basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

# NATIONAL SENIOR CERTIFICATE

## GRADE 12

## INFORMATION TECHNOLOGY P1

## FEBRUARY/MARCH 2012

## MEMORANDUM

**MARKS: 120**

**The memorandum consists of 27 pages.**

**GENERAL INFORMATION:**

- **Pages 2–12 contain the Delphi memoranda of possible solutions for QUESTIONS 1 to 3 in programming code.**
- **Pages 13–21 contain the Java memoranda of possible solutions for QUESTIONS 1 to 3 in programming code.**
- **Pages 22–27 contain ADDENDA A to F which includes a marking grid for each question for candidates using either one of the two programming languages.**
- **Copies of the appropriate ADDENDA should be made for each learner to be completed during the marking session.**

## SECTION A: DELPHI

## QUESTION 1: DELPHI – PROGRAMMING AND DATABASE

```
unit Question1_U_MEMO;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, DB, ADODB, Grids, DBGrids, ExtCtrls, Buttons;

type
  TfrmQuestionOne = class(TForm)
    Panel1: TPanel;
    Panel2: TPanel;
    btnA: TButton;
    btnB: TButton;
    btnC: TButton;
    btnD: TButton;
    btnE: TButton;
    btnF: TButton;
    btnG: TButton;
    bmbClose: TBitBtn;
    qryQOne: TADOQuery;
    dsrOne: TDataSource;
    grdOne: TDBGrid;

    procedure btnAClick(Sender: TObject);
    procedure btnBClick(Sender: TObject);
    procedure btnCClick(Sender: TObject);
    procedure btnDClick(Sender: TObject);
    procedure btnEClick(Sender: TObject);
    procedure btnFClick(Sender: TObject);
    procedure btnGClick(Sender: TObject);
  private
    { Private declarations }
    rq : integer;
  public
    { Public declarations }
  end;

var
  frmQuestionOne: TfrmQuestionOne;

implementation

{$R *.dfm}
```

```
procedure TfrmQuestionOne.btnAClick(Sender: TObject);       //QUESTION 1.1
begin
  qryQOne.Active := False;
  qryQOne.SQL.Text :=  'SELECT StallName, Class, NumOfGuests ✓'  +
                       'FROM tblStalls ✓'+
                       'ORDER BY NumOfGuests DESC✓';
  qryQOne.Active := True;
end;                                                                    (3)
//=========================================================================
procedure TfrmQuestionOne.btnBClick(Sender: TObject);       //QUESTION 1.2
begin
  qryQOne.Active := False;
  qryQOne.SQL.Text :=  'SELECT Teacher ✓ '+
                       'FROM tblStalls ✓' +
                       'WHERE (NumOfGuests >= 100) ✓ AND (StallID like "RC%")✓';
  qryQOne.Active := True;
end;                                                                     (4)
//=========================================================================
procedure TfrmQuestionOne.btnCClick(Sender: TObject);     //QUESTION 1.3
begin
  qryQOne.Active := False;
  qryQOne.SQL.Text :=  'SELECT StallName, NumOfGuests, ✓ ' +
                       'Round✓ (NumOfGuests * 1.25,0) ✓ AS ServingsToPrepare✓ ' +
                       'FROM tblStalls'✓;
  qryQOne.Active := True;
end;                                                                    (5)
//=========================================================================
procedure TfrmQuestionOne.btnDClick(Sender: TObject);        //QUESTION 1.4
var
   sInput : String;
begin
  sInput := INPUTBOX('Question', 'Item ? ', ''); ✓
  qryQOne.Active := False;
  qryQOne.SQL.Text :=  'SELECT format (Sum(Amount),"Currency") AS Total ✓' +
                       'FROM tblDonations ✓' +
                       'WHERE (Item Like "' + sInput + '%"✓) AND NOT Received✓ ';
  qryQOne.Active := True;
end;                                                                    (5)
//=========================================================================
procedure TfrmQuestionOne.btnEClick(Sender: TObject);        //QUESTION 1.5
begin
  qryQOne.Active := False;
  qryQOne.SQL.Text :=  'SELECT Item, Format(sum(Amount) ✓, "Currency") AS '+
                        ' AmountReceived '✓ +
                       'FROM tblStalls, tblDonations '✓ +
                       'WHERE (tblStalls.StallID = tblDonations.StallID) ✓ AND ' +
                       ' Received✓ AND (Class = "12B")✓' +
                       'GROUP BY Item'✓;
  qryQOne.Active := True;
end;                                                                    (7)
//=========================================================================
procedure TfrmQuestionOne.btnFClick(Sender: TObject);       //QUESTION 1.6
begin
  qryQOne.Active := False;
  qryQOne.SQL.Text :=  'UPDATE  tblStalls ✓' +
                       'SET NumOfGuests✓ = NumOfGuests✓ + (0.05 * NumOfGuests)✓' +
                       'WHERE Class like "%A"✓';

  qryQone.ExecSQL;
  MessageDlg('Records Processed Successfully',mtInformation,[mbok],0);
end;                                                                    (5)
//=========================================================================
```

```
procedure TfrmQuestionOne.btnGClick(Sender: TObject);        //QUESTION 1.7
begin
  qryQOne.Active := False;
  qryQOne.SQL.Text :=  'INSERT INTO tblDonations✓ ' +
                       '(StallID, Item, Amount, Received) ✓{NO AutoNumber field}' +
                       'VALUES ✓'+
                       '("HC77", "Other expenses",200, TRUE)'; ✓{All the values}

  qryQOne.ExecSQL;
  MessageDlg('Records Processed Successfully',mtInformation,[mbok],0);
end;                                                                          (4)
//=============================================================================
end.
```

