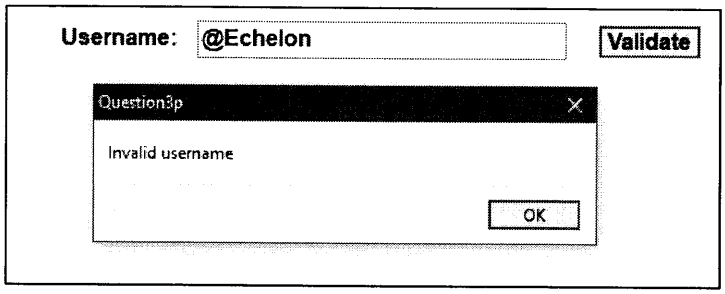
Username	Valid?	Reason
@Echelon	No	< 9 characters in total
RedEchelon	No	Does not start with @ symbol
@RedEchelon	Yes	Starts with @ symbol. >= 9 characters

Write code to:

- Extract the username provided by the user from Edit Box *edtUserName*.
- Validate the username using the criteria provided above.
  NOTE: You only need to validate the presence of the "@" symbol at the beginning of the username and the total length. You do **NOT** have to validate the individual characters.

- If the name is valid make the Group Box *grpProcess* visible.

- If the username is invalid:
  - Clear the Edit Box (*edtUserName*).
  - Set the typing cursor to the Edit Box.
  - Display a dialogue box informing the user that the name is invalid.



(8)

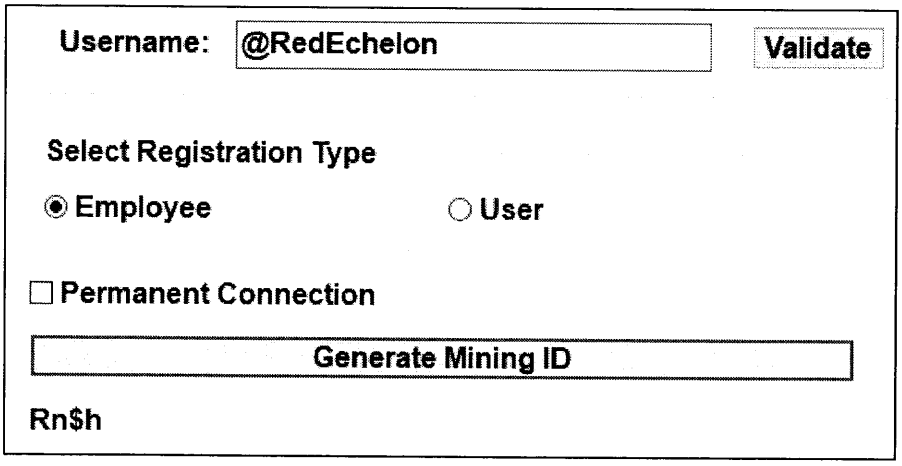
3.1.2 BUTTON [Generate Mining ID]

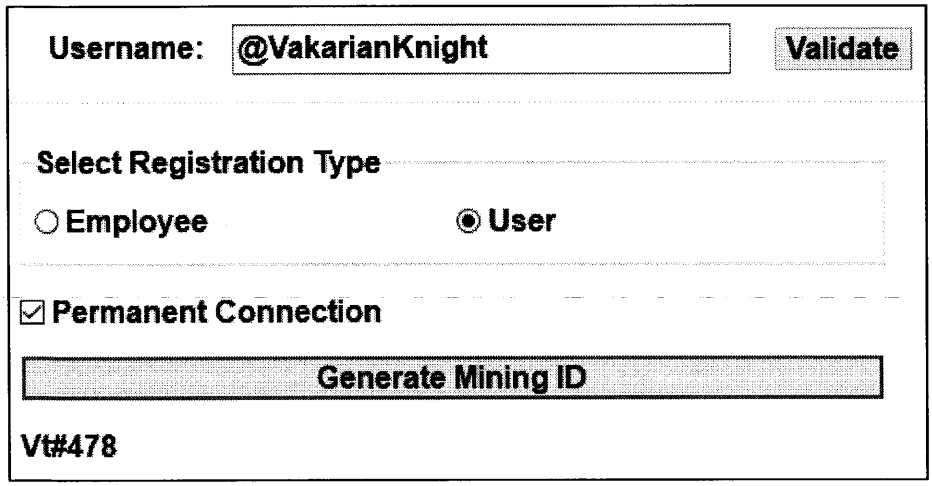
A unique Mining ID is generated for the user based on the input from 3.1.1

Generate the Mining ID by combining the following:

- Copy the second and last characters from the username.
- Extract data from the Radio Group *rgpRegType*. If the ~~miner~~ is an Employee, the $ symbol should follow.

  If the ~~miner~~ is a ~~user~~, a # symbol should follow.
- Extract data from Checkbox *chbPermanent*. If the miner enables a permanent connection to the Blockchain, generate a random 3 digit number and join this number to the Mining ID.
  If the miner has not enabled a permanent connection to the Blockchain, the Mining ID should end with a randomly selected character from the username.
- Display the generated Mining ID in *lblMiningCode*.





