basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

SENIOR CERTIFICATE EXAMINATIONS

INFORMATION TECHNOLOGY P1

2017

MARKING GUIDELINES

MARKS: 150

These marking guidelines consist of 30 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.

- It is acknowledged that there may be different views about some matters of emphasis or detail in the guidelines, and different interpretations of the application thereof.

- Note that candidates 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 were not followed or the requirements of the question were 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 9–17) contain examples of solutions for Java for Questions 1 to 3 in programming code.

- **Annexures G, H and I** (pages 18–30) 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 candidate 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**	**CANDIDATE MARKS**

1.1	**Button - [Question 1.1]**
	
	Extract age from text box, convert to integer✓
	If the passport check box is ticked ✓
	If age < 16 ✓and the adult check box is ticked ✓
	Display 'Boarding confirmed ' ✓
	Else
	Display 'Boarding not confirmed' ✓
	else
	Display 'Boarding is not confirmed' ✓
		7	

1.2	**Button - [Question 1.2]** Extract weight of luggage from text box, convert to double ✓ Obtain the line with the airline and maximum luggage capacity from list box ✓ // Extract the maximum weight from the line Find the position of the hash character Find the number of characters to copy from the line as weight using length or any other function ✓ Copy the weight part of the line ✓ Convert the weight part to a double number ✓ If weight > maximum weight ✓ Calculate excess weight ✓ Calculate cost of excess weight ✓ Display the excess weight ✓ Display the cost of excess weight ✓	**10**	
1.3	**Button - [Question 1.3]** Extract number of passengers from text box, convert to integer ✓ Calculate the number of vegetarian meals to prepare ✓ Calculate the number of non-vegetarian meals to prepare ✓ Display the number of vegetarian meals ✓ Display the number of non-vegetarian meals ✓	**5**	

1.4	**Button - [Question 1.4]**
	
	Check if the length of the depart time = 5 characters ✓
	Check if the 'h' is the third character in string ✓
	If (length of depart time = 5) AND (third character is 'h')
	Extract hour from input as integer ✓
	Extract minute from input as integer ✓
	If (hour <= 23) ✓AND (minute <= 59) ✓
	if minute >= 35 ✓
	minute = minute - 35 ✓
	else
	hour = hour - 1✓
	minute = 60 + minute – 35 ✓✓
	
	If minute < 10 then ✓
	Boarding time = hour + 'h' + '0' + minute ✓
	else
	Boarding time = hour + 'h' + minute ✓
	
	Display the boarding time ✓
	Else
	Display message 'Invalid time entered'
	Clear the depart time text box ✓
	Else
	Display message 'Invalid time entered' ✓
	Clear the depart time text box
		17	

1.5	**Button - [Question 1.5]** Extract distance from text box as a real value✓ Check index / item of radio button selected ✓ Set flyCardStatus to 'Nonmember' Set bonus points to 0 If Silver is selected, Set flyCardStatus to 'Silver' ✓ If Gold is selected, Set flyCardStatus to 'Gold' Set bonus points to 15% of distance ✓ If Platinum is selected, Set flyCardStatus to 'Platinum' Set bonus points to 20% of distance ✓ Set filename to flyCardStatus + '.jpg' ✓ Load the file onto the image component✓ If a card was selected, Show panel named pnlPoints ✓ Calculate points Distance / 1.6 ✓ + bonus points and rounding down✓ Display points on the panel ✓	**11**	
	TOTAL:	**50**	

ANNEXURE B:

SECTION B:

QUESTION 2: MARKING GRID - OBJECT-ORIENTED PROGRAMMING

CENTRE NUMBER:	**EXAMINATION NUMBER:**		
QUESTION	**DESCRIPTION**	**MAX. MARKS**	**CANDIDATE MARKS**
2.1.1	**Constructor:** Heading ✓with three parameters ✓ Correct data types ✓ Assign parameter values to three attributes ✓ Call the setNumPassengers method ✓	**5**	
2.1.2	**Accessor methods:** getFlightNumber ✓with correct return data type ✓ getNumPassengers✓with correct return data type ✓	**4**	
2.1.3	**increasePassengers method:** Heading increasePassengers ✓ Increase the numPassengers attribute by 1 ✓	**2**	
2.1.4	**calcPercBooked method:** Heading: return double, ✓ integer parameter ✓ Calculate percentage: passengers booked/maximum passengers * 100 ✓ return percentage ✓	**4**	
2.1.5	**toString method:** Heading toString ✓ Method definition ✓ Format ✓ and correct attributes ✓ (-1 for each incorrect attribute or method called – maximum 2 marks)) Correct return statement ✓	**5**	

(QUESTION 2.2 on the next page.)

