basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

NATIONAL SENIOR CERTIFICATE

GRADE 12

INFORMATION TECHNOLOGY P1

NOVEMBER 2014

MEMORANDUM

MARKS: 150

This memorandum consists of 28 pages.

GENERAL INFORMATION:

- These marking guidelines are to be used as the basis for the marking session. They were prepared for use by markers. All markers are required to attend a rigorous standardisation meeting to ensure that the guidelines are consistently interpreted and applied in the marking of candidates' work..

- Note that learners who provide an alternate correct solution to that given as example of a solution in the marking guidelines will be given full credit for the relevant solution, unless the specific instructions in the paper was not followed or the requirements of the question was not met

- **Annexures A, B and C** (pages 3-8) include the marking grid for each question for using either one of the two programming languages.

- **Annexures D, E, and F** (pages 9-16) contain examples of solutions for Java for Questions 1 to 3 in programming code.

- **Annexures G, H and I** (pages 17-28) contain examples of solutions for Delphi for Questions 1 to 3 in programming code.

- Copies of **Annexures A, B and C** (pages 3-8) should be made for each learner and completed during the marking session.

ANNEXURE A:

SECTION A:

QUESTION 1: MARKING GRID - GENERAL PROGRAMMING SKILLS

CENTRE NUMBER:	**EXAMINATION NUMBER:**		
QUESTION	**DESCRIPTION**	**MAX. MARKS**	**LEARNER'S MARKS**
	If a learner has a problem reading from a combo box, penalise only once for the error.		
1.1	**Button - [Confirm delivery]** Extract departure from combobox; (to be used as text) AND Extract destination from combobox; (to be used as text)✓ Extract the number of kilometres from text box✓ convert to number✓ (can also be converted in 1.2) Create the string to join departure, destination and distance✓ and assign to the label provided✓	**5**	
1.2	**Button - [Delivery cost]** Extract the choice selected from the list box✓ Correct variable in condition✓ All 4 possibilities (A1..A4 OR item index 0..3)✓ Correct selection structure (if/case/switch)✓ Use the correct tariff for each option✓ Calculate cost: tariff x distance✓ Check if speed post is selected✓, add 100 to cost✓ Set the cost to the text box provided✓ formatted to 1 or 2 decimal places✓ (Accept solutions that did not consider the leading spaces)	**10**	
1.3	**Button - [Delivery box number]** Create variable to store box number ✓ Check if speed post is selected✓ Set box number = 4 ✓ else✓ (could be another if statement) Generate random number ✓ in the correct range✓ Ensure that the value is not 4 ✓ ✓ (Only one mark if generated once more) Display box number✓	**9**	

QUESTION 1: MARKING GRID - GENERAL PROGRAMMING SKILLS (continue)

1.4	**Button - [Validate bar code]** Extract the barcode✓ Set sumOdd and sumEven to 0 ✓ Loop ✓from first to the second last digit ✓ Check if the position of the digit is even✓ Add value of the digit ✓(integer) at position to sumEven✓ Else Add value of the digit at position to sumOdd✓ Multiply sumOdd by 3✓ Add sumEven and sumOdd✓ Calculate checkdigit: Subtract total modulus 10✓ from 10 ✓ If checkdigit equals last digit ✓ (Must be same data types) Display appropriate message that bar code is valid Else ✓ Display message that the bar code is not valid (Display of the check digit not necessary for the if or else)	**14**	
1.5	**Button - [View and save deliveries]** Extract the selected name of the city from the combo box ✓ Display the name of the city in the output area as a heading ✓ Create a text file ✓with the correctly constructed name ✓ Loop from first position✓ to last position in the array ✓ Check if city name is part ✓of the correct array entry✓ Display the entry in the output area if found✓ and store the entry in the text file✓ One delivery per line✓ Close the text file outside loop✓	**12**	
	TOTAL:	**50**	

ANNEXURE B:

SECTION B:

QUESTION 2: MARKING GRID - OBJECT-ORIENTED PROGRAMMING

CENTRE NUMBER:	**EXAMINATION NUMBER:**		
QUESTION	**DESCRIPTION**	**MAX. MARKS**	**LEARNER'S MARKS**
2.1.1	**Constructor:** Heading with ONLY four values✓ Correct data types✓ Assign parameter values to four attributes✓ (default for fuelUsed can be included)	**3**	
2.1.2	**Accessor and mutator METHODs:** setFuelUsed (non return)✓ with parameter value assigned to attribute✓ getFuelUsed(return)✓ with correct return data type✓	**4**	
2.1.3	**calculateDistance METHOD:** return type a number✓ substract odoStart from odoEnd✓ (-1 if received as parameters) return answer✓	**3**	
2.1.4	**determineTollFees METHOD:** Receive route as string parameter ✓ Determine correct row in 2D depending on routeNr ✓✓ (subtract 1 mark each mistake – max 2) Determine correct column in 2D depending on TruckNr Tr1 OR ✓Tr2: 1st column✓, Tr3: 2nd column✓, Tr4 OR Tr5: 3rd column✓ Find and return tollFees value✓ at position [row] ✓ [column]✓ (-1 for incorrect order)	**10**	
2.2.1	**Button – [Get data from file]:** Receive vehicle number from combo box & convert to string✓ {*Delphi: AssignFile, Reset* *Java: Create object to read from file*} ✓✓ Test if file does not exist✓ & display message✓ Use loop counter/delivery number of last entry ✓ Increment last delivery number for new Delivery number✓ Loop through file ✓ Read line from text file✓ Find✓ and keep last occurrence of vehicle number✓ Separate information using (#) – to obtain the odometer reading, use split/copy/pos/indexOf ✓ Obtain odometer from string✓ Display values in text boxes✓	**14**	