**[33]**

## QUESTION 2: DELPHI – OBJECT-ORIENTED PROGRAMMING

```
unit uEventXXXX;
//=============================================================================
// Q 2.1.1                                   (5)

interface

uses
  SysUtils;

Type
   TEvent = class
     private✓
      fName    :String;
      fPerson  :String;   ✓
      fDate    :String;
      fIncome  :real;      ✓
      fExpenses:real;
     public
      constructor Create(Name:String;Person:String;Date:String;
                                                 Income,Expenses:integer );
      function toString : String;
      function calculateProfit : integer;
      function findTerm : integer;
      function constructNameString : String;
   end;

implementation

{ TEvent }

 constructor TEvent.Create(Name:String;Person:String;Date:String;
                                                 Income,Expenses:integer);
 begin
      fName   := Name;
      fPerson := Person;
      fDate   := Date;
      fIncome := Income;
      fExpenses := Expenses;

 end;

 function TEvent.toString:String;
 begin
     result := fName + #9 + fPerson + #9 + fDate + #9 +
                  FloatToStrF(fIncome, ffCurrency, 8,2)+ #9 +
                  FloatToStrF(fExpenses, ffCurrency, 8,2);
 end;


//=============================================================================
// Q 2.1.2                                   (3)

  function TEvent.calculateProfit:real; ✓
  begin
    result✓ := fIncome - fExpenses; ✓
  end;

//=============================================================================
```

Used the correct field names✓

Removed ✓ comment symbols

**Q 2.1.1**
(1)Declare private fields
(1) 3 string type fields
(1) 2 real type fields
(1) Correct field names
(1) Remove comment symbols

**Q 2.1.2**
(1) Return type real
(1) Calculation correct
(1) Return value

**// Q 2.1.3 (7)**

```
function TEvent.findTerm:integer;
 var
   iMonth   :integer;
 begin
    iMonth := StrToInt(copy(fDate,6,2)); ✓
    case iMonth of ✓          // or if statements
       1..3 : result := 1; ✓
       4..6 : result := 2; ✓
✓      7..9 : result := 3 ✓
      else
        result := 0; ✓
    end;
 end;
```

**Q 2.1.3**
(1) Copy month from fDate
(2) Use case/if-statements for testing month value
(3) Test month value for each term and return
(1) 0 if month not in any of the valid terms

```
//=============================================================================
```

**// Q 2.1.4 (10)**

```
function TEvent.constructNameString:String;
  var
    sInitials, sLastName, sFirstName :String;
    iComma, iSpace,iPlace, K :integer;
  begin
    iSpace := pos(' ',fPerson); ✓
    iComma := pos(',', fPerson); ✓
    sLastName := copy(fPerson,iSpace+1,
                              iComma -iSpace-1); ✓

    sFirstName := copy(fPerson,iComma + 1, 100); ✓
    sInitials := copy(sFirstName,1,1); ✓
    for K := 1 to length(sFirstName) do✓
     begin
       if sFirstName[K] = ' ' then✓
          sInitials := sInitials ✓+ copy(sFirstName,K+1,1); ✓
     end;
    result := sInitials + ' ' + sLastName; ✓
  end;
//=============================================================================
```

**Q 2.1.4**
(1) Get position of space
(1) Get position of comma
(1) Copy last name
(1) Copy first name
(1) Copy first initial
(1) Loop through first name
(1) Test each character for space
(1) Copy character following space
(1) Add character to initials
(1) Add last name to initials and return

## unit Question2_UXXXX;

```
unit QuestTwoXXXX_U;

interface

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

type
  TfrmQuest2 = class(TForm)
    MainMenu1: TMainMenu;
    OptionA1: TMenuItem;
    OptionC: TMenuItem;
    Quit1: TMenuItem;
    redOutput: TRichEdit;
    OptionB: TMenuItem;
    procedure Quit1Click(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure OptionA1Click(Sender: TObject);
    procedure OptionCClick(Sender: TObject);
    procedure OptionBClick(Sender: TObject);
```

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

var
  frmQuest2: TfrmQuest2;

implementation

{$R *.dfm}

uses EventXXXX;
//============================================================================
```

**Q 2.2.1 (10)**

```
var
  arrEvents :array[1..20] of TEvent;
  iCount    :integer;


procedure TfrmQuest2.FormActivate(Sender: TObject);
var
  TFile :TextFile;
  sLine, Name, sDate, sPerson, sNumber :String;
  iIncome,iExpenses :integer;
  iColon, iPlace    :integer;

begin
  redOutput.Paragraph.TabCount := 3;
  redOutput.Paragraph.Tab[0] := 100;
  redOutput.Paragraph.Tab[1] := 250;
  redOutput.Paragraph.Tab[2] := 300;