QUESTION 2: MARKING GRID – continue

2.2.1	**Button – [Question 2.2.1]** Extract flight details from combo box ✓ Use the flight details to do the following: Determine the position of the hash character ✓ Copy flight number into a variable ✓ Determine the position of the next hash character ✓ Copy name of city into a variable ✓ Copy date into a variable ✓✓ Instantiate the flight object ✓ using the correct arguments ✓ Enable button btnQuestion222✓ Enable button btnQuestion223✓ Display a message for object instantiated ✓	**12**	
2.2.2	**Button – [Question 2.2.2]** Call the setNumPassengers method of the object ✓ {*Delphi: AssignFile, Reset* *Java: Create object to read from file*} ✓✓ If file does not exist ✓ display message✓, close program ✓ Loop through file ✓ Read line from text file✓ Check if flight number of object is a part of the line✓✓ Call the increasePassengers method ✓ Add the line to the output area of the GUI✓ End Loop Display a blank line ✓ Display details of the object using the toString method ✓ Close the file ✓	**15**	

2.2.3	**Button – [Question 2.2.3]**
	{*Delphi: AssignFile, Append*
	Java: Create object to add to file} ✓✓
	Use dialog to enter maximum number of passengers✓
	Call calcPercBooked method ✓
	If percentage >= 100 display message 'Fully booked'✓
	else
	Use dialog to display percentage booked✓
	Use dialog to enter new name ✓
	//Compile the passenger reference number
	Flight number + '-' ✓+ (numPassengers + 1) ✓
	
	Write the reference number+ ' '+ passenger name ✓to text file ✓
	
	Close the file ✓ Call button Question222 to display the passenger details and flight object ✓
		13	

	TOTAL:	**60**	

ANNEXURE C:

SECTION C:

QUESTION 3: MARKING GRID – PROBLEM-SOLVING

CENTRE NUMBER:	**EXAMINATION NUMBER:**		
QUESTION	**DESCRIPTION**	**MAX. MARKS**	**CANDIDATE MARKS**
3.1	**Button – [3.1 - Display queues]** Use the number of elements in the arrPassengers array ✓ to determine number of check-in counters (columns) as follows: From 1 to 9 passengers: counter 1 ✓ From 10 to 16 passengers: counter 2 ✓ From 17 to 24 passengers: counter 3 More than 24 passengers: counter 4 ✓ (4) **Determine number of rows:** Number of passengers/number of counters ✓ Rounded up ✓ (2) **Sort the passengers in array according to class:** Outer loop ✓; Inner loop ✓; Correct test ✓; Swap elements of array correctly✓✓ (5) **Copy passengers' information to 2D/Grid:** Initialise counter to test if end of array reached ✓ Loop through rows ✓ (for) Initialise counter to 0 ✓ Loop through columns✓ and testing end of array not reached ✓ (while) Copy from passenger array to 2D/Grid ✓ using correct indexes ✓ Updating counter ✓ (8) **Display passengers in queues:** Headings: Loop correct number of times✓ Display headings in columns✓ Data: Loop through rows ✓ Loop through columns ✓ Display passenger ✓ class and number ✓ In correct number of columns and correct number of rows ✓ (7)	**26**	

3.2	**Button – [3.2 – Create new list]** Read flight number of delayed flight from combo box ✓ Display heading containing flight number ✓ Loop through array ✓ Test if flight number = delayed flight number ✓ Increment numPassengers on delayed flight ✓ Display at new counter ✓ (6) **Remove passengers of delayed flight from rows** Decrement number of passengers by subtracting number of passengers on delayed flight from initial number of passengers ✓ Create temporary array ✓ *(size of remaining passengers)* Loop through passenger array✓ If passenger not on delayed flight✓ Copy to temporary array✓ Overwrite original passenger array✓ Update queues at counters ✓ Display updated output ✓ (8)	**14**	
	TOTAL:	**40**	

SUMMARY OF CANDIDATES MARKS:

CENTRE NUMBER:		**EXAMINATION NUMBER:**		
	SECTION A	**SECTION B**	**SECTION C**	
	QUESTION 1	**QUESTION 2**	**QUESTION 3**	**GRAND TOTAL**
MAX. MARKS	**50**	**60**	**40**	**150**
CANDIDATES MARKS				