QUESTION 2: MARKING GRID - OBJECT-ORIENTED PROGRAMMING - continue

2.2.2	**Button – [New delivery]:** Obtain newTripNumber, begin odometer, end odometer, truck number ✓ *Instantiate the object with values:* Left side of assign✓ Right side of assign, Correct number and order of parameters (4) ✓ and type casting ✓ (*or obtain from variables of correct type)* Display message after object has been created✓ Use object✓ to call calculateDistance✓ and calculate fuelUsed✓ (*divide by 5*) (Java: Ensure the fuelUsed value is a double value) Call set method to set fuelUsed attribute✓ to calculated value Enable the fuel used and toll fees buttons✓ (Buttons can also be enabled in 2.2.3)	**10**	
2.2.3	**Button – [Display delivery]:** Display information in text area✓ using toString✓	**2**	
2.2.4	**Button – [Check fuel used]:** Read fuel used from text box and convert to a real number✓ Use object and call getMethod to get fuel used✓ Calculate difference✓, calculate % ✓ Test if less than 10% difference✓ (provide for positive and negative)✓ Change fuel used using set method✓ Display message to indicate change has been made✓ Else Display error message✓	**9**	
2.2.5	**Button – [Calculate toll fees]:** Read route number from text box ✓ Use object to call determineTollFees method✓ and send route number as parameter✓ Display toll fees✓ in currency format (R##.##)✓	**5**	
	TOTAL:	**60**	

ANNEXURE C:

SECTION C:

QUESTION 3: MARKING GRID – PROBLEM SOLVING

CENTRE NUMBER:	**EXAMINATION NUMBER:**		
QUES-TION	**DESCRIPTION**	**MAX. MARKS**	**LEARNER'S MARKS**
3.1	**Button – [Load item]** Declare variable for loading code ✓ Test if fragile is selected ✓ Test if✓ fragileItems < 20 ✓ Create loading code F✓ number✓ Update fragileItems ✓ by adding a * ✓ Else ✓ (Test if non fragile is selected) Test if ✓ nonFragileItems < 30 ✓ Create loading code ✓ Update nonFragileItems by adding a * ✓ Test if there is space for the item✓ Display loading code in the text box✓ Else✓ Display a message indicating the item is not loaded✓ Display heading ✓ Display string representing fragileItems ✓ Display string representing nonFragileItems ✓	**20**	

3.2	**Button – [Check load status]**
	Determine the number of fragile items
	Determine the number of non-fragile items ✓
	(Mark given for assigning given values 4 and 13) OR
	(Mark given for using values from 3.1)
	Calculate percentage fragile items ✓
	Calculate percentage non-fragile items ✓
	Display column headings ✓and detail in columns ✓
	Display fragile item details, ✓
	formatted, percentage to 2 decimal spaces ✓
	Display non-fragile item details ✓ % formatted to 2 decimal places
	If condition
	(*percent fragile >=50✓ AND ✓percent non-fragile >=50✓*)
	Nested if acceptable
	Display message "to progress" on text area ✓
	else
	Display message "may not progress" ✓
	If (condition) ✓ (*numFragile <10*)
	Display fragileItems still required on text area, calculation✓
	If (condition) ✓ (*numNonFragile <15*)
	Display nonFragileItems still required on text area, calculation✓
	Note:The condition can be reversed displaying the corresponding messages.
	Numbers (10 and 15) can be used instead of percentages.
		17	

QUESTION 3: MARKING GRID – PROBLEM SOLVING - continue

3.3	**Button – [Clear load]** Clear fragileItems✓ Clear nonFragileItems ✓ Clear text area ✓	**3**	
	TOTAL:	**40**	

SUMMARY OF LEARNER'S MARKS:

	SECTION A	**SECTION B**	**SECTION C**	
	QUESTION 1	**QUESTION 2**	**QUESTION 3**	**GRAND TOTAL**
MAX. MARKS	**50**	**60**	**40**	**150**
LEARNER'S MARKS				

