basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

NATIONAL SENIOR CERTIFICATE

GRADE 12

INFORMATION TECHNOLOGY P1

FEBRUARY/MARCH 2015

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 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 only 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. All data structures must be defined by you, the programmer. You may not use components provided within the interface to store and later retrieve data.

9. You must save your work regularly on the disk you have been given, or the disk space allocated to you for this examination.

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 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 in the form of a password-protected executable file:

    - Delphi learners must use the file **DelphiDataENG.exe**.
    - Java learners must use the file **JavaDataENG.exe**.

    Do the following:

    - Double click on the file.
    - Click on the extract button.
    - Enter the following password: **ProperTies87&%**

    Once the file has been extracted, the following list of files will be available in the folder **DelphiDataENG/JavaDataENG:**

Delphi files	**Java (Netbeans) files**
Question1:	**Question1:**
Question1_P.dpr	metre1.png
Question1_P.res	Question1.form
Question1_U.dfm	Question1.java
Question1_U.pas	
Question2:	**Question2:**
Abdul Brouwer.txt	Abdul Brouwer.txt
HomeLoan_U.pas	Paula Redhat.txt
Paula Redhat.txt	Xavier Outland.txt
Question2_P.dpr	
Question2_P.res	HomeLoan.java
Question2_U.dfm	Question2.form
Question2_U.pas	Question2.java
Xavier Outland.txt	
Question3:	**Question3:**
Question3_P.dpr	PopulateArrays.java
Question3_P.res	Question3.form
Question3_U.dfm	Question3.java
Question3_U.pas	

SCENARIO

You are employed as a software developer at a company that buys, renovates and sells properties.

SECTION A

QUESTION 1: GENERAL PROGRAMMING SKILLS

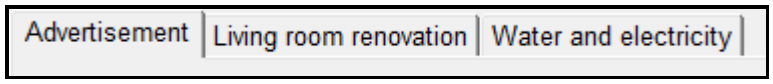
INSTRUCTIONS:

Delphi programmers	Java programmers
• The project **Question1** is provided in the **Delphi** folder.	• The project **Question1** is provided in the **Netbeans** folder.
• Open the incomplete project file **Question1_P.dpr** in the **Question1** folder.	• Open the incomplete class called **Question1.java** in the folders **Source Packages (src)** and **Question1Package**.
• Add your examination number as a comment in the first line of the main form unit **Question1_U.pas**.	• Add your examination number as a comment in the first line of the class **Question1.java**.

Do the following:

- Compile and execute the program. The interface displays three different tabs, namely Advertisement, Living room renovation and Water and electricity.

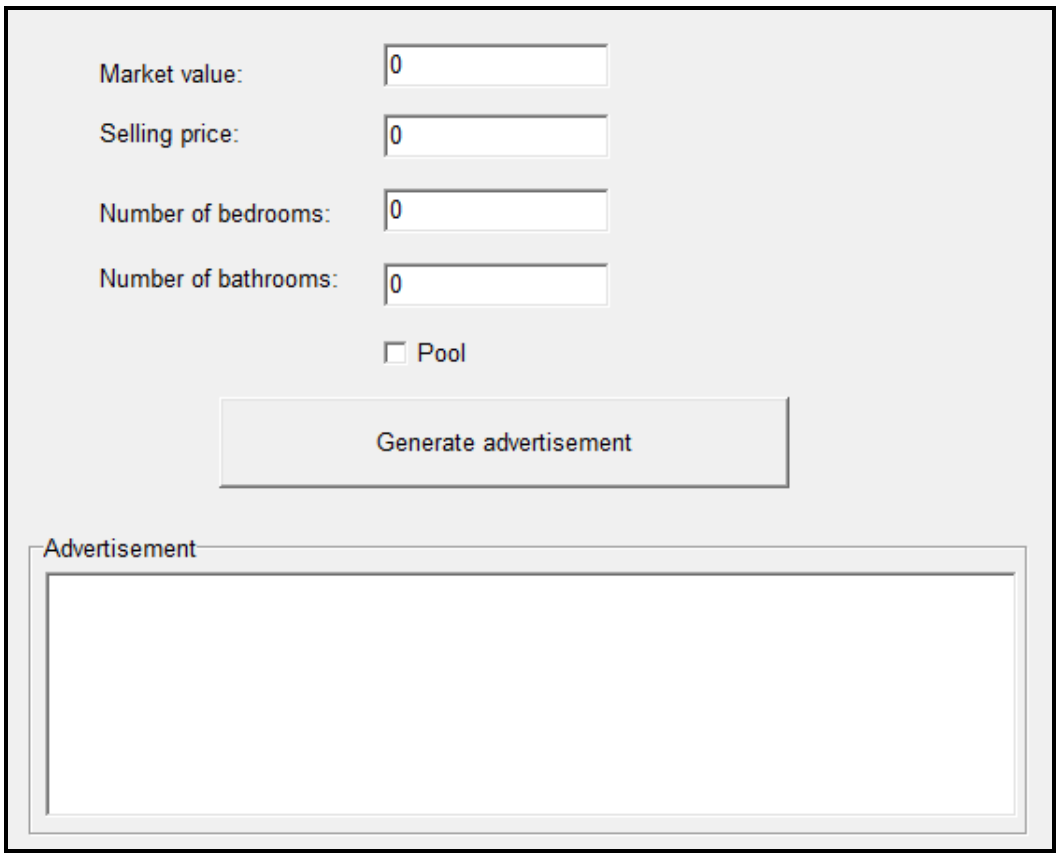
  Example of the tab sheets:



- Complete the code for each tab as described in QUESTION 1.1 to QUESTION 1.3.

  NOTE: QUESTION 1.1 refers to the tab Advertisement.
  QUESTION 1.2 refers to the tab Living room renovation.
  QUESTION 1.3 refers to the tab Water and electricity.

1.1 Interface for the Advertisement tab



NOTE: The **market value** of a property is the municipal value that the property is assessed for.

The **selling price** of a property is the value of the sale of the property.

[Button – Generate advertisement]

Write code to do the following:

- Use information from the provided text boxes to compile an advertisement in the following format:

```
House for sale:
<selling price>#<number of bedrooms>Bed#<number of bathrooms>Bath#
```

Use the following information to compile the advertisement:

- The selling price contained in the advertisement must be formatted to a currency. Do not display any decimal values.
- If the Pool check box is selected, the text 'Pool#' must be added to the advertisement text.
- If the selling price of the property is less than the market value, the word 'Bargain' must be added to the advertisement text.

- Display the advertisement information in the output area provided.

Examples of compiled advertisements:

Example 1: If the market value of the property is R950 000, the selling price is R850 000 and the house has three bedrooms, two bathrooms and a pool, the advertisement should be displayed as follows:

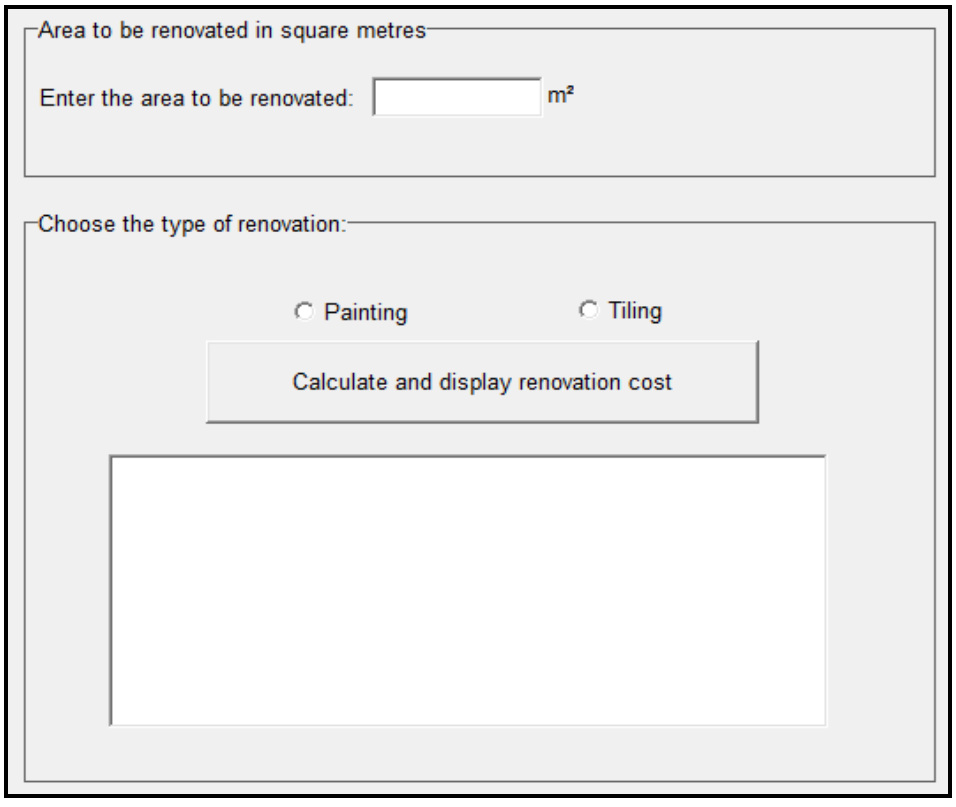
House for sale:
R850000#3Bed#2Bath#Pool#Bargain