ANNEXURE D: SOLUTION FOR QUESTION 1: JAVA

```
//    A solution to Question 1
//    Provided code
      final double COST_PER_KG = 50.0;
==========================================================================
// Question 1.1
==========================================================================
private void btnQues11ActionPerformed(java.awt.event.ActionEvent evt) {
        int age = Integer.parseInt(txfAge.getText().trim());
        if (chbPassport.isSelected()) {
            if (age >= 16 || (age < 16 && chbMinor.isSelected())) {
                txfQues11.setText("Boarding is confirmed");
            } else {
                txfQues11.setText("Boarding is not confirmed");
            }
        } else {
            txfQues11.setText("Boarding is not confirmed");
        }
    }
==========================================================================
// Question 1.2
==========================================================================
    private void btnQues12ActionPerformed(java.awt.event.ActionEvent evt) {

        double excess = 0;
        double cost = 0;
        double weight = Double.parseDouble(txfWeight.getText());
        String airlineDetails = "" + lstAirlineDetails.getSelectedValue();
        int psnHash = airlineDetails.indexOf("#");
        double maxWeight = Double.parseDouble(airlineDetails.substring
        (psnHash + 1, airlineDetails.length() - 2));

        if (weight > maxWeight) {
            excess = weight - maxWeight;
            cost = excess * COST_PER_KG;
        }
        txaQues12.setText("Excess weight:  " + String.format("%-6.2f", excess)
        + "kg" + "\n" + "Cost: " + String.format("%-6.2f", cost));
    }
==========================================================================
// Question 1.3
==========================================================================
private void btnQues13ActionPerformed(java.awt.event.ActionEvent evt) {
        int numPassengers = 0;
        int numVeg = 0;
        int numNonVeg = 0;
        numPassengers = Integer.parseInt(txfNumPassengers.getText());
        numVeg = numPassengers / 3;
        numNonVeg = numPassengers - numVeg;
        txaQues13.setText("Vegetarian meals: " + numVeg);
        txaQues13.append("\nNon-vegetarian meals: " + numNonVeg);
    }
```

```
================================================================================
// Question 1.4
================================================================================
    private void btnQues14ActionPerformed(java.awt.event.ActionEvent evt) {
        String boardingTime = "";
        boolean bLen = (txfFlightTime.getText().length() == 5);
        boolean bFlightTime = (txfFlightTime.getText().charAt(2) == 'h');

        if (bLen && bFlightTime) {
            int hour = Integer.parseInt(txfFlightTime.getText().substring(0, 2));
            int minute = Integer.parseInt(txfFlightTime.getText().substring(3, 5));

            if ((hour >= 0) && (hour <= 23) && (minute >= 0) && (minute <= 59)) {
                if (minute >= 35) {
                    minute -= 35;
                } else {
                    hour -= 1;
                    minute = minute + 60 - 35;
                }
                if (hour < 10) {
                    boardingTime = "0" + hour + "h";

                } else {
                    boardingTime = hour + "h";
                }
                if (minute < 10) {
                    boardingTime = boardingTime + "0" + minute;

                } else {
                    boardingTime = boardingTime + minute;
                }
                txfQues13.setText(boardingTime);
            }
            else {
              JOptionPane.showMessageDialog(null, "Invalid time entered.");
              txfFlightTime.setText("");
            }
        } else {
            JOptionPane.showMessageDialog(null, "Invalid time entered.");
            txfFlightTime.setText("");
        }
    }

================================================================================
// Question 1.5
================================================================================
private void btnQues15ActionPerformed(java.awt.event.ActionEvent evt) {
        double bonusPoints = 0;
        double points = 0;
        String flyerStatus = "NonMember";
        double distance = Double.parseDouble(txfDistance.getText());
        boolean isMember = false;
        if (rbtSilver.isSelected()) {
            isMember = true;
            flyerStatus = "Silver";
        }
        if (rbtGold.isSelected()) {
            isMember = true;
            flyerStatus = "Gold";
            bonusPoints = distance * 0.15;
        }
        if (rbtPlatinum.isSelected()) {
            isMember = true;
```

```
            flyerStatus = "Platinum";
            bonusPoints = distance * 0.2;
        }
        String fileName = flyerStatus + ".jpg";

try {
            lblImage.setIcon(new
            javax.swing.ImageIcon(getClass().getResource(fileName)));
        } catch (Exception e) {
            System.out.println(e);
        }
        if (isMember) {
            pnlPoints.setVisible(true);
            points = distance / 1.6 + bonusPoints;
            lblPointsSetText("Points earned: " + (int)Math.floor(points));
    }
  }
```

ANNEXURE E: SOLUTION FOR QUESTION 2: JAVA

```
//A solution to Question 2
  public class Flight {

    // Provided code
    private String flightNumber;
    private String city;
    private String date;
    private int numPassengers;
    ========================================================================
    // Question 2.1.1
    ========================================================================
    public Flight(String flightNumber, String city, String date) {
        this.flightNumber = flightNumber;
        this.city = city;
        this.date = date;
        setNumPassengers();
    }
    ========================================================================
    // Question 2.1.2
    ========================================================================
    public String getFlightNumber() {
        return flightNumber;
    }
    public int getNumPassengers() {
        return numPassengers;
    }
    ========================================================================
    // Provided code
    ========================================================================
    public void setNumPassengers() {
        this.numPassengers = 0;
    }
    ========================================================================
    // Question 2.1.3
    ========================================================================
    public void increasePassengers()
    {
        numPassengers++;
    }
    ========================================================================
    // Question 2.1.4
    ========================================================================
     public double calcPercBookings(double maxPassengers)
    {
        double perc = numPassengers/maxPassengers * 100;
        return perc;
    }
    ========================================================================
    // Question 2.1.5
    ========================================================================
    public String toString()
    {
        return "Flight number: " + flightNumber + "\nDestination: " + city +
            "\nDeparture date: " + date + "\nNumber of passengers booked: " +
            numPassengers ;
    }
```

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_Memo extends javax.swing.JFrame {
=============================================================================
// Provided code
=============================================================================
Flight objFlight = null;

public Question2_Memo() {
        initComponents();
        this.setLocationRelativeTo(this);
        this.setVisible(true);
        btnQ2_2_2.setEnabled(false);
        btnQ2_2_3.setEnabled(false);    }
=============================================================================
// Question 2.2.1
=============================================================================
private void btnQ2_2_1ActionPerformed(java.awt.event.ActionEvent evt) {
        String line = "" + cmbFlightDetails.getSelectedItem();
        String[] temp = line.split("#");
        objFlight = new Flight(temp[0], temp[1], temp[2]);
        btnQ2_2_2.setEnabled(true);
        btnQ2_2_3.setEnabled(true);
        JOptionPane.showMessageDialog(rootPane, "Flight object has been
        instantiated");
     }

