basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

SENIOR CERTIFICATE EXAMINATION

INFORMATION TECHNOLOGY P1

2015

MEMORANDUM

MARKS: 150

This memorandum consists 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..

- 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 3-7) include the marking grid for each question for using either one of the two programming languages.

- **Annexures D, E and F** (pages 6-17) 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, Band C** (pages 3-7) 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 combobox, penalise only once for the error.		
1.1	**Button - [Question 1.1]** Extract category from the combo box (to be used as text)✓ Construct a line of text using the event and category ✓ Assign the string to the label provided✓	**3**	
1.2	**Button - [Question 1.2]** Extract the number of participants from the text box and convert to a number✓ Check if the number of participants is less than or = to 20 ✓✓ Calculate cost ✓ else ✓ or check if the number of participants is more than 20 Calculate cost ✓ Place cost in text box ✓	**7**	
1.3	**Button - [Question 1.3]** Input value for base✓ Input value for exponent✓ Calculate base raised to the exponent ✓✓ Calculate square root ✓ Display the answer in the text box✓ formatted to 2 decimal places✓	**7**	
1.4	**Button - [Question 1.4]** Calculate total number of slices required for participants✓ Calculate the exact number of pizzas required as a decimal number✓ Display the exact value in the output area to 4 decimal spaces✓ Display a heading for number of pizzas to order ✓ Check if the exact number✓ is equal to the integer part of the exact number ✓ Display the exact number ✓ else ✓ Add 1 to the integer part of the exact number ✓	**9**	

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

1.5	**Button - [Question 1.5]** **Question 1.5.1** If Round 1 radio button is selected ✓ Set number of questions to 5 ✓ If Round 2 radio button is selected Set number of questions to 8 ✓ If Bonus questions checkbox selected ✓ in Round2 if ✓ or add a condition to the bonus question if Add value of 2 to number of questions ✓ **Question 1.5.2** outerLoop from 1 to number of questions ✓ ...innerLoop number of questions✓ downto ✓ value of outerloop ✓	**10**	
1.6	**Button - [Question 1.6]** Set number of seats to 30✓ Set ticket price to 50✓ Set total income to 0 ✓ Display heading ✓ neatly formatted✓ Loop from 1 to 20 ✓ Calculate row income ✓ Add row income to total income ✓ Display row values✓ formatted ✓ Increase the number of seats by 2 ✓ Decrease the ticket price by 2✓ Display the total income in the output area with label formatted to 2 decimal places ✓ All monetary values to currency ✓	**14**	
	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	Declare attributes ✓ with the correct data types (String/Integer ✓; Boolean/double✓)	**3**	
2.1.2	(a) Remove comment symbols from accessor methods✓ (b) outer loop from 1 to number of questions✓ Inner loop from number of questions✓ down to✓ counter of outer loop✓	**4**	
2.1.3	**Constructor:** Parameters name group, single and paid ✓ Correct data types for parameters ✓ Assign parameter values to attributes✓	**3**	
2.1.4	**calcAmount method:** Return type correct✓ numGroup * 15.70 ✓ + numSingle * 12.50 ✓ (-1 if information received as parameters) return answer✓	**4**	
2.1.5	**calcDiscount method:** If numGroup >= 4 OR ✓numSingle >= 4 ✓ and paid = true ✓ discount = totalAmount * 5 / 100 ✓ Return value✓	**5**	
2.1.6	**toString METHOD:** Format and correct attributes (-1 for each incorrect attribute) Name: <Name of person who placed the order>✓ Group: <Group>✓ Single: <Single> ✓ Paid: <Yes or No>✓	**4**	

2.2.1	**Button – [Quest2.2.1]:** Obtain name from combo box✓ {*Delphi: AssignFile, Reset* *Java: Create object to read from file*} ✓✓ If file does not exist/catch ✓& display message✓ and exit✓ else (begin..end)✓ Read first line (outside loop) ✓ and before while loop ✓ While loop through file ✓ Read one line from text file✓ Get name from line of text ✓ Test if name from line matches selected name ✓✓ Get single, group and paid info from line of text ✓✓✓ (Split/pos) Paid set to char data type ✓ Instantiate object ✓with all arguments ✓ in the correct order✓ Message – object created✓ Details displayed using toString method ✓ Display total amount by calling the calcAmount method ✓ Enable the Question 2.2.2 and 2.2.3 buttons✓	**25**	
2.2.2	**Button – [Quest2.2.2]:** Check if not paid✓ Create new text file ✓ using the name field ✓ Write the fields group, single, total amount✓ on separate lines ✓ to the text file ✓ Close the text file ✓ Message to indicate file was created ✓ else ✓ Display message to indicate that account was paid ✓	**10**	
2.2.3	**Button – [Quest2.2.3]:** Call calcDiscount method ✓ Display the amount in text area formatted to currency ✓	**2**	

	TOTAL:	**60**	

ANNEXURE C:
SECTION C:

QUESTION 3: MARKING GRID–PROBLEM SOLVING

CENTRE NUMBER:	**EXAMINATION NUMBER:**

SECTION	**DESCRIPTION**	**MAX. MARKS**	**LEARNER'S MARKS**
User input validation	*Validate the month:* Check if user entered a month ✓ Convert to numeric value ✓ Check range✓✓ If invalid month value display message✓ clear and set focus to the input component✓ If no month value is entered, assign 'All' to string variable ✓ *Validate user input for event to remove duplicates:* Convert the string entered to uppercase/lowercase ✓ or use equalsIgnoreCase method (Java)	**8**	
Display of output	Heading contains event selected or All ✓ and month entered or All✓ Loop through from 1 to count✓ (not 45) If correct month ✓✓ and correct event ✓✓ Display event and date✓ in columns ✓	**9**	
Sorting of arrays	Declare temporary string variables for event and date ✓ Outer loop ✓ Inner loop ✓ if-statement compare dates✓ Swap events arrays✓✓; Swap dates array ✓✓	**8**	
Remove duplicates	Input event ✓, ignore/set the case of the characters✓ Correct outer loop ✓✓ Correct inner loop ✓ with controlling variable ✓ to test for duplicates ✓ Correct condition testing event ✓ Remove duplicates for both arrays / transfer to new arrays✓✓	**10**	
Program techniques	Modular design – at least two procedures/methods ✓✓ Descriptive variable names✓; Proper indentation ✓; Suitable comments ✓	**5**	
	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 possible solution to Question 1

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;
import java.text.DecimalFormat;
import javax.swing.JOptionPane;

public class Question1_Solution extends javax.swing.JFrame {
    DecimalFormat df = new DecimalFormat("0.00");
    String activityType = "Quiz";
    int numParticipants = 50;

    public Question1_Solution() {
        initComponents();
        this.setLocationRelativeTo(this);
        txfEvent.setText(activityType);
    }
=========================================================================
// Question 1.1
=========================================================================
    private void btnQues1_1ActionPerformed(java.awt.event.ActionEvent evt) {
        String category = (String)cmbCategory.getSelectedItem();
        lblOutput.setText(activityType + ":" + category);
    }
=========================================================================
// Question 1.2
=========================================================================
    private void btnQues1_2ActionPerformed(java.awt.event.ActionEvent evt) {
        numParticipants = Integer.parseInt(txfNumParticipants.getText());
        int cost = 0;
        if(numParticipants <= 20){
            cost = numParticipants * 25;
        }
        else {
            cost = (20 * 25) + ((numParticipants - 20) * 30);
        }
        txfIncome.setText("R" + df.format(cost));
    }
=========================================================================
// Question 1.3
=========================================================================
    private void btnQues1_3ActionPerformed(java.awt.event.ActionEvent evt) {
        int base     =  Integer.parseInt(JOptionPane.showInputDialog("Enter the
                                                                    base"));
        int exponent =  Integer.parseInt(JOptionPane.showInputDialog("Enter the
                                                                    exponent"));
        int power    =  (int)(Math.pow(base, exponent));
        double squareRoot = Math.sqrt(power);
        txfAns_1_3.setText("x = " + df.format(squareRoot));
    }
```

```
==========================================================================
// Question 1.4
==========================================================================

private void btnQues1_4ActionPerformed(java.awt.event.ActionEvent evt) {
        int numberOfPieces = numParticipants * 3;
        double numberOfPizzas = numberOfPieces / 8.0;
        txaOutput_1_4.setText("Exact value = " + String.format("%-
                                            6.4f",numberOfPizzas));
        txaOutput_1_4.append("\nNumber of pizzas to order = ");
        if(numberOfPizzas == (int)numberOfPizzas){
            txaOutput_1_4.append("" + (int)numberOfPizzas);
        }
        else
        {
            txaOutput_1_4.append("" + (((int)(numberOfPizzas)) + 1));
        }
    }
==========================================================================
// Question 1.5
==========================================================================
      private void btnQues1_5ActionPerformed(java.awt.event.ActionEvent evt) {
     //Question 1.5.1
     // Code to determine the number of questions

      txaOutput_1_5.setText("");
      int numQuestions = 0;
      if(rbtRound1.isSelected()){
           numQuestions = 5;
      }
      else
      {
        if(rbtRound2.isSelected())
           numQuestions = 8;
        if(chbBonus.isSelected())
           numQuestions += 2;
      }
      // Question 1.5.2
      txaOutput_1_5.setText("");
      txaOutput_1_5.append("Questions to answer\n");
      String output = "";
      for (int i = 1; i <= numQuestions; i++) {
           for (int j = numQuestions; j >= i; j--) {
                output = output + (j + " ");
           }
           output = output + "\n";
       }
      txaOutput_1_5.append(output + "\nQuiz completed");
   }
==========================================================================
// Question 1.6
==========================================================================
    private void btnQues1_6ActionPerformed(java.awt.event.ActionEvent evt) {
       int numSeats = 30;
       double ticketPrice = 50;
       double totalIncome = 0;
       txaOutput_1_6.append(String.format("%-10s%-10s%-16s%-
                              10s\n","Row","Seats","Ticket price","Income"));
       for (int i = 1; i <=20 ; i++) {
          double  rowIncome = numSeats * ticketPrice;
          totalIncome = totalIncome + rowIncome;
          txaOutput_1_6.append(String.format("%-10s%-10sR%-15.2fR%-
```