  if fileExists('Data2011.txt') <> true then✓
    begin
      ShowMessage('The text file ''Data2011.txt''
            does not exist'); ✓
      Application.Terminate;
    end;
    AssignFile(TFile, 'Data2011.txt'); ✓
    Reset(TFile); ✓
    iCount := 0; ✓
    While not eof(TFile) do✓
      begin✓
        inc(iCount); ✓
        readln(TFile, sLine); ✓
        iColon := pos (':', sLine); ✓
        Name := copy(sLine, 1, iColon -1);  }✓
        delete(sLine, 1, iColon);           }
        iPlace := pos (' on ', sLine); ✓
        sPerson := copy(sLine, 1, iPlace -1);}  ✓
        delete(sLine, 1, iPlace + 3);       }
        sDate := sLine; ✓
        readln(TFile, iIncome); ✓
        readln(TFile, iExpenses); ✓
        arrEvents[iCount] ✓ := TEvent.Create✓ (Name, sPerson, sDate,
                                               iIncome, iExpenses); ✓
      end;
     CloseFile(TFile); ✓                                    (20/2 = 10)
end;
//============================================================================
```

**Q 2.2.1 (20/2 = 10)**
(2) Test if file exists
(1) Display message if not
(1) Assign File
(1) Reset File
(1) Loop reading from the file
(1) Inside loop
(1) Increment counter
(1) Read first line
(2) Use the position of the colon to copy the name
(2) Use the position of the word 'on' to copy the name of the organiser
(1) Copy the date
(1) Read the second line to variable
(1) Read the third line to variable
(2) Create object using correct variables
(1) Assign object to array
(1) Close file outside loop

 

***// Q 2.2.2 (4/2 = 2)***

```
procedure TfrmQuest2.OptionA1Click(Sender: TObject);
var
   K :integer;
begin
   redOutput.Lines.Add('Name' + #9+'Organiser' + #9 + 'Date'+ #9 + 'Income'+
                                                        #9 + 'Expenses'); ✓
   for K := 1 to iCount do✓
     begin
       redOutput.Lines.Add(arrEvents[K] ✓.toString); ✓
     end;
end;
```

**Q 2.2.2**
(1) Display heading
(1) Inside loop
(1) Call the toString method
(1) of the object in the array

```
//=============================================================================
```

***// Q 2.2.3 (4)***

```
procedure TfrmQuest2.OptionCClick(Sender: TObject);
var
  K :integer;
begin
  redOutput.Lines.Clear;
  redOutput.Lines.Add('Organiser' + #9 + 'Profit'); ✓
  for K := 1 to iCount do✓
    begin
       redOutput.Lines.Add(arrEvents[K] ✓.constructNameString ✓+ #9 +
      FloatToStrF(arrEvents[K].calculateProfit, ✓ ffCurrency,8,2));
    end;
end;
```

**Q 2.2.3**
(1) Display heading
(1) Inside loop:
(1) Call constructNameString of the element in array to display
(1) Call the calculateProfit method of the object in the array to display

```
//=============================================================================
```

***// Q 2.2.4 (8)***

```
procedure TfrmQuest2.OptionCClick(Sender: TObject);
var
  K : integer;
  rTotal1, rTotal2, rTotal3 :real;
begin
  rTotal1 := 0;
  rTotal2 := 0;   } ✓
  rTotal3 := 0;
  redOutput.Lines.Add(' ');
  redOutput.Lines.Add('Profit per term');
  for K := 1 to iCount do✓
    begin
       case arrEvents[K].findTerm of ✓
       1: begin
            rTotal1 := rTotal1 ✓+ arrEvents[K].calculateProfit; ✓
          end;
       2: begin
            rTotal2 := rTotal2 + arrEvents[K].calculateProfit; ✓
          end;
       3: begin
            rTotal3 := rTotal3 + arrEvents[K].calculateProfit; ✓
          end;
        end; // case
    end;
    redOutput.Lines.Add('');
    redOutput.Lines.Add('Term 1             Term 2              Term 3');
    redOutput.Lines.Add(FloatToStrF(rTotal1,ffCurrency,8,2) + '       ' +
         FloatToStrF(rTotal2,ffCurrency,8,2) + '       ' +                  } ✓
         FloatToStrF(rTotal3,ffCurrency,8,2));

end;
```

**Q 2.2.4**
(1) Initialise variables
(1) Loop
(1) Call findTerm method as part of case/if
(1) For term 1 call the calculateProfit method
(1) Add value to total1
(1) Repeat code for term 2
(1) Repeat code for term 3
(1) Neatly display totals

```
procedure TfrmQuest2.Quit1Click(Sender: TObject);
begin
  Close;
end;

end.
```

**[49]**

## QUESTION 3: DELPHI – PROGRAMMING

**NOTE: This is only a sample – learners may answer this question in any way they see fit. Make use of the generalised rubric in the mark sheets for marking.**

### //QUESTION 3.1 (26)

```
unit q3_memo_u;

interface

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

type

  TfrmQuestion3 = class(TForm)
    redX: TRichEdit;
    BitBtn1: TBitBtn;
    btnGuess: TButton;
    procedure bmbGuessClick(Sender: TObject);
    procedure bmbQuitClick(Sender: TObject);
  private
    { Private declarations }
    arrUser : Array[1..5] of String;
    iPoints, iSize   : Integer;
  public
    { Public declarations }
  end;

var
  frmQuestion3: TfrmQuestion3;
  arrPoints : Array[1..20] of string =
       ('12:40','20:0','13:0','3:0','15:0','9:0', '19:50','10:0',
        '8:90','11:0','1:0', '5:30','16:0','14:100','4:0','17:0',
        '18:20','6:0','7:0','2:20');

implementation

{$R *.dfm}

procedure Tfrm Question3.bmbQuitClick(Sender: TObject);
begin
 Application.Terminate;
end;

procedure TfrmQuestion3.bmbGuessClick(Sender: TObject);
var
    a,b,c,  iChoice : integer;
    bFound       : boolean;
    sDummy       : string;
begin
   iPoints := 0; ✓
   iSize := 20;
   for a := 1 to 5 do✓
    begin
```