=============================================================================
// Question 2.2.2
=============================================================================
private void btnQ2_2_2ActionPerformed(java.awt.event.ActionEvent evt) {
        objFlight.setNumPassengers();
        try
        {
            Scanner scFile = new Scanner(new FileReader("DataQ2.txt"));
            txaOutput.setText("List of passengers: \n");
            while (scFile.hasNext())
            {
                String line = scFile.nextLine();
                if(line.contains(objFlight.getFlightNumber()))
                {
                    txaOutput.append(line + "\n");
                    objFlight.increasePassengers();
                }
            }
            txaOutput.append("\n" + objFlight.toString());
            txaOutput.append("\n"+objFlight.determineStatus()+" flight");
        }
        catch (FileNotFoundException e)
        {
            JOptionPane.showMessageDialog(rootPane, "File does not exists");
            System.exit(0);
        }
}
```

```
================================================================================
// Question 2.2.3
================================================================================
private void btnQ2_2_3ActionPerformed(java.awt.event.ActionEvent evt) {
   double maxPassengers = Double.parseDouble(JOptionPane.showInputDialog(null,
   "Enter number of passengers"));
        double percentage = objFlight.calcPercBookings(maxPassengers);
        if (percentage >= 100)
          JOptionPane.showMessageDialog(rootPane, "Fully booked");
        else
        {
           JOptionPane.showMessageDialog(rootPane, "Percentage booked: " +
                       String.format("%.1f",percentage) + "%");
           String name = JOptionPane.showInputDialog(null, "Enter name of new
                              passenger");
           int numPass = objFlight.getNumPassengers()+ 1;
           String output = objFlight.getFlightNumber() + '-' + numPass + "    "
                         + name;
           try {
             PrintWriter pw = new PrintWriter(new FileWriter("DataQ2.txt",
                                                    true));
             pw.println(output);
             pw.close();
            }
            catch (Exception e)
            {
            JOptionPane.showMessageDialog(rootPane, "Error writing to file");
            }
        btnQ2_2_2.doClick();
       }
    }
================================================================================
```

ANNEXURE F: SOLUTION FOR QUESTION 3: JAVA

CLASS TO POPULATE ARRAYS: Populate Arrays

```
// Provided code

package Question3_Package;

public class PopulateArrays {

static String[] arrPosPassengers = {"E01;TDB2506", "E02;TDB1305",
"E03;TDB1305", "E04;TDB2506", "E05;TDB2506", "B06;TDB4310", "E07;TDB4310",
"B08;TDB1305", "E09;TDB4310", "B10;TDB2506", "E11;TDB1305", "B12;TDB4310",
"B13;TDB2506", "B14;TDB4310", "E15;TDB2506", "E16;TDB1305", "E17;TDB2506",
"E18;TDB1305", "E19;TDB4310", "E20;TDB4310", "E21;TDB1305", "B22;TDB1305",
"B23;TDB2506", "E24;TDB4310", "E25;TDB1305", "E26;TDB4310", "B27;TDB1305",
"B28;TDB1305", "E29;TDB4310", "E30;TDB2506", "B31;TDB1305", "E32;TDB2506",
"E33;TDB2506", "E34;TDB1305", "B35;TDB1305"};

    public static String[] fillRandom() {
        int size = (int) (Math.random() * (35) + 1);

        String[] arrPassengers;
        arrPassengers = new String[size];
        for (int cnt = 0; cnt < size; cnt++) {
            arrPassengers[cnt] = arrPosPassengers[cnt];
        }
        return arrPassengers;

    }

    public static String[][] initialize2D() {
        String[][] arrGrid = new String[9][4];
        for (int row = 0; row < 9; row++) {
            for (int col = 0; col < 4; col++) {
                arrGrid[row][col] = "";
            }
        }
        return arrGrid;
    }
}
```

CLASS: Question3_Memo

```
int numCounters = 0;
String delayed = "";
int rows = 0;
int cols = 0;
int numPass = 0;
//============================================================================
//Provided code
//============================================================================

// Global arrays
String[] arrPassengers = PopulateArrays.fillRandom();
String[][] arrGrid = PopulateArrays.initialize2D();

public Question3_Memo() {
        initComponents();
    }
============================================================================
// Question 3.1
============================================================================
private void btnDisplayActionPerformed(java.awt.event.ActionEvent evt) {
  numPass = arrPassengers.length;
        if (numPass >= 1 && numPass < 10) {
            numCounters = 1;
        }
        if (numPass >= 10 && numPass <= 16) {
            numCounters = 2;
        }
        if (numPass > 16 && numPass < 25) {
            numCounters = 3;
        }
        if (numPass >= 25) {
            numCounters = 4;
        }
        rows = (int) Math.ceil(numPass / (double) numCounters);
        cols = numCounters;
        placeInQueue();
        display();
    }

    public void placeInQueue() {
        for (int outside = 0; outside < numPass - 1; outside++) {
            for (int inside = outside + 1; inside < numPass; inside++) {
                if (arrPassengers[inside].compareTo(arrPassengers[outside])< 0)
                  {
                    String temp = arrPassengers[inside];
                    arrPassengers[inside] = arrPassengers[outside];
                    arrPassengers[outside] = temp;
                }
            }           }
        int counter = 0;
        for (int row = 0; row < rows; row++) {
            int col = 0;
            while (col < cols && counter < numPass) {
                arrGrid[row][col] = arrPassengers[counter];
                col++;
                counter++;
            }
        }
    }
```