```
                            10.2f\n",i,numSeats,ticketPrice,rowIncome));
          numSeats = numSeats + 2;
          ticketPrice = ticketPrice - 2;

        }
        txaOutput_1_6.append("\nTotal income is: R" + df.format(totalIncome));
    }
}
```

ANNEXURE E: SOLUTION FOR QUESTION 2: JAVA

```
//A possible solution for the object class
```

OBJECT CLASS: DETAILS

```
package Question2Package;


================================================================================
// Question 2.1.1
================================================================================
public class Order
{
   private  String name;
   private  int numGroup;
   private  int numSingle;
   private  char paid;

================================================================================
// Question 2.1.2
================================================================================
// Question 2.1.2 (a)
    public String getName()
    {
        return name;
    }

    public int getNumGroup()
    {
        return numGroup;
    }

    public int getNumSingle()
    {
        return numSingle;
    }

    // Question 2.1.2 (b)
    public String getPaid()
    {
        if (paid == 'Y')
          return "Yes";
        else
          return "No";
    }

    ================================================================================
    // Question 2.1.3
    ================================================================================
    public Order(String Name, int numGroup, int numSingle, char paid)
    {
        this.name = Name;
        this.numGroup = numGroup;
        this.numSingle = numSingle;
        this.paid = paid;
    }
```

```
    ================================================================================
    // Question 2.1.4
    ================================================================================
    public double calcAmount()
    {
        double total = (numGroup * 15.70) + (numSingle * 12.50);
        return total;
    }

    ================================================================================
    // Question 2.1.5
    ================================================================================
    public double calcDiscount()
    {
        double discount = 0;
        if ((numGroup >= 4 || numSingle >= 4) &&
                              getPaid().equalsIgnoreCase("Yes")){
           discount = calcAmount() * 5 / 100.00;
       }
       return discount;
    }
    ================================================================================
    // Question 2.1.6
    ================================================================================
    public String toString()
    {
        String data = "Name: " + name + "\nGroup: " + numGroup+"\nSingle:
                                    " + numSingle + "\nPaid: " + getPaid();
        return data;
    }
}
```

GUI CLASS: QUESTION2_SOLUTION

```
package Question2Package;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.PrintWriter;
import java.util.Scanner;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
import javax.swing.JTextArea;


public class Question2_Solution extends javax.swing.JFrame {

    String nameOfSchool;
    String name;
    Details photoPackage;

    public Question2_Solution() {
        initComponents();
        setLocationRelativeTo(this);

    }
    ============================================================================
    // Question 2.2.1
    ============================================================================

    private void btnQues2_2_1ActionPerformed(java.awt.event.ActionEvent evt) {

        boolean bFound;
        txaOutput.setText("");
        name = cmbNames.getSelectedItem().toString();
        try
        {
            Scanner inFile = new Scanner(new FileReader("Photographs.txt"));
            nameOfSchool = inFile.nextLine();
            lblSchoolName.setText(nameOfSchool);
            bFound = false;
            while (inFile.hasNext() && !bFound) {
                String line = inFile.nextLine();
                Scanner scLine = new Scanner(line).useDelimiter("#|;");
                String newName = scLine.next();
                if (newName.equalsIgnoreCase(name)) {
                    int group = scLine.nextInt();
                    int single = scLine.nextInt();
                    char paid = scLine.next().charAt(0);

                    photoPackage = new Order(name, group, single, paid);
                    txaOutput.append(photoPackage.toString());
                    txaOutput.append(String.format("\n\nAmount: R%-5.2f",
                                    photoPackage.calcAmount()));
                    bFound = true;
                }
            }
            inFile.close();
            if (bFound == false){
                btnQues2_2_2.setEnabled(false);
                btnQues2_2_3.setEnabled(false);
                JOptionPane.showMessageDialog(null, name + " was not found in
```

```
                                                    text file");
            } else {
                btnQues2_2_2.setEnabled(true);
                btnQues2_2_3.setEnabled(true);
            }

        } catch (FileNotFoundException ex)
        {
            JOptionPane.showMessageDialog(null, "File does not exist");
            System.exit(0);
        }

    }
================================================================================
// Question 2.2.2
================================================================================
    private void btnQues2_2_2ActionPerformed(java.awt.event.ActionEvent evt) {

        PrintWriter outFile;
        if (!photoPackage.getPaid().equals("Yes")) {
            try
            {
                outFile = new PrintWriter(photoPackage.getName() + ".txt");

                outFile.println("Group: " + photoPackage.getNumGroup());
                outFile.println("Single: " + photoPackage.getNumSingle());
                outFile.println("Total amount: " + photoPackage.calcAmount());
                txaOutput.append("\nText file created ");
                outFile.close();
            } catch (FileNotFoundException ex)
            {
                JOptionPane.showMessageDialog(null, "File not found");
                System.exit(0);
            }

        } else
        {
            txaOutput.append("\nAccount paid ");
        }
    }