```
       iChoice := StrToInt(InPutBox('Treasure Hunt',
                  'Enter Choice '+ IntToStr(a), '0')); ✓
      if ( iChoice < 1 OR iChoice > 20) then✓
        begin
            arrUser[a] := IntToStr(iChoice) +
                                      ' INVALID NUMBER'; ✓
            dec(iPoints, 5); ✓ // or iPoints := iPoints – 5;
       end
      else
       begin
         bFound := False; ✓
         b := 0; ✓
         while NOT bFound AND (b <=iSize) do✓
          begin
           inc(b, 1); ✓ // or inc(b); or b = b + 1;
           if✓pos(IntToStr(iChoice)✓+':'✓,arrPoints[b])=1✓
            then
              begin
                bFound := true; ✓
                sDummy :=  arrPoints[b];
                arrUser[a] := sDummy;  } ✓
                delete(sDummy, 1, pos ('*',sDummy)); ✓
                inc(iPoints✓, StrToInt(sDummy) ✓);

                for c := b to iSize-1✓ do
                    arrPoints[c] ✓ := arrPoints[c+1] ✓;

                dec(iSize, 1) ✓;// or dec(iSize); or iSize = iSize + 1;
              end;
        end;//while
       if NOT bFound✓
        then
         begin
            arrUser[a] := IntToStr(iChoice) + ' ALREADY SELECTED'; ✓
            dec(iPoints, 5); ✓ // or iPoints := iPoints – 5;
         end;
      end; // outer if
    end;//for
//============================================================================
```

**Q 3.1**
(1) Initialise total points to 0
(1) Run a loop to 5 for input
(1) Input choice
(3) Test input in range
(1) Initialise Boolean variable
(1) Initialise counter
(1) Correct conditions in while loop
(1) Find position of the ':'
(1) Copy the number {*pos()=1*}
(1) Convert the number to integer
(1) Copy the points
(1) Check if choice = number {*if*}
(1) Set Boolean value to true
(1) Convert points to integer
(1) Add points to total points
(1) Store user choice in new array
(4) Code to delete the entry selected by the user
(1) Increment counter of while loop.
(3) Checking if a choice is repeated, deducting 5 from total points.

**//QUESTION 3.2 (12)**

**Q 3.2 (continues on next page)**
(2) Display contents of original array
(2) Display all 5 user choices

```
    //Display results
    redOutput.Clear;
    redOutput.Lines.Add('Numbers not selected');
    redOutput.Lines.Add('====================');
    for a := 1 to iSize✓ do
     redOutput.Lines.Add(arrPoints[a]); ✓

    //Determine and display the Prize
    redOutput.Lines.Add(' ');
    redOutput.Lines.Add('Participants'' choices');
    redOutput.Lines.Add('====================');
    for a := 1 to 5 do✓
     redOutput.Lines.Add(arrUser[a]); ✓
```

```
    if iPoints < 0 then
       iPoints := 0; ✓
    redOutput.Lines.Add('Points: ' + IntToStr(iPoints)); ✓

    case ✓iPoints of
      0        : redOutput.Lines.Add✓ ('NO Prize✓ ');
      1..100   : redOutput.Lines.Add('Prize: Teddy Bear'); ✓
      101..200 : redOutput.Lines.Add('Prize: Fishing Rod'); ✓
      else
        redOutput.Lines.Add('Prize: Gym Membership'); ✓
    end;

end;

end.
```

**Q 3.2 (continued)**
(1) Change total points to 0 if it is a negative value
(1) Display total points
(1) If total points = 0 - NO prize.
(1) Display prize
(1) Check if total points in 0–100 range and display correct prize
(1) Check if total points in 101–200 range and display correct prize
(1) Check if total points > 200 and display correct prize
(1) Case or if-statements

**[38]**

**END OF SECTION A: DELPHI** **TOTAL SECTION A: 120**

## SECTION B: JAVA

## QUESTION 1: JAVA – PROGRAMMING AND DATABASE

```
import java.io.*;
import java.sql.*;
import javax.swing.*;
import java.util.Scanner;

public class TestFundsXXXX
{
   public static void main (String[] args) throws SQLException,IOException
   {
    BufferedReader inKb = new BufferedReader (new InputStreamReader (System.in));
    Funds DB = new Funds();
    System.out.println();
    char choice = ' ';
     do
      {
         System.out.println("       MENU");
         System.out.println();
         System.out.println("    Option A");
         System.out.println("    Option B");
         System.out.println("    Option C");
         System.out.println("    Option D");
         System.out.println("    Option E");
         System.out.println("    Option F");
         System.out.println("    Option G");
         System.out.println();
         System.out.println("    Q - QUIT");
         System.out.println(" ");
         System.out.print("    Your Choice? ");
         choice = inKb.readLine().toUpperCase().charAt(0);
         System.out.println(" ");
         String sql = "";
         switch(choice)
         {
           case 'A':                                          //QUESTION 1.1
              {
            sql = "SELECT StallName,Class,NumOfGuests✓ FROM
                          tblStalls ✓ ORDER BY NumOfGuests DESC✓";
                 DB.query(sql);
                 break;
              }                                                          (3)
//=============================================================================
           case 'B':                                          //QUESTION 1.2
              {
            sql = "SELECT Teacher✓ FROM tblStalls✓ WHERE
                   NumOfGuests >= 100✓ AND StallID like 'RC%'✓";
                 DB.query(sql);
                 break;
              }                                                          (4)
//=============================================================================
           case 'C':                                          //QUESTION 1.3
              {
            sql = "SELECT  StallName,NumOfGuests✓,round✓(NumOfGuests * 1.25,0) ✓
                   AS ServingsToPrepare ✓ FROM tblStalls✓";
                 DB.query(sql);
                 break;
              }                                                          (5)
//=============================================================================
```