Example 2: If the market value of the property is R750 000, the selling price is R800 000 and the house has two bedrooms, one bathroom and no pool, the advertisement should be displayed as follows:

House for sale:
R800000#2Bed#1Bath#

(11)

1.2 Interface for the Living room renovation tab



Write code to do the following:

1.2.1 [Radio Button – Painting]

If the radio button Painting is selected, use a variable to save the selected type of renovation as the character P. (1)

1.2.2 **[Radio Button – Tiling]**

If the radio button Tiling is selected, use a variable to save the selected type of renovation option as the character T. (1)

1.2.3 **[Button – Calculate and display renovation cost]**

The user is required to enter the number of square metres to be renovated as an integer value and select the type of renovation to be done. The cost of each type of renovation must be calculated as explained below.

(a) If the type of renovation is **Painting**, use the following information to calculate how much paint will be required as well as the total cost:

- One litre of paint is used to paint an area of eight square metres.
- The pricing structure of paint per drum is as follows:
    - 1-litre drum R 55.50
    - 2-litre drum R 92.30
    - 5-litre drum R 199.00

Display the calculated values in the provided output area using the following format:

Area: **<area>** square metres
Volume of paint required: **<volume>** litre

1-litre drums: **<number of 1-litre drums>**
2-litre drums: **<number of 2-litre drums>**
5-litre drums: **<number of 5-litre drums>**

Total cost: **<Total cost of paint>**

The total cost must be displayed rounded to two decimal places.

Example: If the area entered is 148 m^2, the output will be as follows:

Area: 148 square metres
Volume of paint required: 18.5 litres

1-litre drums: **0**
2-litre drums: **2**
5-litre drums: **3**

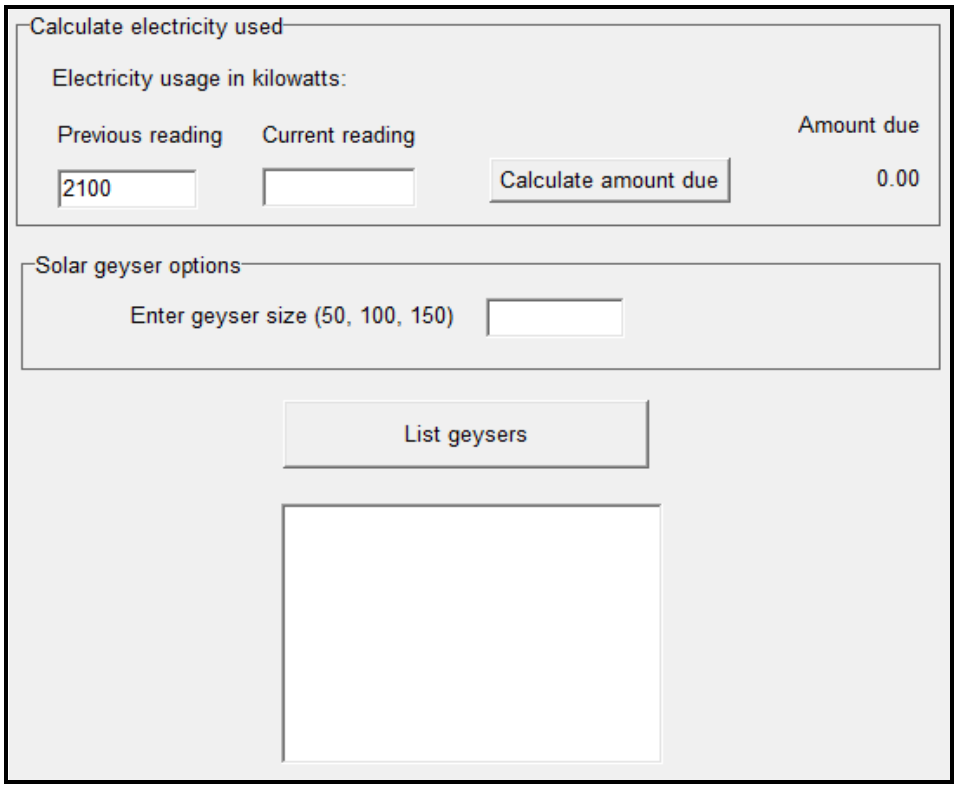
Total cost: R 781.60 (17)