    ================================================================================
// Question 2.2.3
================================================================================
private void btnQues2_2_3ActionPerformed(java.awt.event.ActionEvent evt) {

        txaOutput.append(String.format("\nDiscount: R%-5.2f",
                                photoPackage.calcDiscount()) + "\n");
    }
}
```

ANNEXURE F: SOLUTION FOR QUESTION 3: JAVA

```
// A possible solution to Question 3

package Question3Package;

import java.text.DateFormat;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;

public class Question3_Solution extends javax.swing.JFrame {

    DateFormat df = new SimpleDateFormat("yyyy/MM/dd");

    String arrEvents[] = {
    "Hockey", "Chess", "Rugby", "Chess", "Debating", "Hockey", "Soccer",
    "Rugby", "Debating", "Chess", "Rugby", "Debating", "Swimming", "Rugby",
    "Soccer", "Rugby", "Chess", "Choir", "Tennis", "Debating",
    "Debating", "Rugby", "Tennis", "Tennis", "Tennis", "Soccer",
    "Chess", "Swimming", "Hockey", "Rugby", "Chess", "Hockey", "Soccer",
    "Swimming", "Chess", "Choir", "Soccer", "Tennis", "Debating",
    "Rugby", "Choir", "Chess", "Choir", "Debating", "Rugby"};

  String arrDates[] = {
    "2015/10/04", "2015/03/20", "2015/09/17", "2015/03/20", "2015/05/24",
    "2015/10/04", "2015/06/03", "2015/07/19", "2015/09/16", "2015/07/01",
    "2015/03/17", "2015/11/19", "2015/07/29", "2015/09/18", "2015/10/23",
    "2015/07/15", "2015/03/07", "2015/06/08", "2015/04/01", "2015/03/07",
    "2015/09/12", "2015/11/12", "2015/10/03", "2015/05/11", "2015/10/03",
    "2015/05/28", "2015/09/28", "2015/05/10", "2015/09/24", "2015/10/14",
    "2015/04/23", "2015/03/23", "2015/06/03", "2015/07/29", "2015/09/18",
    "2015/08/29", "2015/11/10", "2015/04/01", "2015/04/10", "2015/03/23",
    "2015/10/04", "2015/10/07", "2015/05/03", "2015/07/30", "2015/11/25"};

    int monthNumber;
    int count = 45;
    String eventName, sMonth;


    public Question3_Solution() {
        initComponents();
        setLocationRelativeTo(this);
        cmbEvents.setSelectedIndex(0);
    }

    private void btnCloseActionPerformed(java.awt.event.ActionEvent evt) {
        System.exit(0);
    }
//=========================================================================
    private void btnRemoveActionPerformed(java.awt.event.ActionEvent evt) {

      sortArrays();
      removeDuplicates();
      display();
    }
//=========================================================================

    private void btnDisplayActionPerformed(java.awt.event.ActionEvent evt) {
```

```
      sortArrays();
      display();
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Question3_Solution().setVisible(true);
            }
        });
    }

//=============================================================================

    public void display(){
    if (!txfMonthNumber.getText().equals("")) {
         monthNumber = Integer.parseInt(txfMonthNumber.getText());
         if (monthNumber < 1 || monthNumber > 12) {
            JOptionPane.showMessageDialog(null, "Please enter a valid number in
                                        the range of 1 to 12");
            txfMonthNumber.setText("");
         }
        }
        if (txfMonthNumber.getText().equals("")) {
            sMonth = "All";
        }
        else {
            sMonth = "" + monthNumber;
        }
        //JOptionPane.showMessageDialog(null,eventName.equals("All"));
        eventName = (String) (cmbEvents.getSelectedItem());

        txaOutput.setText("List of " + eventName.toLowerCase() + " events for
                                    month: " + sMonth + "\n\n");

        txaOutput.append(String.format("%-15s%-10s\n", "EVENTS", "DATES"));

         for (int i = 0; i < count; i++) {
            int testmonth = Integer.parseInt(arrDates[i].substring(5, 7));
            if (((monthNumber == testmonth) || (sMonth.equals("All"))) &&
                        ((eventName.equalsIgnoreCase(arrEvents[i])) ||
                        (eventName.equals("All")))) {
                txaOutput.append(String.format("%-15s%-10s\n", arrEvents[i],
                        arrDates[i]));
            }

           }
        }
    }
//=============================================================================

    public void sortArrays()
    {
       String tempDate = "";
       String tempEvent = "";
       for (int i = 0; i < count - 1; i++)
            for (int j = i + 1; j < count; j++) {
```

```
                if (arrDates[i].compareTo(arrDates[j]) > 0) {
                    tempDate = arrDates[i];
                    arrDates[i] = arrDates[j];
                    arrDates[j]  = tempDate;

                    tempEvent = arrEvents[i];
                    arrEvents[i] = arrEvents[j];
                    arrEvents[j] = tempEvent;
                }
            }
    }
//=============================================================================
```