ANNEXURE D: SOLUTION FOR QUESTION 1: JAVA

```
//    A solution to Question 1

package Question1Package;

import java.io.FileNotFoundException;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.Calendar;
import java.util.Scanner;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;

public class Question1_Solution extends javax.swing.JFrame {

    int kilometres = 635;
    public Question1_Solution() {
        initComponents();
        this.setLocationRelativeTo(this);
        this.setVisible(true);
        lstKgs.setSelectedIndex(0);
        txfBarCode.setText("639382000393");
    }
=============================================================================
// Question 1.1
=============================================================================
private void btnDeliveryActionPerformed(java.awt.event.ActionEvent evt) {
        String departure = (String) (cmbDepart.getSelectedItem());
        String destination = (String) (cmbDestination.getSelectedItem());
        kilometres = Integer.parseInt(txfDistance.getText());
        lblDelivery.setText(departure + " to " + destination + " : " +
           kilometres + " km");
    }
=============================================================================
// Question 1.2
=============================================================================
private void btnDeliveryCostActionPerformed(java.awt.event.ActionEvent evt) {
        int position = (int) (lstKgs.getSelectedIndex());
        double costTransport = 0;
        switch (position) {
            case 0:
                costTransport = 0.6 * kilometres;
                break;
            case 1:
                costTransport = 1.0 * kilometres;
                break;
            case 2:
                costTransport = 1.25 * kilometres;
                break;
            case 3:
                costTransport = 1.65 * kilometres;
                break;
        }

        if (chbSpeedPost.isSelected()) {
            costTransport += 100;
        }
        txfCost.setText(String.format("R%2.2f",costTransport));
    }
```

```
================================================================================
// Question 1.3
================================================================================
   private void btnBoxNumberActionPerformed(java.awt.event.ActionEvent evt) {
        int boxNumber = 0;
        if (chbSpeedPost.isSelected()) {
            boxNumber = 4;
        } else {
            do {
                boxNumber = (int) (Math.random() * 5) + 1;
            } while (boxNumber == 4);
        }
        txfBoxNumber.setText("" + boxNumber);
    }
================================================================================
// Question 1.4
================================================================================
private void btnBarCodeActionPerformed(java.awt.event.ActionEvent evt) {
    String barCode = txfBarCode.getText();
    int sumOdd = 0;
    int sumEven = 0;
    for (int cnt = 0; cnt < barCode.length()-1; cnt ++)
    {
      if ((cnt+1) % 2  ==0)
        sumEven = sumEven + Integer.parseInt(barCode.substring(cnt, cnt + 1));
      else
         sumOdd = sumOdd + Integer.parseInt(barCode.substring(cnt, cnt + 1));
    }
    int sum = sumOdd * 3 + sumEven;
    int checkDigit = 10 - (sum % 10);
    if(checkDigit == Integer.parseInt(barCode.substring(barCode.length()-1)))
    {
        txfDisplayBarCode.setText("The bar code is valid. Check digit: " +
          checkDigit);
    }
    else
    {
        txfDisplayBarCode.setText("The bar code is NOT valid.  Correct check
          digit: " + checkDigit);
    }
}
================================================================================
// Question 1.5
================================================================================
private void btnViewDeliveriesActionPerformed(java.awt.event.ActionEvent evt) {
      String place = (String)(cmbCityName.getSelectedItem());
      outputArea.setText(place+"\n");
        try {
          PrintWriter out = new PrintWriter(new FileWriter(
                      "December2014"+place+".txt"));
           for (int i = 0;i<arrDecDeliveries.length;i++){
           if(arrDecDeliveries[i].indexOf(place) >=0){
               outputArea.append(arrDecDeliveries[i]+"\n");
               out.println(arrDecDeliveries[i]);
           }
       }
           out.close();
      } catch (IOException e) {
          JOptionPane.showMessageDialog(null,"Error");
      }
    }
```

ANNEXURE E: SOLUTION FOR QUESTION 2: JAVA

```
//    A solution to Question 2
```

OBJECT CLASS: DELIVERY (GIVEN)

```
public class Delivery {

//=============================================================================
//This code is given in the program
//=============================================================================
    private int deliveryNum;
    private String truckNum;
    private double fuelUsed;
    private int odoStart;
    private int odoEnd;
    double[][] tollFees = {{105.50, 135.00, 210.00},
                           {35.00, 54.00, 82.00},
                           {85.00, 129.00, 205.00},
                           {112.00, 170.00, 219.00}};


public String toString() {
        DecimalFormat df = new DecimalFormat("0.0");
        String output = "Delivery number: " + deliveryNum + "\nTruck number: "
                      + truckNum + "\nOdometer reading: \n\t(Start) " +
                      odoStart + "\n\t(End) " + odoEnd + "\nFuel used: " +
                      df.format(fuelUsed) + " litres";
        return output;
    }

=============================================================================
// Question 2.1.1
=============================================================================

public Delivery(int deliveryNum, String truckNum, int odoStart, int
                odoEnd) {
        this.deliveryNum = deliveryNum;
        this.truckNum = truckNum;
        this.odoStart = odoStart;
        this.odoEnd = odoEnd;
    }

=============================================================================
// Question 2.1.2
=============================================================================
//Accessor method

public double getFuelUsed() {
        return fuelUsed;
    }

//Mutator method

    public void setFuelUsed(double fuel) {
        fuelUsed = fuel;
    }
```

```
//============================================================================
// Question 2.1.3
//============================================================================

 public int calculateDistance()
    {
        return odoEnd - odoStart;
    }


//============================================================================
// Question 2.1.4
//============================================================================

public double determineTollFees(String route) {
        double tollAmount = 0;
        int row = Integer.parseInt(route.substring(2, 3)) - 1;

        if (truckNum.equals("Tr1") || truckNum.equals("Tr2")) {
            tollAmount = tollFees[row][0];
        } else if (truckNum.equals("Tr3")) {
            tollAmount = tollFees[row][1];
        } else {
            tollAmount = tollFees[row][2];
        }
        /* Alternative:
        switch (truckNum) {
            case "Tr1":
            case "Tr2":
                tollAmount = tollFees[row][0];
                break;
            case "Tr3":
                tollAmount = tollFees[row][1];
                break;
            default:
                tollAmount = tollFees[row][2];
                break;
        }
        */
        return tollAmount;
    }
```