```
    public void display() {
        txaOutput.setText("");
        for (int cnt = 1; cnt <= cols; cnt++) {
            txaOutput.append(String.format("%-12s", "Counter " + cnt));
        }
        txaOutput.append("\n");
        int counter = 0;
        for (int row = 0; row < rows; row++) {
            int col = 0;
            while (col < cols && counter < numPass) {
                String[] line = arrGrid[row][col].split(";");
                txaOutput.append(String.format("%-12s", line[0]));
                col++;
                counter++;
            }
            txaOutput.append("\n");
        }
    }
=============================================================================
// Question 3.2
=============================================================================
private void cmbDelayedActionPerformed(java.awt.event.ActionEvent evt) {
        delayed = cmbDelayed.getSelectedItem() + "";
    }

private void btnNewListActionPerformed(java.awt.event.ActionEvent evt) {
        int howMany = 0;
        txaFastService.setText("Flight number: " + delayed + "\n");
        for (int cnt = 0; cnt < numPass; cnt++) {
            if (arrPassengers[cnt].contains(delayed)) {
                howMany++;
                String[] temp = arrPassengers[cnt].split(";");
                txaDelayed.append(temp[0] + "\n");
             }
        }
        numPass = numPass - howMany;
        adjust();
    }

    public void adjust() {
        String[] temp = new String[numPass];
        int counter = 0;
        for (int cnt = 0; cnt < arrPassengers.length; cnt++) {
            if (!arrPassengers[cnt].contains(delayed)) {
                temp[counter] = arrPassengers[cnt];
                counter++;
            }
        }
        arrPassengers = temp;
        placeInQueue();
        display();
    }
```

ANNEXURE G: SOLUTION FOR QUESTION 1: DELPHI

```
unit Question1_U;

interface

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

type
  TfrmQuestion1 = class(TForm)
    Panel1: TPanel;
    GroupBox1: TGroupBox;
    Label1: TLabel;
    edtAge: TEdit;
    GroupBox2: TGroupBox;
    chbMinor: TCheckBox;
    btnQues11: TButton;
    edtQues11: TEdit;
    chbPassport: TCheckBox;
    lstMaxWeight: TListBox;
    edtWeight: TEdit;
    Label2: TLabel;
    btnQues12: TButton;
    GroupBox3: TGroupBox;
    edtFlightTime: TEdit;
    Label3: TLabel;
    btnQues14: TButton;
    edtQues13: TEdit;
    GroupBox4: TGroupBox;
    btnQues15: TButton;
    redQues12: TRichEdit;
    Label6: TLabel;
    Label8: TLabel;
    GroupBox5: TGroupBox;
    Label4: TLabel;
    edtNumPassengers: TEdit;
    btnQues13: TButton;
    redQues13: TRichEdit;
    Label5: TLabel;
    edtDistance: TEdit;
    rgpFlyerCard: TRadioGroup;
    imgFlyerCard: TImage;
    pnlPoints: TPanel;
    procedure btnQues11Click(Sender: TObject);
    procedure btnQues12Click(Sender: TObject);
    procedure btnQues14Click(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure btnQues13Click(Sender: TObject);
    procedure btnQues15Click(Sender: TObject);

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

const
  rCostPerKg = 50;  // To be used in Question 1.2
```

```
var
  frmQuestion1: TfrmQuestion1;
  iCode: integer = -1; // set index of radio buttons
implementation
{$R *.dfm}
=============================================================================
// Question 1.1
=============================================================================
procedure TfrmQuestion1.btnQues11Click(Sender: TObject);
var
  sMessage: string;
  iAge: integer;
begin
  iAge := StrToInt(edtAge.Text);
  if (chbPassport.Checked) then
  begin
    if (iAge >= 16) or ((iAge < 16) and chbMinor.Checked) then
      edtQues11.Text := 'Boarding is confirmed.'
    else
      edtQues11.Text := 'Boarding is not confirmed.';
  end
  else
    edtQues11.Text := 'Boarding is not confirmed.';
end;
=============================================================================
// Question 1.2
=============================================================================
procedure TfrmQuestion1.btnQues12Click(Sender: TObject);
var
  rWeight, rMaxWeight, rExcess, rCost: real;
  sAirline: string;
  iPosnHash: integer;
begin
  rExcess := 0;
  rCost := 0;
  rWeight := StrToFloat(edtWeight.Text);
  sAirline := lstMaxWeight.Items[lstMaxWeight.ItemIndex];
  iPosnHash := pos('#', sAirline);
  Delete(sAirline, 1, iPosnHash);
  rMaxWeight := StrToFloat(Copy(sAirline, 1, length(sAirline) - 2));
  if rWeight > rMaxWeight then
  begin
    rExcess := rWeight - rMaxWeight;
    rCost := rExcess * rCostPerKg;
  end;
  redQues12.Text := 'Excess weight: ' + FloatToStrF(rExcess, ffFixed, 5, 2)
    + 'kg' + #13 + 'Cost: ' + FloatToStrF(rCost, ffCurrency, 4, 2);
end;
=============================================================================
// Question 1.3
=============================================================================
procedure TfrmQuestion1.btnQues13Click(Sender: TObject);
Var
  iNumPassengers, iNumVeg, iNumNonNeg: integer;
begin
  iNumPassengers := StrToInt(edtNumPassengers.Text);
  iNumVeg := iNumPassengers div 3;
  iNumNonNeg := iNumPassengers - iNumVeg;
  redQues13.Clear;
  redQues13.Lines.Add('Vegetarian meals: ' + IntToStr(iNumVeg));
  redQues13.Lines.Add('Non-vegetarian meals: ' + IntToStr(iNumNonNeg));
end;
=============================================================================
```