```
public void removeDuplicates(){
     String event = JOptionPane.showInputDialog(null, "Enter the
                         event","");
     int i = 0;
     while (i < count) {
        int j = i + 1;
        while ((arrEvents[i].equalsIgnoreCase(event)) &&
            (arrDates[i].equals(arrDates[j]) && j <= count)) {
         for (int k = j; k < count - 1; k++){
           arrDates[k] = arrDates[k + 1];
           arrEvents[k] = arrEvents[k + 1];
         }
         count--;
        }
        i++;
     }
  }
```

ANNEXURE G: SOLUTION FOR QUESTION 1: DELPHI

```
unit Question1U_MEMO;
//A possible solution to Question 1

interface

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

type
  TfrmQuestion1 = class(TForm)
    pnlButtons: TPanel;
    bmbClose: TBitBtn;
    grbQ1_1: TGroupBox;
    lblOutput: TLabel;
    lblCategory: TLabel;
    lblEvent: TLabel;
    edtEvent: TEdit;
    cboCategory: TComboBox;
    btnQuestion1_1: TButton;
    grbQ1_2: TGroupBox;
    lblNumParticipants: TLabel;
    edtNumParticipants: TEdit;
    btnQuestion1_2: TButton;
    edtIncome: TEdit;
    grbQ1_3: TGroupBox;
    btnQuestion1_3: TButton;
    edtAnswer: TEdit;
    grbQ1_4: TGroupBox;
    btnQuestion1_4: TButton;
    grbQ1_5: TGroupBox;
    btnQuestion1_5: TButton;
    redQ14: TRichEdit;
    redQ15: TRichEdit;
    grbQ1_6: TGroupBox;
    btnQuestion1_6: TButton;
    redQ16: TRichEdit;
    cbkBonus: TCheckBox;
    rgpRound: TRadioGroup;
    procedure FormCreate(Sender: TObject);
    procedure btnQuestion1_1Click(Sender: TObject);
    procedure btnQuestion1_2Click(Sender: TObject);
    procedure btnQuestion1_3Click(Sender: TObject);
    procedure btnQuestion1_4Click(Sender: TObject);
    procedure btnQuestion1_6Click(Sender: TObject);
    procedure btnQuestion1_5Click(Sender: TObject);
    procedure rgpRoundClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmQuestion1: TfrmQuestion1;
```

```
implementation

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

uses Math, DateUtils;

var
    sEvent : String = 'Quiz';
    iNumParticipants : Integer = 50;

=============================================================================
// Question 1.1
=============================================================================
procedure TfrmQuestion1.btnQuestion1_1Click(Sender: TObject);
begin
    lblOutput.Caption := edtEvent.Text + ':' +
                         cboCategory.Items[cboCategory.ItemIndex];
end;
=============================================================================
// Question 1.2
=============================================================================
procedure TfrmQuestion1.btnQuestion1_2Click(Sender: TObject);
var
  rCost : real;
begin
  rCost := 0;
  iNumParticipants := StrToInt(edtNumParticipants.Text);

  if iNumParticipants <= 20 then
    begin
       rCost := iNumParticipants * 25.00;
    end
   else
    begin
       rCost := (20 * 25.00) + ((iNumParticipants-20)* 30.00);
    end;

   edtIncome.Text := FloatToStrF(rCost, ffCurrency, 8, 2);
end;
=============================================================================
// Question 1.3
=============================================================================
procedure TfrmQuestion1.btnQuestion1_3Click(Sender: TObject);
var
   iBase, iExp : integer;
   rAnswer     : real;
begin
  iBase := StrToInt(InputBox('Base value', 'Enter the base', ' '));
  iExp := StrToInt(InputBox('Exponent value',
                            'Enter the exponent', ' '));
  rAnswer := Sqrt(Power(iBase, iExp));

  edtAnswer.Text := 'x = ' + FloatToStrF(rAnswer, ffFixed, 8,2);
end;
```

```
================================================================================
// Question 1.4
================================================================================
procedure TfrmQuestion1.btnQuestion1_4Click(Sender: TObject);
var
   rPizzas : real;
begin
  rPizzas := (iNumParticipants * 3) / 8;
  redQ14.Clear;
  redQ14.Lines.Add('Exact value = ' + FloatToStrF(rPizzas, ffFixed, 8,4) +
                13+ 'Number of pizzas to order = ' +
                  FloatToStrF(Round(rPizzas + 0.5), ffFixed, 8, 0));
end;
================================================================================
// Question 1.5
================================================================================
procedure TfrmQuestion1.btnQuestion1_5Click(Sender: TObject);
var
   K,J,iNumQuestions : integer;
   sOutput : String;
begin
  //Question 1.5.1
  // Code to determine the number of questions
  if rgpRound.ItemIndex = 0 then
    begin
     iNumQuestions := 5;
     cbkBonus.checked := false;
    end
  else
    begin
     if rgpRound.ItemIndex = 1 then
     iNumQuestions := 8;

     if cbkBonus.checked then
     inc(iNumQuestions,2);
    end;
  //Question 1.5.2
  //Correct incorrect code to display pattern
  redQ15.Clear;
  redQ15.Lines.Add('Questions to answer') ;
  sOutput := '';
  for I := 1 to iNumQuestions do
    begin
      for J:= iNumQuestions downto I do
        sOutput := sOutput + IntToStr(J) + ' ';
      sOutput := sOutput + #13;
    end;
   redQ15.Lines.Add(sOutput + #13 + 'Quiz completed');
end;
================================================================================
// Question 1.6
================================================================================
procedure TfrmQuestion1.btnQuestion1_6Click(Sender: TObject);
var
   A, iRow, iNumSeats : integer;
   rTotalIncome, rRowIncome, rTicketPrice : real;
begin
  iNumSeats    := 30;
  rTotalIncome := 0.0;
  rTicketPrice := 50.00;
```