GUI CLASS: QUESTION2_SOLUTION

```
package Question2Package;

import java.io.File;
import java.io.FileReader;
import java.text.DecimalFormat;
import java.util.Scanner;
import javax.swing.JOptionPane;


public class Question2_Solution extends javax.swing.JFrame {
=============================================================================
// Given code
=============================================================================
Delivery objDelivery;

public Question2_Solution() {
        initComponents();
        this.setLocationRelativeTo(this);
        this.setVisible(true);
        btnTollFees.setEnabled(false);
        btnFuelChange.setEnabled(false);
    }


// Code not copied for graphics
=============================================================================
// Question 2.2.1
=============================================================================
private void btnGetFromFileActionPerformed(java.awt.event.ActionEvent evt) {
        File file = new File("DeliveryInfo.txt");
        if (!file.exists()) {
            JOptionPane.showMessageDialog(rootPane, "File does not exists");
            System.exit(0);
        } else {

            String truckNr = (String) cmbVehicleNumber.getSelectedItem();
            try {
                String lastTruckLine = "", line = "";
                Scanner sc = new Scanner(new FileReader("DeliveryInfo.txt"));
                String[] temp;
                while (sc.hasNext()) {
                    line = sc.next();
                    if (line.contains(truckNr)) {
                        lastTruckLine = line;
                    }
                }
                temp = line.split("#");
                int newTrip = Integer.parseInt(temp[0]) + 1;
                txfNewTripNum.setText("" + newTrip);

                temp = lastTruckLine.split("#");
                txfStartOdometer.setText(temp[2]);
            }
            catch (Exception e) {
            }
        }
    }
```

```
============================================================================
// Question 2.2.2
============================================================================

private void btnNewDeliveryActionPerformed(java.awt.event.ActionEvent evt) {
   int newTripNum = Integer.parseInt(txfNewTripNum.getText());
   int startOdoReading = Integer.parseInt(txfStartOdometer.getText());
   int endOdoReading = Integer.parseInt(txfEndOdometer.getText());
   String truckNr = (String) cmbVehicleNumber.getSelectedItem();
   objDelivery = new Delivery(newTripNum, truckNr, startOdoReading,
              endOdoReading);
   JOptionPane.showMessageDialog(rootPane, "Delivery object created
               successfully.");
   int distance = objDelivery.calculateDistance();
   objDelivery.setFuelUsed(distance / 5.0);
   btnTollFees.setEnabled(true);
   btnFuelChange.setEnabled(true);
}

============================================================================
// Question 2.2.3
============================================================================
private void btnDisplayDeliveryActionPerformed(java.awt.event.ActionEvent evt)
{
        txaOptionA.setText(objDelivery.toString());
    }

============================================================================
// Question 2.2.4
============================================================================
private void btnFuelChangeActionPerformed(java.awt.event.ActionEvent evt) {
    double fuelAdded = Double.parseDouble(txfFuel.getText());
    double fuelUsed = objDelivery.getFuelUsed();
    if (Math.abs(fuelAdded - fuelUsed) / fuelUsed < 0.1) {
        objDelivery.setFuelUsed(fuelAdded);
        txfFuelMessage.setText("Fuel changed from " + fuelUsed + " to " +
          fuelAdded + " litres");
    } else {
        txfFuelMessage.setText("ERROR: Difference in fuel used is too great");
    }
}

============================================================================
// Question 2.2.5
============================================================================
private void btnTollFeesActionPerformed(java.awt.event.ActionEvent evt) {
        String output = String.format("%-23sR%2.2f", "Toll fees to be paid:",
        objDelivery.determineTollFees(txfRoute.getText()));
        lblTollFees.setText(output);
    }
```

ANNEXURE F: SOLUTION FOR QUESTION 3: JAVA

```
//    A possible solution to Question 3

package Question3Package;

import javax.swing.JOptionPane;

public class Q3 extends javax.swing.JFrame {

//Global variables
   String fragileItems = "";
   String nonFragileItems = "";

//=========================================================================
//This code is given in the program
//=========================================================================

public Question3_Solution() {
        initComponents();
        this.setLocationRelativeTo(this);
        this.setVisible(true);
        rbtFragile.setSelected(true);
    }
=========================================================================
// Question 3.1
=========================================================================
private void btnLoadActionPerformed(java.awt.event.ActionEvent evt) {
    String loadingCode = "";
    if (rbtFragile.isSelected()) {
       if (fragileItems.length() < 20) {
          loadingCode = "F" + (fragileItems.length() + 1);
          fragileItems += "*";
       }
    } else {
      if (nonFragileItems.length() < 30) {
          loadingCode = "NF" + (nonFragileItems.length() + 1);
          nonFragileItems += "*";
      }
    }
    txfLoadingCode.setText(loadingCode);
    if (loadingCode.equals("")) {
            JOptionPane.showMessageDialog(null, "Loading of item cannot be
             processed - No loading space\n", "Information", WIDTH);
    }
    txaOutput.setText("Loading progress display area:
                      \n============================\n\n");
    txaOutput.append(String.format("%-20s%-25s%n", "Fragile items:",
                     fragileItems));
    txaOutput.append(String.format("%-20s%-25s", "Non-fragile items:",
                     nonFragileItems));
}
```

```
=============================================================================
// Question 3.2
=============================================================================

private void btnStatusActionPerformed(java.awt.event.ActionEvent evt) {
     int numFragile = fragileItems.length();
     int numNonFragile = nonFragileItems.length();
     double percFragile = (numFragile) / 20.0 * 100;
     double percNonFragile = (numNonFragile) / 30.0 * 100;
     txaOutput.setText(" Load status report:\n
                         ====================\n");
     txaOutput.append(String.format("%-15s%-25s%-15s%n", " Item type",
                      "Number of items", "Percentage loaded"));
     txaOutput.append(String.format("%-15s%-25s%-13.2f%n", " Fragile",
                      numFragile, percFragile));
     txaOutput.append(String.format("%-15s%-25s%-13.2f%n", " Non-fragile",
                      numNonFragile, percNonFragile));
     if (percFragile >= 50 && percNonFragile >= 50) {
         txaOutput.append("\n The delivery may progress.");
     }
     if (numFragile < 10 || numNonFragile < 15) {
         txaOutput.append("\n The delivery may not progress.");
         if (numFragile < 10) {
           txaOutput.append("\n Number of fragile items still required : "
                 + (10 - numFragile));
         }
         if (numNonFragile < 15) {
             txaOutput.append("\n Number of non-fragile items still
             required : " + (15 - numNonFragile));
         }
     }
 }
=============================================================================
// Question 3.3
=============================================================================
private void btnClearActionPerformed(java.awt.event.ActionEvent evt) {
        fragileItems="";
        nonFragileItems="";
        txaOutput.setText("");
}
```