```
// Question 1.4
=============================================================================
procedure TfrmQuestion1.btnQues14Click(Sender: TObject);
Var
  bLen,bFlightTime:boolean;
  sFlightTime, sBoardingTime: string;
  iHour, iMinute: integer;
begin
  sBoardingTime := '';

  bLen:= length(edtFlightTime.Text) = 5;       //1
  bFlightTime:= edtFlightTime.Text[3] = 'h';         //1

if bLen and bFlightTime then
 begin
  iHour := StrToInt(Copy(edtFlightTime.Text, 1, 2)); // 1
  iMinute := StrToInt(Copy(edtFlightTime.Text, 4, 2)); //1

   if  (ihour in [0..23]) and (iMinute in [0..59]) then   //1
         begin
          if (iMinute >= 35) then // 1
            iMinute := iMinute - 35 // 1
             else
                  begin
                    iHour := iHour - 1; // 1
                     iMinute := 60 + iMinute - 35; // 1
                  end;
          if iHour < 10 then // 1
                sBoardingTime := '0' + IntToStr(iHour) // 1
              else
                sBoardingTime := IntToStr(iHour); // 1

          if iMinute < 10 then // 1
                sBoardingTime := sBoardingTime + 'h' + '0' + IntToStr(iMinute)
// 1
               else
                sBoardingTime := sBoardingTime + 'h' + IntToStr(iMinute); // 1

           edtQues13.Text := sBoardingTime; // 1
           end
           else
           begin
                 ShowMessage('Invalid time was entered.');
                  edtFlightTime.Text := '';
           end; // if format valid
         end
  else
  begin
    ShowMessage('Invalid time was entered.'); // 1
    edtFlightTime.Text := '';
  end;
end;
=============================================================================
// Question 1.5
=============================================================================
procedure TfrmQuestion1.btnQues15Click(Sender: TObject);
var
  rDistance, rBonusPoints, rPoints: real;
  sFlyerStatus, sFilename: String;
  I: integer;
begin
  sFlyerStatus := 'NonMember';
  rDistance := StrToFloat(edtDistance.Text);
```

```
  iCode := rgpFlyerCard.ItemIndex;
  rBonusPoints := 0;
  case iCode of
    0:
      begin
        sFlyerStatus := 'Silver';
      end;
    1:
      begin
        sFlyerStatus := 'Gold';
        rBonusPoints := rDistance * 0.15;

      end;
    2:
      begin
        sFlyerStatus := 'Platinum';
        rBonusPoints := rDistance * 0.20;
      end;
  end;
  sFilename := sFlyerStatus + '.jpg';
  if FileExists(sFilename) then
    imgFlyerCard.Picture.LoadFromFile(sFilename);
  if iCode >= 0 then
  begin
    pnlPoints.Show;
    rPoints := Floor(rDistance / 1.6 + rBonusPoints);
    pnlPoints.Caption := 'Points earned: ' + FloatToStr(rPoints);
  end;
end;

procedure TfrmQuestion1.FormActivate(Sender: TObject);
begin
  currencyString := 'R';
  pnlPoints.Hide;
end;

end.
```

ANNEXURE H: SOLUTION FOR QUESTION 2: DELPHI

```
unit Flight_U;

interface

uses SysUtils, Math;

type
  TFlight = class
  private
    fFlightNumber : String;
    fCity : String;
    fDate : String;
    fNumPassengers : integer;

  public
    constructor create(sFlightNumber:String; sCity: String; sDate: String);
    function  getFlightNumber:String;
    function getNumPassengers:integer;
    procedure setNumPassengers;
    procedure increasePassengers;
    function calcPercBooked(iMax :integer):double;
    function toString:String;

  end;

implementation

{ TFlight }

=============================================================================
// Question 2.1.1
=============================================================================
constructor TFlight.create(sFlightNumber:String; sCity:String; sDate: String);
 begin
  fFlightNumber := sFlightNumber;
  fCity := sCity;
  fDate := sDate;
  setNumPassengers;
 end;
=============================================================================
// Question 2.1.2
=============================================================================
function TFlight.getFlightNumber: String;
begin
      Result := fFlightNumber;
end;

function TFlight.getNumPassengers: integer;
begin
   Result := fNumPassengers;
end;
=============================================================================
// Provided code
=============================================================================
procedure TFlight.setNumPassengers;
 begin
  fNumPassengers:=0;
 end;
```

```
================================================================================
// Question 2.1.3
================================================================================
procedure TFlight.increasePassengers;
begin
  Inc(fNumPassengers);
end;
================================================================================
// Question 2.1.4
================================================================================
function TFlight.calcPercBooked(iMax :integer):double;
 begin
    result := fNumPassengers/iMax*100;
 end;
================================================================================
// Question 2.1.5
================================================================================
function TFlight.toString: string;
begin
  result := 'Flight number: ' + fFlightNumber + #13 + 'Destination: ' + fCity +
      #13 + 'Departure date: ' + fDate + #13 + 'Number of passengers booked: '
           + IntToStr(fNumPassengers);
end;
end.
```

MAIN FORM UNIT: QUESTION2_U.PAS