```
  redQ16.Clear;
  redQ16.Paragraph.TabCount := 3;
  redQ16.Paragraph.Tab[0]   := 50;
  redQ16.Paragraph.Tab[1]   := 110;
  redQ16.Paragraph.Tab[2]   := 180;
  redQ16.Lines.Add('Row' + #9 + 'Seats' + #9 + 'Ticket price' +
                        #9 + 'Income');
  for iRow := 1 to 20 do
   begin
      rRowIncome   := (iNumSeats * rTicketPrice);
      rTotalIncome := rTotalIncome + rRowIncome;

      redQ16.Lines.Add(IntToStr(iRow) + #9 + IntToStr(iNumSeats) + #9 +
             FloatToStrF(rTicketPrice, ffCurrency, 8, 2) + #9 +
             FloatToStrF(rRowIncome, ffCurrency, 8, 2));

      rTicketPrice := rTicketPrice - 2;
      Inc(iNumSeats, 2);
   end;
  redQ16.Lines.Add(#13 + 'Total income: ' +
                   FloatToStrF(rTotalIncome, ffCurrency, 8, 2));
end;
//=====================================================================
procedure TfrmQuestion1.FormCreate(Sender: TObject);
begin
   edtEvent.Text := sEvent;
   CurrencyString := 'R';

end;

procedure TfrmQuestion1.rgpRoundClick(Sender: TObject);
begin
   cbkBonus.Enabled := rgpRound.ItemIndex = 1;
end;

end.
```

ANNEXURE H: SOLUTION FOR QUESTION 2: DELPHI

OBJECT CLASS: DETAILS

```
unit OrderU;
  //A possible solution for the object class
interface
=============================================================================
// Question 2.1.1
=============================================================================
type
    TOrder = class(TObject)
      private
        fName             : String;
        fNumGroup,
        fNumSingle        : integer;
        fPaid             : Char;
      public
        function getName : String;
        function getNumGroup : integer;
        function getNumSingle: integer;
        function getPaid : String;
        constructor Create(Name : String; Group, Single : integer; Paid :
                                                                    Char);
        function calcAmount : real;
        function calcDisount : real;
        function toString    : String;
     end;

implementation

uses SysUtils, Math, DateUtils;

{ TDetails }
=============================================================================
// Question 2.1.2
=============================================================================
// Question 2.1.2 (a)
function TOrder.getName: String;
begin
  Result := fName;
end;

function TOrder.getNumGroup: integer;
begin
  Result := fNumGroup;
end;

function TOrder.getNumSingle: integer;
begin
  Result := fNumSingle;
end;
//Question 2.1.2 (b)
function TOrder.getPaid: String;
begin
  if (fPaid = 'Y') then
    Result := 'Yes'
  else
    Result := 'No';
end;
```

```
================================================================================
// Question 2.1.3
================================================================================
constructor TOrder.Create(Name: String; Group, Single: integer;
  Paid: char);
begin
   fName     := Name;
   fNumGroup := Group;
   fNumSingle:= Single;
   fPaid     := Paid;
end;

================================================================================
// Question 2.1.4
================================================================================
function TOrder.calcAmount: real;
begin
   Result := (fNumGroup * 15.70) + (fNumSingle * 12.50);
end;
================================================================================
// Question 2.1.5
================================================================================
function TOrder.calcDisount: real;
var
  rDiscount : real;
begin
  if ((fNumGroup >= 4) OR (fNumSingle >= 4)) AND (fPaid = 'Y') then
   begin
     rDiscount := calcAmount * 5 / 100.00;
   end;
   Result := rDiscount;
end;
================================================================================
// Question 2.1.6
================================================================================
function TOrder.toString: String;
begin
   Result := 'Name: ' + fName + #13 +
             'Group: ' + IntToStr(fNumGroup) + #13 +
             'Single: ' + IntToStr(fNumSingle)+ #13 +
             'Paid: ' + getPaid;
end;

end.
```

MAIN FORM UNIT: QUESTION2_U.PAS

```
unit Question2U_MEMO;
//==============================
// A possible solution for Question 2
//==============================
interface

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

type
  TfrmQuestionTwo = class(TForm)
    pnlButtons: TPanel;
    bmbClose: TBitBtn;
    cboNames: TComboBox;
    lblHeading: TLabel;
    lblSelect: TLabel;
    lblSchoolName: TLabel;
    redQ2: TRichEdit;
    btnQuestion2_2_1: TButton;
    btnQuestion2_2_2: TButton;
    btnQuestion2_2_3: TButton;
    procedure btnQuestion2_2_1Click(Sender: TObject);
    procedure btnQuestion2_2_2Click(Sender: TObject);
    procedure btnQuestion2_2_3Click(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmQuestionTwo: TfrmQuestionTwo;

implementation

uses DetailsU;
{$R *.dfm}
{$R+}

var
  PhotoPackage: TOrder;