```
                case 'D':                                          //QUESTION 1.4
                   {
                      System.out.println("Enter an item");
                      String input = kb.nextLine();✓
                      sql = "SELECT Sum (Amount) AS Total✓ FROM tblDonations✓ WHERE
                             Item LIKE '"+input+"'✓ AND Received = false"✓;
                     DB.query(sql);
                     break;
                   }                                                          (5)
//=============================================================================
                case 'E':                                          //QUESTION 1.5
                   {
                     sql = "SELECT Item, Sum (Amount) ✓ AS AmountReceived ✓
                            FROM tblStalls, tblDonations✓ WHERE tblStalls.StallID =
                            tblDonations.StallID✓ AND
                            Class LIKE '12B' ✓AND Received✓ GROUP BY Item"✓;
                      DB.query(sql);

                     break;
                   }                                                          (7)
//=============================================================================
                case 'F':                                          //QUESTION 1.6
                   {
                    sql = "UPDATE tblStalls ✓SET NumOfGuests✓ =
                            NumOfGuests+NumOfGuests✓ * 0.05 ✓  // or 5/100
                            WHERE Class LIKE '%A'";✓
                     DB.query(sql);

                     break;
                   }                                                          (5)
//=============================================================================
               case 'G':                                           //QUESTION 1.7
                   {
                    sql = "INSERT INTO tblDonations ✓(StallID, Item, Amount,Received) ✓
                             VALUES ✓ ('HC77','Other expenses',200,true)";✓
                     DB.query(sql);
                      // No Autonumber field                      // All the values
                     break;
                   }                                                          (4)
//=============================================================================
            }
         }while (choice != 'Q');
         DB.disconnect();
         System.out.println("Done");
      }
   }
//=============================================================================
```

**[33]**

## QUESTION 2: JAVA – OBJECT-ORIENTED PROGRAMMING

```
// Q 2.1.1                                    (5)
public class EventXXXX
{
    private✓ String name;
    private String person;                    }✓
    private String date;
    private double income;                    }✓
    priv double expenses;
```

Used the correct field names✓

**Q 2.1.1**
(1) Declare private fields
(1) 3 string type fields
(1) 2 double type fields
(1) Correct field names
(1) Remove comment symbols

```
    public EventXXXX (String Name, String Person, String Date,
                      double Income, double Expenses)
    {
     name = Name;
     person = Person;
     date = Date;
     income = Income;
     expenses = Expenses;
    }
     public String toString()
    {
       String objStr = "";
       objStr = objStr + String.format("%-25s%-35s%-
       13s%5s%8.2f%5s%8.2f",name, person, date,"R",income,"R", expenses);
       return objStr;
     }
//=========================================================================
```

✓Removed comment symbols

```
   // Q 2.1.2                                 (3)
 public double✓ calculateProfit()
 {
    return ✓ income - expenses; ✓
 }
//=========================================================================
```

**Q 2.1.2**
(1) Return type double
(1) Calculation correct
(1) Return value

```
   // Q 2.1.3                                 (7)
public int findTerm()
 {
   int month = Integer.parseInt(date.substring(5,7));✓
   if ((month >= 1) &&✓ (month <= 3)) ✓
     return 1; ✓
   else
   if ((month >= 4) && (month <= 6)) ✓
     return 2;
   if ((month >= 7) && (month <= 9)) ✓
     return 3;
   else  return 0; ✓
}
//=========================================================================
```

**Q 2.1.3**
(1) Copy month from date
(2) Use switch/if-statements for testing month value
(3) Test month value for each term and return
(1) 0 if month not in any of the valid terms

**// Q 2.1.4 (10)**

```
public String constructNameString()
  {
    int space = person.indexOf(" ");✓
    int comma = person.indexOf(",");✓
    String lastname = person.substring(space+1, comma);✓
    String firstName = person.substring(comma + 1); ✓
    String initials = firstName.substring(0,1); ✓
    for (int k = 0; k < firstName.length(); k++)✓
     {
       if (firstName.charAt(k) == ' ' ) ✓
       {
           initials  = initials ✓+ firstName.charAt(k+1); ✓
        }
    }
    return initials + " " + lastname; ✓
  }
//===========================================================================
}
```

**Q 2.1.4**
(1) Get position of space
(1) Get position of comma
(1) Copy last name
(1) Copy first name
(1) Copy first initial
(1) Loop stepping through first name
(1) Test each character for space
(1) Copy character following space
(1) Add character to initials
(1) Add last name to initials and return

**TestQuestion2XXXX.java**