(b) If the type of renovation is **Tiling**, a dialog box must be used to allow the user to enter the cost of tiling per square metre. Add five square metres to the original square metre value that was entered to provide for breakages, and then calculate the cost. Display the area to be tiled in square metres. The total cost must be displayed rounded to two decimal places.

Example: If the area entered is 100 m^2 and the cost of tiling is R50,00 per square metre, the following will be displayed:

Area: 100 square metres
Total cost: R 5250.00 (6)

1.3 Interface for the Water and electricity tab



Write code to do the following:

1.3.1 [Button – Calculate amount due]

The previous reading for electricity used is supplied in a text box as an integer. The user must enter the current reading in a text box as an integer.

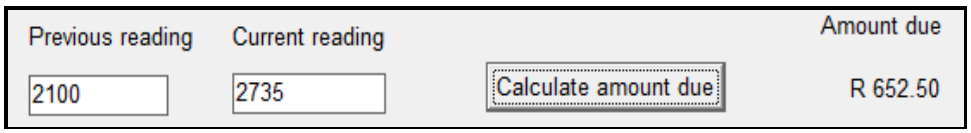
- If the current reading entered by the user is less than the previous reading, display a suitable message and clear the current reading text box.

- If the current reading is equal to or greater than the previous reading, use the following tariffs to calculate the amount due for electricity used:

Units	Tariffs
0–600	R1,00 per unit
>600	R600 + R1,50 per unit for all units that exceed 600

- Display the amount in the label provided in a format that includes a currency correct two decimal places.

Example:



(11)

1.3.2 **[Button – List geysers]**

In an effort to save electricity, solar geysers are investigated as a possibility. Three sizes of solar geysers are available based on their capacity in litres. The user is required to enter one of the three sizes: 50 (litres), 100 (litres) or 150 (litres).

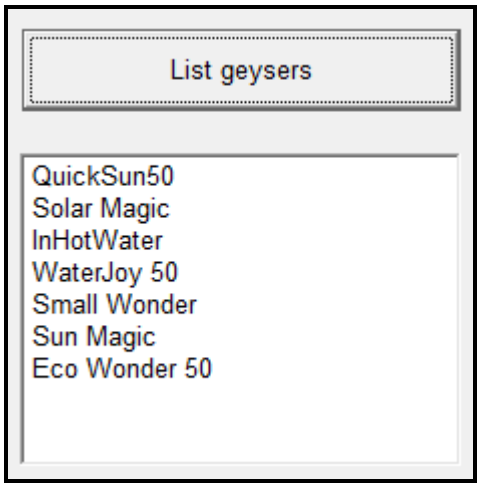
An array that contains strings of text which each describes a type of geyser is provided as part of the supplied code.

The description of each type of geyser starts with the capacity of the geyser in litres, followed by a dash and the brand name of the geyser.

Example: The geyser described as '50-Small Wonder' can hold 50 litres of water and the brand name of the geyser is 'Small Wonder'.

Write code to display the types of geysers in the array that matches the capacity entered by the user when the **List geysers** button is clicked. No validation of the input value is required.

Example: List of geysers when the value 50 is entered:



(6)

- Ensure that your examination number is entered 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: 53

SECTION B

QUESTION 2: OBJECT-ORIENTATED PROGRAMMING

Clients who want to buy properties usually apply for home loans. The following information about submitted applications are usually required:

- Does the applicant qualify for a home loan?
- What is the status of the application based on the duration of the loan in years and the interest rate?
- What is the instalment amount to be paid each month?