```
unit Quest2_U;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Flight_U, ComCtrls, ExtCtrls;
type
  TfrmFlight = class(TForm)
    cmbFlightDetails: TComboBox;
    btnQues221: TButton;
    redQ2: TRichEdit;
    btnQues222: TButton;
    btnQues223: TButton;
    edtName: TEdit;
    GroupBox1: TGroupBox;
    Label1: TLabel;
    procedure btnQues222Click(Sender: TObject);
    procedure btnQues221Click(Sender: TObject);
    procedure btnQues223Click(Sender: TObject);
    procedure FormActivate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  frmFlight: TfrmFlight;
  objFlight: TFlight;

implementation
{$R *.dfm}
=============================================================================
// Question 2.2.1
=============================================================================
procedure TfrmFlight.btnQues221Click(Sender: TObject);
Var
  sFlightDetails, sFlightNumber, sCity, sDate: String;
  iPos: integer;
begin
  redQ2.Lines.Clear;
  sFlightDetails := cmbFlightDetails.Items[cmbFlightDetails.ItemIndex];
  iPos := pos('#', sFlightDetails);
  sFlightNumber := copy(sFlightDetails, 1, iPos - 1);
  Delete(sFlightDetails, 1, iPos);

  iPos := pos('#', sFlightDetails);
  sCity := copy(sFlightDetails, 1, iPos - 1);
  Delete(sFlightDetails, 1, iPos);

  sDate := sFlightDetails;

  objFlight := TFlight.create(sFlightNumber, sCity, sDate);
  btnQues222.Enabled:=true;
  btnQues223.Enabled:=true;
  ShowMessage('Flight object has been instantiated.');
end;
```

```
=============================================================================
// Question 2.2.2
=============================================================================
procedure TfrmFlight.btnQues222Click(Sender: TObject);
var
  myFile : TextFile;
  oneline: string;
begin
  redQ2.Lines.Clear;
  objFlight.setNumPassengers;
  redQ2.Lines.Add('List of passengers');
  AssignFile(myFile, 'DataQ2.txt');
  if not FileExists('DataQ2.txt') then
   begin
    ShowMessage('The file does not exist');
    exit;
   end
  else
   begin
    reset(myFile);
    while not eof(myFile) do
     begin
      readln(myFile, oneline);
      if pos(objFlight.getFlightNumber, oneline) > 0 then
      begin
        objFlight.increasePassengers;
        redQ2.Lines.Add(oneline);
      end;
     end;
    redQ2.Lines.Add(#13 + objFlight.toString);
   end;
  CloseFile(myFile);
end;
=============================================================================
// Question 2.2.3
=============================================================================
procedure TfrmFlight.btnQues223Click(Sender: TObject);
var
  sName, sReference: String;
  myFile: TextFile;
  iMaxPassengers, iNumPassengers: integer;
  rPercentage : real;
begin
  iMaxPassengers := StrToInt(Inputbox('','Enter the maximum number of
passengers', ''));
  rPercentage := objFlight.calcPercBooked(iMaxPassengers);
  if (rPercentage >= 100) then
    Showmessage('Fully booked')
  else
    begin
      Showmessage('Percentage booked: ' + FloatToStrF(rPercentage, ffFixed,0,1)
+ '%');
      sName := Inputbox('','Name of new passenger: ','');
      AssignFile(myFile, 'DataQ2.txt');
      Append(myFile);
      sReference := objFlight.getFlightNumber + '-' +
                    IntToStr(objFlight.getNumPassengers + 1);
      writeln(myFile, sReference + '    ' + sName);
      CloseFile(myFile);
      btnQues222.Click;
    end;
end;
procedure TfrmFlight.FormActivate(Sender: TObject);
```

```
begin
  btnQues222.Enabled:=false;
  btnQues223.Enabled:=false;
end;
end.
```

ANNEXURE I: SOLUTION FOR QUESTION 3: DELPHI

```
unit Question3_U;

interface

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

type

  TfrmQuest3 = class(TForm)
    Panel1: TPanel;
    gpbCounters: TGroupBox;
    gpbDelayed: TGroupBox;
    btnDisplayQueue: TButton;
    btnNewList: TButton;
    cbbFlightNumber: TComboBox;
    Label1: TLabel;
    redDelayed: TRichEdit;
    stgCounters: TStringGrid;
    gpbDelayedFlight: TGroupBox;
    procedure FormActivate(Sender: TObject);
    procedure btnDisplayQueueClick(Sender: TObject);
    procedure btnNewListClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmQuest3: TfrmQuest3;
  arrPosPassengers,arrPassengers,arrTemp:array[1..35] of string;
  iCounters,iSize,iRow, iCol: integer;
  arrGrid: array[1..9,1..4] of string;
  iNumRows,iNum:integer;

implementation

{$R *.dfm}
 // Sorts the array
Procedure Sortit;
var
  I: Integer;
  j: Integer;
  sTemp:string;
begin
  for I := 1 to iSize - 1 do
    for j := 1+i to iSize do
       if arrPassengers[I] > arrPassengers[J] then
       begin
           sTemp:=arrPassengers[i];
           arrPassengers[I] := arrPassengers[J];
           arrPassengers[J] := sTemp;
       end;
end;
```