```
import java.io.*;
import java.util.*;
import javax.swing.*;

public class TestQuestion2XXXX
{
   EventXXXX arrEvents[] = new EventXXXX[20];
   int count  = 0;

//=================================================
```

**// Q 2.2.1 (10)**

```
public TestEventXXXX()
 {
   try ✓
   {
     Scanner sc = new Scanner (new FileReader
                              ("Data2011.txt"));✓✓
     while (sc.hasNext())✓
     {✓
       String line = sc.nextLine();✓
       int colon = line.indexOf(":");✓
       String name = line.substring(0,colon); ✓
       int space = line.indexOf(" on");✓
       String person = line.substring(colon + 1,space); ✓
       int slash = line.indexOf("/");
       String date = line.substring(space + 4, slash + 6);✓
       line = sc.nextLine();
       double income = Double.parseDouble(line); ✓
       line = sc.nextLine();
       double expenses = Double.parseDouble(line); ✓
       arrEvents[count] ✓ = new EventXXXX✓ (name,person,date,income,expenses); ✓
       count++;✓

      }//while
      sc.close();✓
    }//try
```

**Q 2.2.1 (20/2 = 10)**
(2) Test if file exists
(1) Display message if not
(1) Open file
(1) To read from
(1) Loop reading from the file
(1) Inside loop
(1) Increment counter
(1) Read first line
(2) Use the position of the colon to copy the name
(2) Use the position of the word 'on' to extract the name of the organiser
(1) Copy the date
(1) Read the second line to variable
(1) Read the third line to variable
(2) Create object using correct variables
(1) Assign object to array
(1) Close file outside loop

```
    catch(FileNotFoundException e) ✓
    {
       System.out.println("Error:"+e.getMessage());✓
    }//catch

   }//constr                                                    (20/2 = 10)

//============================================================================
    // Q 2.2.2                              (4/2=2)
public void optionA()
{
  System.out.printf( "%-25s%-32s%-23s%-11s%-11s\n", "Name", "Organiser", "Date of
event", "Income", "Expenses");✓


   for (int k = 0 ; k < count ; k++)✓
    {
      System.out.println (arrEvents[k] ✓.toString()✓);
    }

 }//optionA
//============================================================================
    // Q 2.2.3                              (4)
  public void optionB()
   {
    System.out.printf( "%-22s%8s\n", "Name", "Profit");✓
    for (int k = 0 ; k < count ; k++)
    {
     System.out.printf ("%20s%2s%8.2f\n",arrEvents[k]✓
             .constructNameString()✓,"R
                ",arrEvents[k].calculateProfit()✓);
    }
  }//optionB
//============================================================================
// Q 2.2.4                              (8)
  public void optionC()
  {
    double total1 = 0;
    double total2 = 0;   } ✓  // Initialise variables
    double total3 = 0;
    System.out.println(" ");
    for (int k = 0 ; k < count ; k++)✓
    {


      if (arrEvents[k].findTerm()==1) ✓
             total1 = total1✓ + arrEvents[k].calculateProfit();✓
      if (arrEvents[k].findTerm()==2)
             total2 = total2 + arrEvents[k].calculateProfit();✓
      if (arrEvents[k].findTerm()==3)
             total3 = total3 + arrEvents[k].calculateProfit();✓
    }
    System.out.println("Profit per term");
    System.out.printf("%-17s%-17s%-7s\n", "Term 1",   "Term 2",   "Term 3");
    System.out.printf("%2s%-15.2f%2s%-15.2f%2s%-15.2f\n","R ",total1,"R      } ✓
                          ",total2,"R ",total3);
 }//optionC
```

**Q 2.2.2**
(1) Display heading
(1) Inside loop
(1) Call the toString method
(1) of the object in the array

**Q 2.2.3**
(1) Display heading
(1) Inside loop:
(1) Call constructNameString of the element in array to display
(1) Call the calculateProfit method of the object in the array to display

**Q 2.2.4**
(1) Initialise variables
(1) Loop
(1) Call findTerm method as part of switch/if
(1) For term 1 call the calculateProfit method
(1) Add value to total1
(1) Repeat code for term 2
(1) Repeat code for term 3
(1) Neatly display totals

```
  public static void main (String[] args) throws IOException
  {
   TestQuestion2XXXX obj = new TestQuestion2XXXX();
   System.out.println ("\f");
   BufferedReader inKb = new BufferedReader (new InputStreamReader (System.in));
   char ch = ' ';
   while (ch != 'Q')
    {
      System.out.println ("        Menu ");
      System.out.println (" ");
      System.out.println ("    A - Option A");
      System.out.println ("    B - Option B");
      System.out.println ("    C - Option C");
      System.out.println ("    Q - QUIT");

      System.out.println (" ");
      System.out.print ("    Your choice? :");
      ch = inKb.readLine ().toUpperCase ().charAt (0);
      switch (ch)
      {

         case 'A':
          {
              obj.optionA();
              break;
          }
         case 'B':
          {
              obj.optionB();
              break;
           }
         case 'C':
           {
               obj.optionC();
               break;
           }

          case 'Q':
           {
               System.exit (0);
           }

            }//switch
         }//while


      }//main method
   }//class

//============================================================================
```

**[49]**

## QUESTION 3: JAVA – PROGRAMMING

**NOTE: This is only a sample – learners may answer this question in any way they see fit. Make use of the generalised rubric in the mark sheets for marking.**

### //Q 3.1 (26)

**TestQuestion3.java**