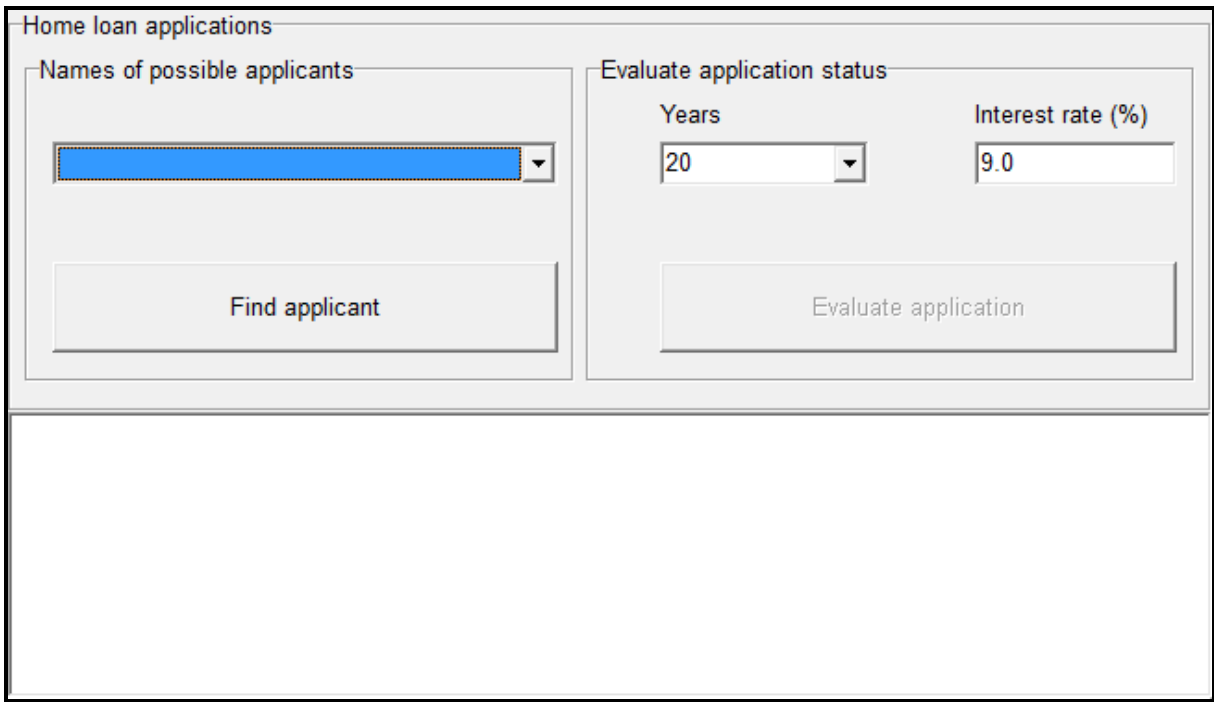
INSTRUCTIONS:

Delphi programmers	Java programmers
• The project **Question2** is provided in the **Delphi** folder which contains: o A main form file called **Question2_U.pas** o An incomplete unit file called **HomeLoan_U.pas** o A text file for each applicant	• The project **Question2** is provided in the **Netbeans** folder which contains: o A GUI class file called **Question2.java** o An incomplete object class called **HomeLoan.java** o A text file for each applicant
• Open the incomplete project file called **Question2_P.dpr** in the **Question2** folder.	• Open the incomplete file called **Question2.java** and **HomeLoan.java** in the folders **Source Packages (src)** and **Question2Package**.
• View (Ctrl+F12) the unit file **HomeLoan_U.pas** and add your examination number as a comment in the first line of both unit files **Question2_U.pas** and **HomeLoan_U.pas**.	• Add your examination number as a comment in the first line of the classes **Question2.java** and **HomeLoan.java**.

Do the following:

- Compile and execute the program. Currently the program has no functionality.

  Example of the interface (see next page):



- Complete the code as described in QUESTION 2.1 and QUESTION 2.2 to add functionality to the program.