```
// Provided code
 procedure cleanGridArray;
 begin
 for iRow := 1 to 9 do
  for iCol := 1 to 4 do
    begin
      arrGrid[iRow,iCol]:='';
      frmQuest3.stgCounters.Cells[iCol,iRow]:='';
    end;
  end;
//=========================================================================
procedure DisplayDetails;
 begin
 //filling 2D array and displaying information in string grid
   iNumRows:=ceil(iSize/iCounters);
   iNum:=0;
   iRow:=1;
   while iNum < iSize do
    begin
        for iCol := 1 to iCounters do
         begin
          inc(iNum);
          arrGrid[iRow,iCol]:=
copy(arrPassengers[iNum],1,pos(';',arrPassengers[iNum])-1);
         end;
       inc(iRow);
   end;
 for iRow := 1 to iNumRows do
     for iCol := 1 to iCounters do
         frmQuest3.stgCounters.Cells[icol,iRow] := arrGrid[iRow,iCol];
end;
=========================================================================
 //Question 3.1
=========================================================================
procedure TfrmQuest3.btnDisplayQueueClick(Sender: TObject);
var
  iLoop: Integer;
begin

if iSize <= 10  then
  iCounters := 1
else
  if iSize <= 16  then
    iCounters:=2
  else
    if iSize <= 24 then
      iCounters:= 3
    else
      iCounters:= 4;

 for iLoop := 1 to iCounters do
   stgCounters.Cells[iLoop,0]:= 'Counter ' + IntToStr(iLoop);
Sortit;
DisplayDetails;
end;
```

```
=============================================================================
//Question 3.2
=============================================================================
procedure TfrmQuest3.btnNewListClick(Sender: TObject);
var
  I, iLoop,iNum,iNewCounter : Integer;
  sFlightNum : string;
begin
 redDelayed.Clear;
 iNewCOunter:=0;
 sFlightNum:=cbbFlightNumber.Items[cbbFlightNumber.itemindex];
 redDelayed.Lines.Add('Flight number:' + sFlightNum);
 for iLoop := 1 to iSize do
  begin
     if pos(sFlightNum,arrPassengers[iLoop]) > 0 then
      begin
         redDelayed.Lines.Add(copy(arrPassengers[iLoop],1,
                    pos(';',arrPassengers[iLoop])-1));
      end
     else
      begin
        inc(iNewCounter);
        arrTemp[iNewCounter]:=arrPassengers[iLoop];
      end;
 end;

 for I := 1 to 35 do
   arrPassengers[I]:='';
 iSize := iNewCounter;
 arrPassengers:=arrTemp;
 CleanGridArray;
 Sortit;
 DisplayDetails;
end;

// Provided code
=============================================================================
procedure TfrmQuest3.FormActivate(Sender: TObject);
var
   a:integer;
begin
   arrPosPassengers[1] := 'E01;TDB2506';
   arrPosPassengers[2] := 'E02;TDB1305';
   arrPosPassengers[3] := 'E03;TDB1305';
   arrPosPassengers[4] := 'E04;TDB2506';
   arrPosPassengers[5] := 'E05;TDB2506';
   arrPosPassengers[6] := 'B06;TDB4310';
   arrPosPassengers[7] := 'E07;TDB4310';
   arrPosPassengers[8] := 'B08;TDB1305';
   arrPosPassengers[9] := 'E09;TDB4310';
   arrPosPassengers[10] := 'B10;TDB2506';
   arrPosPassengers[11] := 'E11;TDB1305';
   arrPosPassengers[12] := 'B12;TDB4310';
   arrPosPassengers[13] := 'B13;TDB2506';
   arrPosPassengers[14] := 'B14;TDB4310';
   arrPosPassengers[15] := 'E15;TDB2506';
   arrPosPassengers[16] := 'E16;TDB1305';
   arrPosPassengers[17] := 'E17;TDB2506';
   arrPosPassengers[18] := 'E18;TDB1305';
```

```
    arrPosPassengers[19] := 'E19;TDB4310';
    arrPosPassengers[20] := 'E20;TDB4310';
    arrPosPassengers[21] := 'E21;TDB1305';
    arrPosPassengers[22] := 'B22;TDB1305';
    arrPosPassengers[23] := 'B23;TDB2506';
    arrPosPassengers[24] := 'E24;TDB4310';
    arrPosPassengers[25] := 'E25;TDB1305';
    arrPosPassengers[26] := 'E26;TDB4310';
    arrPosPassengers[27] := 'B27;TDB1305';
    arrPosPassengers[28] := 'B28;TDB1305';
    arrPosPassengers[29] := 'E29;TDB4310';
    arrPosPassengers[30] := 'E30;TDB2506';
    arrPosPassengers[31] := 'B31;TDB1305';
    arrPosPassengers[32] := 'E32;TDB2506';
    arrPosPassengers[33] := 'E33;TDB2506';
    arrPosPassengers[34] := 'E34;TDB1305';
    arrPosPassengers[35] := 'B35;TDB1305';

    CleanGridArray;

    iSize:= random(35)+1;

    for a := 1 to iSize do
      arrPassengers[a]:=arrPosPassengers[a];

end;

end.
```