```
import java.util.*;
import java.io.*;
public class TestQuestion3
{
 Scanner kb = new Scanner (System.in);

 String[]arrPoints={"12:40","20:0","13:0","3:0","15:0","9:0",
                    "19:50","10:0","8:90","11:0","1:0","5:30",
                    "16:0","14:100","4:0","17:0","18:20","6:0",
                    "7:0","2:20"};
      String [] user = new String[5];

      int size = 20;
      int points  = 0 ; ✓

       public void processChoices()
      {
         for (int i = 0; i< 5; i++)✓
         {
            System.out.println("Enter choice "+(i+1));
            int choice = kb.nextInt();✓
          if ((choice < 1) || (choice > 20)) ✓
            {
               points = points - 5; ✓
               user[i]=" " +choice + " INVALID NUMBER" ;✓
            }
        else
      {
           boolean found = false; ✓
          int j =0; ✓
           while(!found && j < size) ✓
           {
          int psnColon = arrPoints[j].indexOf(":");✓
          String strNum = arrPoints[j].substring(0,psnColon); ✓
          int number = Integer.parseInt(strNum); ✓

              if (choice == number) ✓
              {
                 found = true; ✓
                 String pnts = arrPoints[j].substring(psnColon+1); } ✓
                 int numPnts = Integer.parseInt(pnts);
                 points = points + numPnts; ✓
                 user[i] = arrPoints[j]; ✓
                 deleteChoice(j); ✓
              }//if

              j++;       ✓
           }//while
```

**Q 3.1**
(1) Initialise total points to 0
(1) Run a loop to 5 for input
(1) Input choice
(3) Test input in range
(1) Initialise Boolean variable
(1) Initialise counter
(1) Correct conditions in while loop
(1) Find position of the ':'
(1) Copy the number
(1) Convert the number to integer
(1) Copy the points
(1) Check if choice = number {*if*}
(1) Set Boolean value to true
(1) Convert points to integer
(1) Add points to total points
(1) Store user choice in new array
(4) Code to delete the entry selected by the user
(1) Increment counter of while loop.
(3) Check if a choice is repeated, deduct 5 from total points.

```
            if (!found) ✓
            {
               points = points - 5; ✓
               user[i]=" " +choice+" ALREADY SELECTED ";✓
            }
       } // outer if
        }//for

        }//processChoices

        public void deleteChoice(int posn) ✓
       {
          for (int i= posn;i<size-1;i++)✓
          {
             arrPoints[i] = arrPoints[i+1]; ✓
          }
          size--;✓
       }//deleteChoice
//============================================================================
```

**// Q 3.2 (12)**

```
       public String getPrize()
      {
         if (points <0)
         {
            points = 0;  ✓
         }

         String prize = "No prize";✓
         if (points >0 && points <=100)
         {

            prize = "Teddy Bear";✓
         }
         else if (points > 100 && points <= 200)
         {
            prize = "Fishing Rod";✓
         }

         else if (points > 200)
         {
            prize = "Gym Membership";✓
         }
         return prize; ✓
      }
public void display()
{
         System.out.println("\nNumbers not selected\n===");
         for (int i = 0;i< size;i++)✓
         {
            System.out.println(arrPoints[i]); ✓
         }
         System.out.println("\nUser choices\n=============" );

         for (int i = 0;i< 5;i++)✓
         {
            System.out.println(user[i]); ✓
         }

         System.out.println("Points:"+points); ✓

         System.out.println("Prize:"+getPrize());✓
         }
```

**Q 3.2**
(1) Change points to 0 if points are negative
(1) Initialise String prize
(1) Check if points in 0–100 range and assign correct prize
(1) Check if points in 101–200 range and assign correct prize
(1) Check if points > 200 and assign correct prize
(1) Return prize
(2) Display contents of original array after user choices
(2) Display all 5 user choices
(1) Display total points
(1) Display prize

```
     public static void main (String [] args)
      {
      TestQuestion3 obj = new TestQuestion3();
      obj.processChoices();