2.1 Complete the code in the home loan class **THomeLoan/HomeLoan** as described below.

2.1.1 Write code to add the following five attributes to the class:

Description	**Names of attributes**	
	Delphi	**Java**
Name of applicant	fApplicantName	applicantName
Disposable income	fDisposableIncome	disposableIncome
Home loan amount applied for	fLoanAmount	loanAmount
Number of years to repay the home loan	fYears	years
Interest rate	fInterestRate	interestRate

(4)

2.1.2 Write code to create a constructor which receives the following parameter values:

name of applicant
disposable income
loan amount applied for

Initialise the relevant attributes using the parameter values. Initialise the attributes for the number of years and the interest rate to 0. (5)

2.1.3 Write an accessor method for the name of the applicant. (2)

2.1.4 Write mutator methods for the attributes Years and Interest rate. (4)

2.1.5 A method called **calculateInstalmentAmount** has been provided. Uncomment the code that calculates the monthly instalments in order to make the method functional. (1)

2.1.6 Write code for a method called **isApproved** to return a Boolean value indicating whether the home loan has been approved or not.

A home loan will only be approved if the following criteria are met:

- Loans of more than R800 000 must be repaid over a period of more than 20 years (25 or 30 years).
- The monthly disposable income of an applicant who applies for a loan up to R600 000 must be at least 130% of the monthly instalments. (9)

2.1.7 Write code to create a **toString** method to return a string formatted as follows:

```
Name of applicant: <applicant name>
Disposable income: <disposable income>
Loan amount: <loan amount>
Number of years: <number of years>
Interest rate: <interest rate> %
```

(5)

2.2 You have been provided with **three text files** which each contains information about a different home loan application. The name of each text file is the name of the applicant.

Example: If the applicant's name is 'Abdul Brouwer', the name of the text file is **Abdul Brouwer.txt**.

Each text file contains four lines of information about a home loan applicant in the following format:

<name and surname of applicant>
<monthly income>
<monthly expenditure>
<home loan amount applied for>

Content of the three text files for each of the following applicants:
Abdul Brouwer, Paula Redhat and Xavier Outland

Abdul Brouwer	Paula Redhat	Xavier Outland
28000	25785	44741
12500	11190	20384
750000	888000	978000

The GUI is supplied with the names of different applicants stored in the combo box that is provided. An object of the **THomeLoan**/**HomeLoan** class has been declared globally.

2.2.1 **[Button – Find applicant]**

The user is required to select the name of an applicant from the combo box.

Write code to do the following:

- Test whether an applicant was selected or not and display a suitable message.
- Test whether a text file with the applicant's name exists or not, and display a suitable message which includes the applicant's name, and whether an application was submitted or not.
    - If a text file is not found for the applicant selected from the combo box, no home loan application was submitted.

      Example of output of a home loan application by John Smith that was not submitted:

      ```
      No home loan application was submitted by
      John Smith.
      ```

      If a home loan application was not submitted, disable the 'Evaluate application' button.
    - If a text file with the applicant's name exists (was submitted), do the following:
        - Read the income, expenditure and loan amount from the text file.
        - Calculate the disposable income by subtracting the expenses from the income.
        - Instantiate a **THomeLoan**/**HomeLoan** object for the applicant.
        - Enable the button 'Evaluate application'. (18)

2.2.2 **[Button – Evaluate application]**

The user must select the number of years for the loan to be repaid from the combo box provided and enter the interest rate in the text box provided for this purpose.

Write code to do the following:

- Use the mutator methods and the information provided by the user to update the applicable attributes of the home loan object for the applicant.

- Test whether the loan was approved (adhered to the requirements) or not.

    - If the loan was approved, display the details of the loan application in the output area using the **toString** method. Display a message with the following format:

      Loan APPROVED with a monthly instalment of **<monthly instalment>**

      Example of output for the approved home loan application by Abdul Brouwer for R750 000 over 20 years at an interest rate of 9%:

```
Name of applicant: Abdul Brouwer
Disposable income: R 15500.00
Loan amount: R 750000.00
Number of years: 20
Interest rate: 9.0%

Loan APPROVED with a monthly instalment of
R 6847.00
```

    - If the home loan was not approved, display the applicant's name and surname and a message indicating that the loan was not approved.

      Example of output for the home loan application by Xavier Outland for R870 000 over 20 years at an interest rate of 9%, which was not approved:

```
Name of applicant: Xavier Outland
Loan NOT APPROVED
```

(12)

- Ensure that your examination number is entered as a comment in the first line of the class and the form.
- Save all the files.
- A printout of the code in both classes/units may be required.