ANNEXURE G: SOLUTION FOR QUESTION 1: DELPHI

```
unit Question1_U_Memo;

interface
    //Possible solution for Question 1


uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
Forms,
  Dialogs, StdCtrls, ExtCtrls, ComCtrls, StrUtils;

type
  TfrmQuestion1 = class(TForm)
    grpQ1_1: TGroupBox;
    grpQ1_3: TGroupBox;
    grpQ1_2: TGroupBox;
    grpQ1_4: TGroupBox;
    lblDeliveryFrom: TLabel;
    lblDeliveryTo: TLabel;
    lblNoKM: TLabel;
    edtKm: TEdit;
    btnDeliveryConfirmation: TButton;
    grpDLabel: TGroupBox;
    lblDeliveryCode: TLabel;
    grpSpeedpost: TGroupBox;
    btnDeliveryCost: TButton;
    grpRange: TGroupBox;
    edtDeliveryCost: TEdit;
    lstRangeKM: TListBox;
    btnDeliveryBoxNumber: TButton;
    edtDeliveryBoxNumber: TEdit;
    btnCreateBarCode: TButton;
    edtCreateBarCode: TEdit;
    cboDeliveryFrom: TComboBox;
    cboDeliveryTo: TComboBox;
    lblUPCBarCode: TLabel;
    edtUPCBarCode: TEdit;
    chkSpeedPost: TCheckBox;
    grpQ1_5: TGroupBox;
    cboCityName: TComboBox;
    btnViewDeliveries: TButton;
    redOutputArea: TRichEdit;
    lblCity: TLabel;
    procedure btnDeliveryConfirmationClick(Sender: TObject);
    procedure btnDeliveryCostClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure btnDeliveryBoxNumberClick(Sender: TObject);
    procedure btnCreateBarCodeClick(Sender: TObject);
    procedure btnViewDeliveriesClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmQuestion1: TfrmQuestion1;
  iKilometres : Integer = 635; //default value
```

```
//given do not change
  arrDecDeliveries : array[1..32] of String =
  ('2013-12-01 Durban to Cape Town',
  '2013-12-01 Polokwane to Johannesburg',
  '2014-12-02 Cape Town to Johannesburg ',
  '2014-12-02 Polokwane to Potchefstroom ',
  '2014-12-02 Bloemfontein to Port Elizabeth',
  '2013-12-03 Polokwane to Potchefstroom',
  '2014-12-03 Cape Town to Port Elizabeth ',
  '2014-12-03 Port Elizabeth to Potchefstroom ',
  '2014-12-04 Port Elizabeth to Durban',
  '2013-12-04 Polokwane to Kimberley',
  '2014-12-04 Cape Town to Kimberley ',
  '2014-12-04 Polokwane to Potchefstroom ',
  '2014-12-04 Kimberley to Port Elizabeth',
  '2014-12-05 Durban to Kimberley',
  '2014-12-05 Bloemfontein to Potchefstroom',
  '2014-12-05 Durban to Potchefstroom',
  '2013-12-05 Cape Town to Potchefstroom',
  '2013-12-05 Polokwane to Cape Town',
  '2014-12-06 Cape Town to Johannesburg ',
  '2014-12-06 Polokwane to Potchefstroom ',
  '2014-12-06 Bloemfontein to Kimberley',
  '2013-12-06 Polokwane to Johannesburg',
  '2014-12-07 Cape Town to Port Elizabeth ',
  '2014-12-07 Port Elizabeth to Potchefstroom ',
  '2014-12-07 Potchefstroom to Durban',
  '2013-12-07 Cape Town to Kimberley',
  '2014-12-08 Cape Town to Kimberley ',
  '2014-12-08 Polokwane to Potchefstroom ',
  '2014-12-08 Kimberley to Port Elizabeth',
  '2014-12-08 Potchefstroom to Kimberley',
  '2014-12-09 Bloemfontein to Polokwane',
  '2014-12-09 Durban to Bloemfontein');

implementation

{$R *.dfm}
procedure TfrmQuestion1.btnDeliveryConfirmationClick(Sender: TObject);
begin
=============================================================================
//Question 1.1
=============================================================================
  ikilometres := StrToInt(edtKm.Text);
  lblDeliveryCode.Caption :=
        cboDeliveryFrom.Items[cboDeliveryFrom.ItemIndex] + ' to ' +
            cboDeliveryTo.Items[cboDeliveryTo.ItemIndex] + ' : ' +
        edtKm.Text + 'km';
end;

procedure TfrmQuestion1.btnDeliveryCostClick(Sender: TObject);
var
  iPositon       : integer;
  rCostTransport : real;
begin
=============================================================================
//Question 1.2
=============================================================================
iPositon := lstRangeKM.ItemIndex;
   rCostTransport := 0;
   Case (iPositon) of
```

```
     0: rCostTransport := 0.60 * ikilometres;
     1: rCostTransport := 1.00 * ikilometres;
     2: rCostTransport := 1.25 * ikilometres;
     3: rCostTransport := 1.65 * ikilometres;
   end;

   if  chkSpeedPost.Checked = True
   then
     begin
       rCostTransport := rCostTransport + 100;
     end;
   edtDeliveryCost.Text := FloatToStrF(rCostTransport, ffCurrency, 8,2);
end;

procedure TfrmQuestion1.btnDeliveryBoxNumberClick(Sender: TObject);
var
  iBoxNumber : integer;
begin
=============================================================================
//Question 1.3
=============================================================================
  if chkSpeedPost.Checked = true
  then
    begin
      iBoxNumber := 4;
    end
  else
    begin
     //generate a random number between 1 to 5 which is not 4
     repeat
       iBoxNumber := random(5)+1;
     until iBoxNumber <> 4;
    end;
  edtDeliveryBoxNumber.Text := IntToStr(iBoxNumber);
end;

procedure TfrmQuestion1.btnCreateBarCodeClick(Sender: TObject);
var
  sBarCode : string;
  iSumOdd, iSumEven, iCounter, iTotal, iCheckDigit : Integer;
begin
=============================================================================
//Question 1.4
=============================================================================
  sBarCode := edtUPCBarCode.Text;
  iSumOdd := 0;
  iSumEven := 0;
  for iCounter := 1 to Length(sBarCode)-1 do
   begin
      if (iCounter MOD 2) = 0
       then inc(iSumEven, StrToInt(sBarCode[iCounter]))
       else inc(iSumOdd,  StrToInt(sBarCode[iCounter]));
   end;
  iTotal := (iSumOdd * 3) + iSumEven;
  iCheckDigit := 10 - (iTotal mod 10);
  if iCheckDigit = StrToInt(sBarCode[Length(sBarCode)])
   then
    edtCreateBarCode.Text := 'The bar code is valid. ' +
                             'Check digit: ' + IntToStr(iCheckDigit)
   else
    edtCreateBarCode.Text := 'The bar code is NOT valid. ' +
```

```
                                ' Correct check digit: ' +
    IntToStr(iCheckDigit) ;
end;

procedure TfrmQuestion1.btnViewDeliveriesClick(Sender: TObject);
var
  sCity, sFileName    : string;
  txtFile   : TextFile;
  iCounter  : Integer;
begin
=============================================================================
//Question 1.5
=============================================================================
  redOutputArea.Clear;
  sCity := cboCityName.Items[cboCityName.ItemIndex];
  redOutputArea.Lines.Add(sCity);
  sFileName := 'December2014'+sCity + '.txt';
  AssignFile(txtFile, sFileName);
  Rewrite(txtFile);
  for iCounter := 1 to  32 do
   begin
     if pos(sCity, arrDecDeliveries[iCounter]) > 0
     then
      begin
          redOutputArea.Lines.Add(arrDecDeliveries[iCounter]);
          Writeln(txtFile, arrDecDeliveries[iCounter]);
      end;
     end;
  CloseFile(txtFile);
end;

procedure TfrmQuestion1.FormCreate(Sender: TObject);
begin
  lstRangeKM.Selected[0] := True;
  CurrencyString := 'R';
  Randomize;
end;

end.
```