(15)

3.2 **BUTTON [Load Data]**
CryptoMiner details are recorded in a textfile MINERS.TXT.

Storing this text file without any security would put the system at risk. As a result, a miner's name is obfuscated (hidden) by inserting dummy data between the actual data.

The actual data is found at the even positions of the provided text. **For example:** FTBaYlHoKnU represents the name Talon

F	T	B	a	Y	l	H	o	K	n	U
1	**2**	3	**4**	5	**6**	7	**8**	9	**10**	11

First 3 lines from the Text File:

rMMhUlOeCnDgKiB
SAWsIaNnCdHaI
VABnOdKiFlVeK

Write code to perform the following tasks:

- Check if file MINERS.TXT exists. If it does not exist, display an error message and terminate the app.
    - Connect to the file and open it for reading.
    - Loop through the text file, extracting a single line from it each time the loop runs.
    - Process the extracted line, copying only the characters at even positions in the extracted line.
    - Display the processed line (decrypted) in the Output Area (*redOut*).

(22)

- Enter your name as a comment in the first line of the program file
- Save your program
- Print the code if required

TOTAL SECTION C: 45

SECTION D

QUESTION 4: PROBLEM SOLVING

CryptoCurrency transactions are processed by millions of computers connected to the peer-to-peer network globally. This process is known as "mining" and works by providing several miners with a mathematical problem. The first miner to solve the problem processes the transaction and earns a reward in this way.

Do the following:

- Compile and execute the program in the **Question4** folder. The program currently has no functionality

The problem:

Two equations are provided to a miner:

Equation 1: $x + 3y = Answer\ A$

Equation 2: $y + 8x = Answer\ B$

Answer A and *Answer B* are provided to the miner. It is the miner's job to use this data to find the values of *x* and *y* that will satisfy both equations.

For example: If *Answer A* = 220 and *Answer B* = 380, then :
x = 40 and *y* = 60 ;

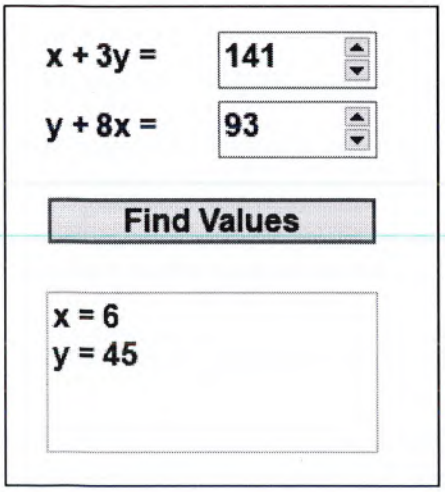
because:
40 + (3 x 60) = 220 (**Equation 1**)
60 + (4 x 40) = 380 (**Equation 2**)

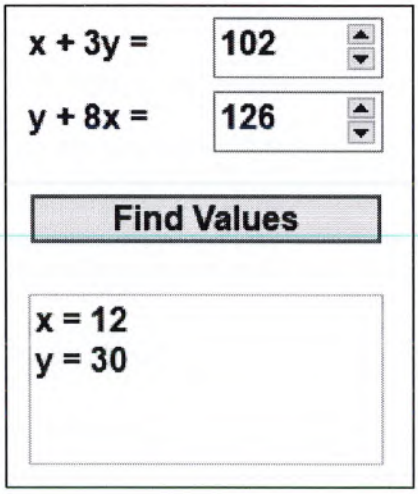
NOTE:
The values of both *x* and *y* range from 1 to 100 (inclusive).

You are required to code **Button [Find Values]** to:

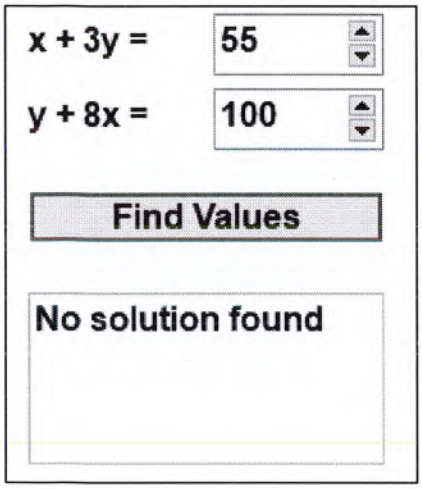
- Extract the value of Answer A from SpinEdit *sedEq1*.
- Extract the value of Answer B from SpinEdit *sedEq2*.
- Determine the value of x and y for which both equations are satisfied at the same time (simultaneously).
- **HINT:** Your looping structure should terminate when either**:**
  - x and y are found; **OR** when;
  - Every possible combination of x and y has been tested.

- If x and y are found, display the values in the Output Area (redOut).





- If x and y was not found, display "No solution found" in the Output Area.



(20)

- Enter your name as a comment in the first line of the program file
- Save your program
- Print the code if required

TOTAL SECTION D: 20

GRAND TOTAL: 150