TOTAL SECTION B: 60

SECTION C

QUESTION 3: PROBLEM-SOLVING PROGRAMMING

Six estate agents are employed at a local estate agency. The manager of the agency requires software that is able to evaluate the profiles of their sales agents annually.

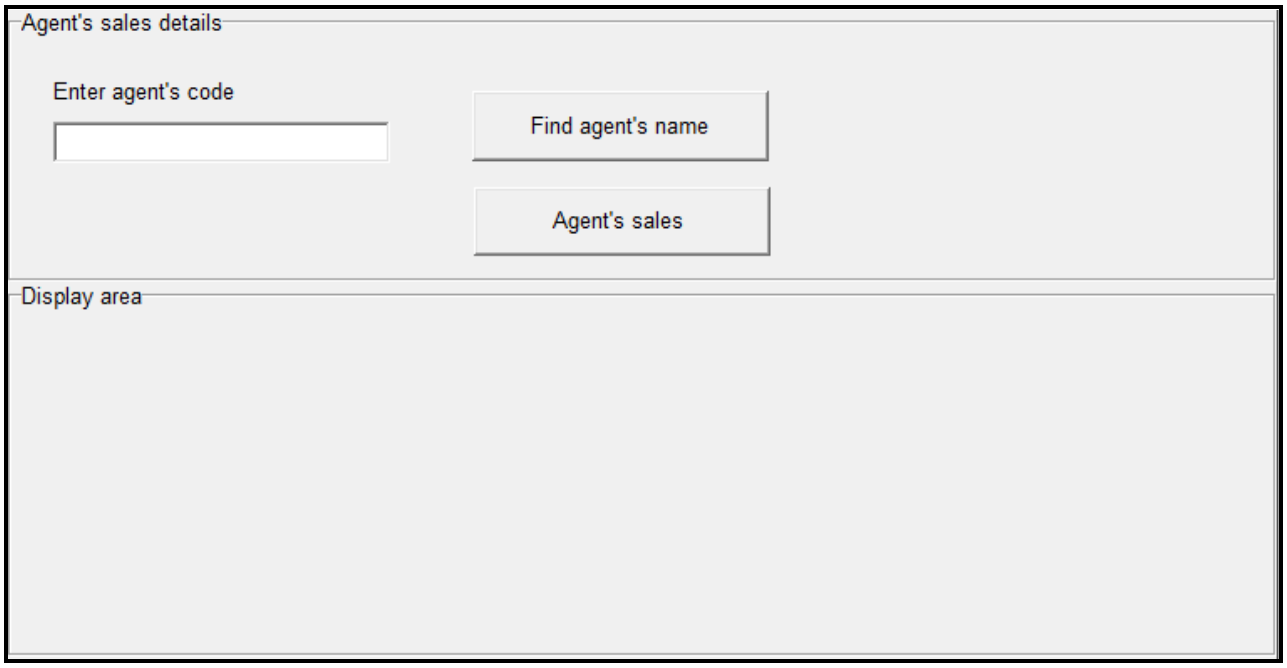
INSTRUCTIONS:

Delphi programmers	Java programmers
• The project **Question3** is provided in the **Delphi** folder. • Open the incomplete project file **Question3_P.dpr** in the **Question3** folder. • Add your examination number as a comment in the first line of the main form unit file **Question3_U.pas**.	• The project **Question3** is provided in the **Netbeans** folder. • Open the incomplete classes called **Question3.java** and **PopulateArrays.java** in the folders **Source Packages (src)** and **Question3Package**. • Add your examination number as a comment in the first line of the class file **Question3.java**.

Do the following:

- Compile and execute the program. Currently the program has no functionality.

  Example of interface:



Two arrays called **arrAgents** and **arrSales** are supplied. You may create and use additional arrays as part of your solution.

The array **arrAgents** stores the codes and names of a number of sales agents in the following format:

```
<Agent's code>:<Agent's name>
```

Example:

A120:Wes Seelig

NOTE: The information of sales agents with agent's codes A120–A125 is supplied.