//=====================================================================
procedure TfrmQuestionTwo.btnQuestion2_2_1Click(Sender: TObject);
var
  tFile : TextFile;
  sLine : String;
  sSchool, sName: String;
  bFound: boolean;
  iGroup, iSingle: integer;
  cPaid : Char;
begin
//  ===================================================================
// Question 2.2.1
//===================================================================
  redQ2.Clear;
  sName := cboNames.Items[cboNames.ItemIndex];
```

```
  if NOT FileExists('Photographs.txt') then
    begin
      MessageDlg('File does not exists', mtError, [mbOk], 0);
      Exit;
    end;

  AssignFile(tFile, 'Photographs.txt');
  Reset(tFile);
  readln(tFile, sSchool);
  lblSchoolName.Caption := sSchool;
  bFound := False;
  while NOT EOF(tFile) AND NOT bFound do
    begin
      readln(tFile, sLine);
      if pos(Uppercase(sName), Uppercase(sLine)) = 1 then
        begin
          Delete(sLine, 1, pos('#', sLine));
          iGroup := StrToInt(Copy(sLine, 1, pos(';', sLine) - 1));
          Delete(sLine, 1, pos(';', sLine));
          iSingle := StrToInt(Copy(sLine, 1, pos(';', sLine) - 1));
          Delete(sLine, 1, pos(';', sLine));
          cPaid := sLine[1]; // true = Y / false = not Y

          PhotoPackage := TOrder.Create(sName, iGroup, iSingle, cPaid);

          redQ2.Lines.Add(PhotoPackage.toString + #13 + #13+
                 'Amount: ' +
                 FloatToStrF(PhotoPackage.calcAmount, ffCurrency, 8, 2));
          bFound := True;
        end; // name found
    end;

  if NOT bFound then
   begin
    btnQuestion2_2_2.Enabled := False;
    btnQuestion2_2_3.Enabled := False;
    MessageDlg(sName + ' was not found in text file', mtError, [mbOk], 0);
   end
  else
   begin
    btnQuestion2_2_2.Enabled := True;
    btnQuestion2_2_3.Enabled := True;
   end;
end;
// ============================================================================
procedure TfrmQuestionTwo.btnQuestion2_2_2Click(Sender: TObject);
var
   tFile     : TextFile;
   sFileName : String;
begin
//  ============================================================================
// Question 2.2.2
// ============================================================================
if PhotoPackage.getPaid = 'Yes' then
    begin
       redQ2.Lines.Add('Account paid');
    end
   else
    begin
       sFileName := PhotoPackage.getName + '.txt';
       AssignFile(tFile, sFileName);
       Rewrite(tFile);
```

```
       Writeln(tFile, 'Group: ' + IntToStr(PhotoPackage.getNumGroup));
       Writeln(tFile, 'Single: ' + IntToStr(PhotoPackage.getNumSingle));
       Writeln(tFile, 'Amount: ' + FloatToStrF(PhotoPackage.calcAmount,
                                ffCurrency, 8, 2));
       CloseFile(tFile);
       redQ2.Lines.Add('Text file created');
    end;
end;
// ============================================================================
procedure TfrmQuestionTwo.btnQuestion2_2_3Click(Sender: TObject);
begin
//  ============================================================================
// Question 2.2.3
// ============================================================================
  redQ2.Lines.Add('Discount: ' +
            FloatToStrF(PhotoPackage.calcDisount, ffCurrency,8,2))
end;
// ============================================================================
// Given Code
procedure TfrmQuestionTwo.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  if NOT(PhotoPackage = nil) then
    begin
      FreeAndNil(PhotoPackage);
    end;
end;

procedure TfrmQuestionTwo.FormCreate(Sender: TObject);
begin
   CurrencyString := 'R';
end;
// ============================================================================
end.
```

ANNEXURE I: SOLUTION FOR QUESTION 3: DELPHI

```
unit Question3U_Memo;
// A possible solution for Question 3

interface

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

type
  TfrmQuestion3 = class(TForm)
    pnlButtons: TPanel;
    bmbClose: TBitBtn;
    lblHead: TLabel;
    btnDisplay: TButton;
    grbInput: TGroupBox;
    cboEvent: TComboBox;
    lblEvent: TLabel;
    edtMonth: TEdit;
    lblMonth: TLabel;
    redQ3: TRichEdit;
    btnRemoveDuplicates: TButton;
    procedure btnDisplayClick(Sender: TObject);
    procedure btnRemoveDuplicatesClick(Sender: TObject);
    procedure Display;
    procedure SortArrays;
    Procedure RemoveDuplicates;
    procedure bmbCloseClick(Sender: TObject);