ANNEXURE H: SOLUTION FOR QUESTION 2: DELPHI

CLASS UNIT: DELIVERY_U.PAS

```
unit Delivery_U;
   //Possible solution for Question 2 - class unit.

interface

uses
  sysUtils;

  type
    TDelivery = class(TObject)
    private
      fDeliveryNum : integer;
      fTruckNum    : string;
      fFuelUsed    : real;
      fOdoStart    : integer;
      fOdoEnd      : integer;
    public
      function toString: string;
      constructor Create(iDeliverNumber: integer; sTruckNumber : string;
                                          iOdoStart, iOdoEnd : integer);
      function getFuelUsed: real;
      procedure setFuelUsed (rFuelUsed : Real);
      function calculateDistance: Integer;
      function determineTollFees(sRoute : string): real;
  end;

var
//=========================================================================
//Given to be used in question 2.1.4
//=========================================================================
  tollFees : array[1..4,1..3] of real =
    ((105.50, 135.00, 210.00), (35.00, 54.00, 82.00),
     (85.00,129.00,205.00), (112.00, 170.00, 219.00));

implementation
//=========================================================================
// Question 2.1.1.
//=========================================================================
constructor TDelivery.Create(iDeliverNumber: integer; sTruckNumber: string;
  iOdoStart, iOdoEnd: integer);
begin
  fDeliveryNum := iDeliverNumber;
  fTruckNum    := sTruckNumber;
  fOdoStart    := iOdoStart;
  fOdoEnd      := iOdoEnd;
end;
//=========================================================================
// Question 2.1.2.
//=========================================================================
function TDelivery.getFuelUsed: real;
begin
  Result := fFuelUsed;
end;
```

```
procedure TDelivery.setFuelUsed(rFuelUsed: Real);
begin
  fFuelUsed := rFuelUsed;
end;
=============================================================================
// Question 2.1.3.
=============================================================================
function TDelivery.calculateDistance: Integer;
begin
  Result := fOdoEnd - fOdoStart;
end;
=============================================================================
// Question 2.1.4.
=============================================================================
function TDelivery.determineTollFees(sRoute: string): real;
var
   iRow : integer;
begin
  Result := 0;
  iRow := StrToInt(sRoute[3]);//3rd character
  if (fTruckNum = 'Tr1') OR (fTruckNum = 'Tr2')
   then Result := tollFees[iRow, 1]
   else if (fTruckNum = 'Tr3')
         then Result := tollFees[iRow, 2]
         else Result := tollFees[iRow, 3];

{Alternative:
  case fTruckNum[3] of
    '1', '2' : Result := tollFees[iRow, 1];
    '3'      : Result := tollFees[iRow, 2];
    '4', '5' : Result := tollFees[iRow, 3];
  end;//case
}
end;

function TDelivery.toString: string;
begin
   Result :=  'Delivery Number: ' + IntToStr(fDeliveryNum) + #13 +
              'Truck number: '+ fTruckNum + #13 +
              'Odometer reading: '+#13+
               #9 + '(Start) ' + IntToStr(fOdoStart) + #13 +
               #9 + '(End)   ' +  IntToStr(fOdoEnd) + #13 +
               'Fuel used: ' + FloatToStr(fFuelUsed) + ' litres';
end;

end.
```