The array **arrSales** stores information about sold properties in the following format:

```
<month>#<type>#<agent's code>;<price>
```

Description of information provided in one line of text:

- **month:** The number of the month of the year in which the sale transaction was conducted
- **type:** The type of property sold:
  - **C:** Commercial
  - **R:** Residential
  - **A:** Agricultural
- **agent's code:** The code of the agent who sold the property
- **price:** The price at which the property was sold

Example:

02#C#A123;650000

- Complete the code for QUESTION 3 as described in QUESTION 3.1 and QUESTION 3.2 below.

3.1 **[Button – Find agent's name]**

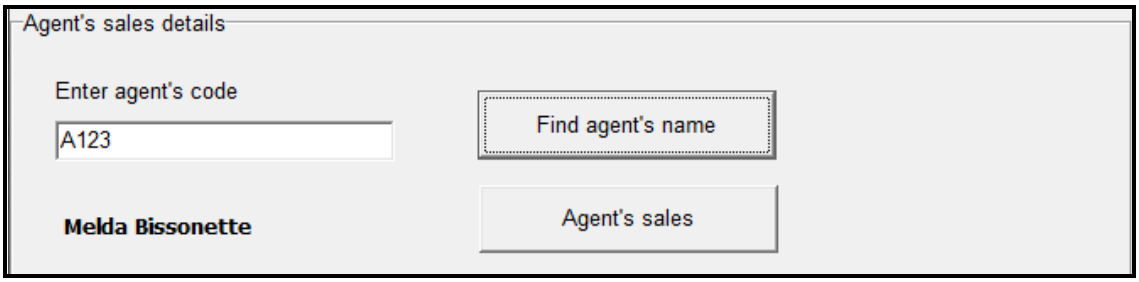
The user is required to enter the code for an agent in the text box provided.

Write code that will search the **arrAgents** array for the code of the agent that was entered by the user. The search process should be constructed in such a way that the search process stops when the agent's code is found.

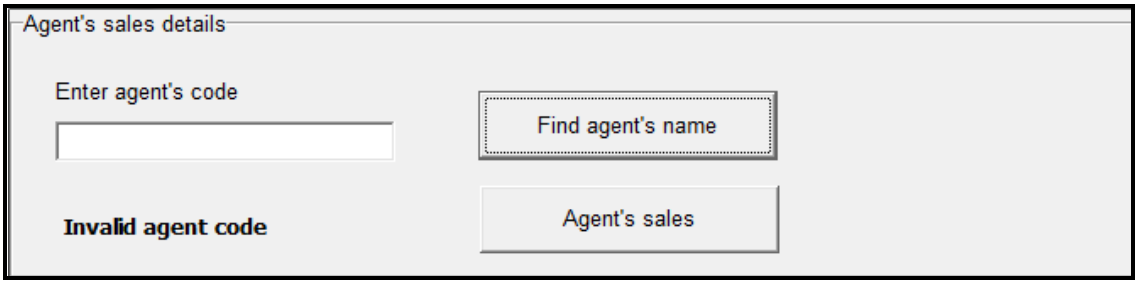
If the agent's code is found, display the agent's name in the label provided for the name of the agent.

If the agent's code is not found, display the message 'Invalid agent's code' in the label which is provided for the name of the agent and clear the text box provided for entering the agent's code.

Example of output if the agent's code entered is A123:



Example of output if the agent's code entered is B123:



(11)

3.2 **[Button – Agent's sales]**

3.2.1 Add a suitable output component in the **Display area** on the GUI to provide for the output for QUESTION 3.2.2. Rename the component with an appropriate name. (2)

3.2.2 A summary of the sales conducted by the sales agent identified in QUESTION 3.1 must be displayed in the component added in the GUI in QUESTION 3.2.1. Use the information in the **arrSales** array and summarise the sales of the agent as shown below.

Example of output if the agent's code is A123:

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	
Commercial	1	3	3	5	0	0	1	2	0	2	1	6	24
Residential	0	0	1	0	0	1	0	0	0	0	0	0	2
Agricultural	0	0	3	4	0	0	3	1	3	0	0	0	14

Total value of sales: R 79640000

NOTE: The values displayed in the last column indicate the total number of the different types of properties sold by this agent.

The total value of sales displayed below the summary indicates the total value of all properties sold by this agent.

Any other suitable output component may be used other than the one used in the given example. (24)

- Ensure that your examination number is entered 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: 37
GRAND TOTAL: 150