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

Type
  StringArray = Array [1 .. 45] of String;

var
  frmQuestion3: TfrmQuestion3;

implementation

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

Uses DateUtils, Math;

var
  arrEvents: array [1 .. 45] of String = (
    'Hockey', 'Chess', 'Rugby', 'Chess', 'Debating', 'Hockey', 'Soccer',
    'Rugby', 'Debating', 'Chess', 'Rugby', 'Debating', 'Swimming', 'Rugby',
    'Soccer', 'Rugby', 'Chess', 'Choir', 'Tennis', 'Debating',
    'Debating', 'Rugby', 'Tennis', 'Tennis', 'Tennis', 'Soccer',
    'Chess', 'Swimming', 'Hockey', 'Rugby', 'Chess', 'Hockey', 'Soccer',
    'Swimming', 'Chess', 'Choir', 'Soccer', 'Tennis', 'Debating',
    'Rugby', 'Choir', 'Chess', 'Choir', 'Debating', 'Rugby');
```

```
  arrDates: array [1 .. 45] of string = (
    '2015/10/04', '2015/03/20', '2015/09/17', '2015/03/20', '2015/05/24',
    '2015/10/04', '2015/06/03', '2015/07/19', '2015/09/16', '2015/07/01',
    '2015/03/17', '2015/11/19', '2015/07/29', '2015/09/18', '2015/10/23',
    '2015/07/15', '2015/03/07', '2015/06/08', '2015/04/01', '2015/03/07',
    '2015/09/12', '2015/11/12', '2015/10/03', '2015/05/11', '2015/10/03',
    '2015/05/28', '2015/09/28', '2015/05/10', '2015/09/24', '2015/10/14',
    '2015/04/23', '2015/03/23', '2015/06/03', '2015/07/29', '2015/09/18',
    '2015/08/29', '2015/11/10', '2015/04/01', '2015/04/10', '2015/03/23',
    '2015/10/04', '2015/10/07', '2015/05/03', '2015/07/30', '2015/11/25');

  iCount : Integer = 45;


// =====================================================================

procedure TfrmQuestion3.bmbCloseClick(Sender: TObject);
begin
  Application.Terminate;
end;

procedure TfrmQuestion3.btnDisplayClick(Sender: TObject);
begin
   SortArrays;
   Display;
end;

//=============================================================================

procedure TfrmQuestion3.btnRemoveDuplicatesClick(Sender: TObject);
begin
    RemoveDuplicates;
    SortArrays;
    Display;
end;

//=============================================================================

Procedure TfrmQuestion3.Display;
var
  sEvent, sMonth: String;
  iMonth,iTestMonth, A : Integer;

begin
  redQ3.Clear;
  redQ3.Paragraph.TabCount := 1;
  redQ3.Paragraph.Tab[0] := 150;

  sMonth := edtMonth.Text;
  if sMonth <> '' then
   begin
    iMonth := StrToInt(sMonth);
    if NOT(iMonth IN [1 .. 12]) then
      begin
         MessageDlg('Please enter a number in the range of 1 to 12',
                 mtError, [mbOk], 0);
         edtMonth.SetFocus;
         Exit;
      end;
```

```
    end
    else
       sMonth := 'All';

   // Display for selected event

   sEvent := cboEvent.text;
   redQ3.Lines.Add('List of ' + LowerCase(sEvent) + ' events for month: ' +
sMonth + #13);
   redQ3.Lines.Add('EVENTS ' + #9 + 'DATES');
    for A := 1 to iCount do        // or  to high(arrEvents) do
      begin
        iTestMonth := StrToInt(copy(arrDates[A],6,2));
        if ((iTestMonth = iMonth) OR (sMonth = 'All')) AND ((sEvent =
arrEvents[A]) OR (sEvent = 'All')) then
              redQ3.Lines.Add(arrEvents[A]+ #9 + arrDates[A]);
      end;

end;

//=============================================================================

Procedure TfrmQuestion3.SortArrays;
var
  A, B : Integer;
  sTempDate    : String;
  sTempEvent   : String;

begin
  // Sort arrays: arrDates & arrEvents

  for A := 1 to iCount - 1 do
    for B := A + 1 to iCount do
      if arrDates[B] < arrDates[A] then
       begin
        sTempDate := arrDates[A];
        arrDates[A] := arrDates[B];
        arrDates[B] := sTempDate;

        sTempEvent := arrEvents[A];
        arrEvents[A] := arrEvents[B];
        arrEvents[B] := sTempEvent;
       end;
 end;

//=============================================================================

Procedure TfrmQuestion3.RemoveDuplicates;
var
  i, j, indx : Integer;
  sEvent  : String;


begin
    // Remove duplicates
    indx := 0;
    sEvent := Uppercase(InputBox('Event', 'Enter an event', ''));
    i := 1;
    while i < iCount do
```

```
     begin
      j := i + 1;
       While (Uppercase(arrEvents[i]) = sEvent) And (arrDates[j] =
arrDates[i])
           And (j <= iCount) do
           begin
              for indx := j to iCount-1 do
               begin
                 arrdates[indx] := arrdates[indx+1];
                 arrEvents[indx] := arrevents[indx+1];
               end;
               Dec(iCount)
           end;
       inc(i);
      end;
end;

end.
```