MAIN FORM UNIT: QUESTION2_U.PAS

```
unit Question2_U_Memo;
//Possible solution for Question 2 - Formunit.

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
Forms,
  Dialogs, ExtCtrls, StdCtrls, Spin, Delivery_U, ComCtrls;

type
  TfrmQuestion2 = class(TForm)
    pnlTitle: TPanel;
    grpCreateDisplay: TGroupBox;
    GroupBoxOptionC: TGroupBox;
    lblVehicleNumber: TLabel;
    cboVehicleNumber: TComboBox;
    redOutput: TRichEdit;
    lblDistanceTravelled: TLabel;
    edtEndOdometer: TEdit;
    btnCreateNewDelivery: TButton;
    btnShowDelivery: TButton;
    lblActualFuelUsed: TLabel;
    edtFuelUsed: TEdit;
    btnFuelChange: TButton;
    edtFuelMessage: TEdit;
    grpTollFees: TGroupBox;
    lblRoute: TLabel;
    edtRoute: TEdit;
    btnTollFee: TButton;
    pnlTollFees: TPanel;
    lblStartOdoReading: TLabel;
    edtStartOdometer: TEdit;
    lblNewTripNum: TLabel;
    edtNewTripNum: TEdit;
    btnGetFromFile: TButton;
    lblTollFees: TLabel;

    procedure FormCreate(Sender: TObject);
    procedure btnTollFeeClick(Sender: TObject);
    procedure btnCreateNewDeliveryClick(Sender: TObject);
    procedure btnShowDeliveryClick(Sender: TObject);
    procedure btnFuelChangeClick(Sender: TObject);
    procedure btnGetFromFileClick(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmQuestion2: TfrmQuestion2;

  Delivery : TDelivery;

implementation
```

```
{$R *.dfm}
{$R+}

procedure TfrmQuestion2.FormCreate(Sender: TObject);
begin
  CurrencyString := 'R';
end;

procedure TfrmQuestion2.btnGetFromFileClick(Sender: TObject);
var
  txtFile : TextFile;
  sLine, sTripNo, sTruckNumber, sStartOdo : string;
  iNewTrip : Integer;
begin

=============================================================================
// Question 2.2.1
=============================================================================
  if NOT FileExists('DeliveryInfo.txt')
   then
    begin
      MessageDlg('DeliveryInfo.txt does not exist', mtError, [mbOK], 0);
      Exit;
    end;

  sTruckNumber := cboVehicleNumber.Items[cboVehicleNumber.ItemIndex];

  AssignFile(txtFile,'DeliveryInfo.txt');
  Reset(txtFile);
  while not EOF(txtFile) do
  begin
    readln(txtFile,sline);
    sTripNo := copy(sline, 1, pos('#',sline)-1);
    if Pos(sTruckNumber, sLine) > 0
     then
      begin
         Delete(sline, 1, pos('#',sline));
         Delete(sline,1,pos('#',sline));
         sStartOdo := sLine;
      end;//if
  end;
  closeFile(txtFile);

  iNewTrip := StrToInt(sTripNo) + 1;
  edtNewTripNum.Text := IntToStr(iNewTrip);
  edtStartOdometer.Text := sStartOdo;
end;

procedure TfrmQuestion2.btnCreateNewDeliveryClick(Sender: TObject);
var
   iDistance, iNewTripNum, iStartOdoReading, iEndOdoReading : integer;
   sTruckNumber : string;
begin
=============================================================================
// Question 2.2.2
=============================================================================
  sTruckNumber := cboVehicleNumber.Items[cboVehicleNumber.ItemIndex];
  iNewTripNum := StrToInt(edtNewTripNum.text);
  iStartOdoReading := StrToInt(edtStartOdometer.text);
  iEndOdoReading := StrToInt(edtEndOdometer.text);
```

```
  Delivery := TDelivery.Create(iNewTripNum, sTruckNumber, iStartOdoReading,
              iEndOdoReading);
  MessageDlg('Delivery object created successfully.', mtInformation,
              [mbOK],0);

  iDistance := Delivery.calculateDistance;
  Delivery.setFuelUsed(iDistance / 5.0);

  btnTollFee.Enabled := True;
  btnFuelChange.Enabled := True;
end;

procedure TfrmQuestion2.btnShowDeliveryClick(Sender: TObject);
begin
=============================================================================
// Question 2.2.3
=============================================================================
  redOutput.Clear;
  redOutput.Lines.Add(Delivery.toString);
end;

procedure TfrmQuestion2.btnFuelChangeClick(Sender: TObject);
var
    rFuelAdded, rFuelUsed : real;
begin
=============================================================================
// Question 2.2.4
=============================================================================
  rFuelAdded := StrToFloat( edtFuelUsed.Text);
  rFuelUsed :=  delivery.getFuelUsed;

  if (Abs(rFuelAdded - rFuelUsed) / rFuelUsed) < 0.1
   then
    begin
      Delivery.setFuelUsed(rFuelAdded);
      edtFuelMessage.Text := 'Fuel used changed from ' +
             FloatToStrF(rFuelUsed, ffFixed, 12, 1) + ' to '+
             FloattoStrF(rFuelAdded, ffFixed, 12, 1) + ' litres';
    end
   else
     edtFuelMessage.Text := 'ERROR: : Difference in fuel used is too great';
end;

procedure TfrmQuestion2.btnTollFeeClick(Sender: TObject);
var
  sRouteNum : string;
begin
=============================================================================
// Question 2.2.5.
=============================================================================
  sRouteNum := edtRoute.Text;
  lblTollFees.Caption :=  'Toll fees to be paid: ' +
  FloatToStrF(Delivery.determineTollFees(sRouteNum), ffCurrency, 8, 2);
end;

end.
```

ANNEXURE I: SOLUTION FOR QUESTION 3: DELPHI

```
unit Question3_U_Memo;
//Possible solution for Question 3.

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
Forms,
  Dialogs, ExtCtrls, StdCtrls, ComCtrls;

type
  TfrmQuestion3 = class(TForm)
    grpLoadingZone: TGroupBox;
    btnClearLoad: TButton;
    btnLoadItem: TButton;
    btnCheckLoadingStatus: TButton;
    rgpItemType: TRadioGroup;
    redQ3: TRichEdit;
    edtLoadingCode: TEdit;
    lblLoadingCode: TLabel;
    procedure btnClearLoadClick(Sender: TObject);
    procedure btnLoadItemClick(Sender: TObject);
    procedure btnCheckLoadingStatusClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmQuestion3: TfrmQuestion3;

  sFragileItems    : string;
  sNonFragileItems : string;

implementation
{$R *.dfm}
{$R+}

procedure TfrmQuestion3.btnLoadItemClick(Sender: TObject);
var
  sLoadingCode : string;
begin
=============================================================================
// Question 3.1
=============================================================================
  case rgpItemType.ItemIndex of
    0 : begin
          if length(sFragileItems) < 20
           then
             begin
               sLoadingCode := 'F' + IntToStr(Length(sFragileItems)+1);
               sFragileItems :=  sFragileItems + '*';
             end
           else sLoadingCode := '';
        end;//fragile
```

```
    1 : begin
        if length(sNonFragileItems) < 30
         then
           begin
              sLoadingCode := 'NF' + IntToStr(Length(sNonFragileItems)+1);
              sNonFragileItems := sNonFragileItems + '*';
           end
         else sLoadingCode := '';
      end;//non-fragile
  end;//case

  if sLoadingCode = ''
   then
    begin
      edtLoadingCode.Clear;
      MessageDlg('Loading of item cannot be processed - No loading space',
                          mtInformation, [mbok], 0);
    end//if no space
   else
    begin
      edtLoadingCode.Text := sLoadingCode;
      redQ3.Clear;
      redQ3.Paragraph.TabCount := 1;
      redQ3.Paragraph.Tab[0]   := 150;
      redQ3.Lines.Add('Loading progress display area:');
      redQ3.Lines.Add('==============================');
      redQ3.Lines.Add(' ');
      redQ3.Lines.Add('Fragile items:' + #9 + sFragileItems);
      redQ3.Lines.Add('Non-fragile items:' +#9 + sNonFragileItems);
    end;//space available
end;

procedure TfrmQuestion3.btnCheckLoadingStatusClick(Sender: TObject);
var
  iNumFragile, iNumNonFragile  : integer;
  rPecFragile, rPercNonFragile : real;
begin
=============================================================================
// Question 3.2
=============================================================================
  iNumFragile    := Length(sFragileItems);
  iNumNonFragile := Length(sNonFragileItems);
  rPecFragile     := iNumFragile / 20 * 100;
  rPercNonFragile := iNumNonFragile / 30 * 100;
  redQ3.Clear;
  redQ3.Paragraph.TabCount := 2;
  redQ3.Paragraph.Tab[0]   := 100;
  redQ3.Paragraph.Tab[1]   := 275;
  redQ3.Lines.Add('Load status report:');
  redQ3.Lines.Add('==================');
  redQ3.Lines.Add(' ');
  redQ3.Lines.Add('Item type' + #9 + 'Number of items' + #9 + 'Percentage
                   loaded');
  redQ3.Lines.Add('Fragile' + #9 + IntToStr(iNumFragile) + #9 +
                            FloatToStrF(rPecFragile, ffFixed, 8,2));
  redQ3.Lines.Add('Non-fragile' + #9 + IntToStr(iNumNonFragile) + #9 +
                            FloatToStrF(rPercNonFragile, ffFixed, 8,2));
  redQ3.Lines.Add(' ');
```

```
if (rPecFragile >= 50) and (rPercNonFragile >= 50)
   then
    begin
      redQ3.Lines.Add('The delivery may progress.');
    end
   else
    begin
      redQ3.Lines.Add('The delivery may not progress.');
      if(iNumFragile <= 10 )
       then redQ3.Lines.Add('Number of fragile items still required: ' +
                                  IntToStr(10 - iNumFragile) );

      if (iNumNonFragile <= 15 )
       then redQ3.Lines.Add('Number of non-fragile items still required: '
          + IntToStr(15 - iNumNonFragile));
    end;
end;

procedure TfrmQuestion3.btnClearLoadClick(Sender: TObject);
begin
=============================================================================
// Question 3.3
=============================================================================
  sFragileItems := '';
  sNonFragileItems := '';
  redQ3.Clear;
end;

procedure TfrmQuestion3.FormCreate(Sender: TObject);
begin
   CurrencyString := 'R';
end;

end.
```