      obj.display();
      }//main method
      }//class
//============================================================================
```

**[38]**

**END OF SECTION B: JAVA** **TOTAL SECTION B: 120**
**GRAND TOTAL: 120**

## ADDENDUM A

## QUESTION 1: DELPHI – PROGRAMMING AND DATABASE

| CENTRE NUMBER: …… | EXAMINATION NUMBER: …… |
|---|---|

**QUESTION 1: DELPHI – MARKING GRID**

| QUESTION | ASPECT | MAX. MARKS | LEARNER'S MARKS |
|---|---|---|---|
| **1.1** | SELECT StallName, Class, NumOfGuests ✓<br>FROM tblStalls ✓ORDER BY NumOfGuests DESC✓ | **3** | |
| **1.2** | SELECT Teacher✓ FROM tblStalls✓ WHERE NumOfGuests >= 100✓ AND StallID like "RC%"✓ | **4** | |
| **1.3** | SELECT StallName,NumOfGuests✓,<br>round✓(NumOfGuests * 1.25,0) ✓AS ServingsToPrepare✓<br>FROM tblStalls✓ | **5** | |
| **1.4** | sInput := INPUTBOX('QUESTION', 'Item? ', ' '); ✓<br><br>SELECT format (Sum(Amount),"Currency") AS Total ✓<br>FROM tblDonations ✓ WHERE (Item Like "' + sInput + '%"✓)<br>AND Received = false ✓ ***//* or** NOT Received | **5** | |
| **1.5** | SELECT Item, Format(sum(Amount) ✓, "Currency") AS AmountsReceived '✓ FROM tblStalls, tblDonations ✓<br>WHERE (tblStalls.StallID = tblDonations.StallID) ✓ AND Received✓ AND (Class = "12B")✓ GROUP BY Item✓ | **7** | |
| **1.6** | UPDATE tblStalls ✓ SET NumOfGuests✓ =<br>NumOfGuests✓+ (0.05 * NumOfGuests) ✓ // OR 5/100<br>WHERE Class like "%A"✓ | **5** | |
| **1.7** | INSERT INTO tblDonations✓(StallID, Item, Amount, Received) ✓**{NO AutoNumber field}** VALUES ✓("HC77", "Other expenses", 200, TRUE)✓**{all the values}** | **4** | |
| | **TOTAL:** | **33** | |

## ADDENDUM B

## QUESTION 2: DELPHI – OBJECT-ORIENTED PROGRAMMING

| CENTRE NUMBER: …… | EXAMINATION NUMBER: …… |
|---|---|

**QUESTION 2: DELPHI – MARKING GRID**

| QUESTION | ASPECT | MAX. MARKS | LEARNER'S MARKS |
|---|---|---|---|
| **2.1** | | | |
| **2.1.1** | Declare private (1) fields, 3 string types(1) and 2 real types(1), correct fields names(1), remove comment symbols(1) | **5** | |
| **2.1.2** | **calculateProfit:**<br>return type real(1) calculation correct(1), return value(1) | **3** | |
| **2.1.3** | **findTerm:**<br>Copy month from fDate(1), use case / if-statements for term test lower and upper limit (2) Test month value for each term and return (3), 0 if month not in any of the valid terms(1) | **7** | |
| **2.1.4** | **constructNameString:**<br>Get position space, (1) Get position comma(1), Copy last name(1), copy first name(1), copy first initial(1) loop through first name (1), Test each character for space(1), copy character following space(1), add character to initials(1), add last name to initials and return(1) | **10** | |
| **2.2** | | | |
| **2.2.1** | Test if file exists(2), display Message if not(1)<br>Assign file(1) and reset file(1), loop reading lines from file(1), inside loop(1), increment counter(1), read first line(1), use the position of the colon to copy the name(2), use the position of the word ' on' to copy the name of the organiser(2), copy the date(1), read the second line to variable(1), read the third line to variable(1), create object(1), using correct variables(1), assign object to array(1), close file outside loop(1) | **20/2=10** | |
| **2.2.2** | Display heading (1), inside loop(1), call the toString method (1), of the object in the array(1) | **4/2=2** | |
| **2.2.3** | Display heading (1), inside loop (1), call constructNameString of the element in array(1), call the calculateProfit method of the object in the array(1) | **4** | |
| **2.2.4** | Initialise variables(1), Loop(1),call findTerm method(1), as part of case / if(1), for term 1 call the calculateProfit method(1), add value to total1(1), repeat code for term2(1), repeat code for term3(1),<br>neatly display totals (1) | **8** | |
| | **TOTAL:** | **49** | |

## ADDENDUM C

## QUESTION 3: DELPHI – PROGRAMMING

| CENTRE NUMBER: …….. | EXAMINATION NUMBER: …….. | | |
|---|---|---|---|
| **QUESTION 3: DELPHI – MARKING GRID** | | | |
| **QUESTION** | **ASPECT** | **MAX. MARKS** | **LEARNER'S MARKS** |
| **3.1** | (1) Initialise points to 0.<br>(1) Run a loop to 5 for input<br>(1) Input choice<br>(3) Test input for valid range<br>If valid, loop to look up the input number<br>(1) Initialise Boolean variable<br>(1) Initialise counter<br>(1) Correct conditions in while loop<br>(1) Find position of the ":"<br>(1) Copy the number {*pos()=1*}<br>(1) Convert the number to integer<br>(1) Copy the points<br>(1) Check if choice = number {*if*}<br>(1) Set boolean value to true<br>(1) Convert points to integer<br>(1) Add points to total<br>(1) Store user choice in new array<br>(4) Write code to delete the entry selected by the user<br>(1) Increment loop value in the while loop.<br>(3) Checking if a choice is repeated, deducting 5 from the total points. | **26** | |
| **3.2** | (2) Display contents of original array<br>(2) Display all 5 user choices<br>(1) Change points to 0 if points is negative<br>(1) Display total points<br>(1) If points = 0 > NO prize.<br>(1) Display prize<br>(1) Check if points in 0 – 100 range and display correct prize<br>(1) Check if points in 101 – 200 range and display correct prize<br>(1) Check if points > 200 and display correct prize<br>(1) case or if-statements | **12** | |
| | **TOTAL:** | **38** | |

## ADDENDUM D

## QUESTION 1: JAVA – PROGRAMMING AND DATABASE

| CENTRE NUMBER: …… | EXAMINATION NUMBER: …… |
|---|---|

**QUESTION 1: JAVA – MARKING GRID**

| QUESTION | ASPECT | MAX. MARKS | LEARNER'S MARKS |
|---|---|---|---|
| **1.1** | SELECT StallName,Class,NumOfGuests✓ FROM tblStalls ✓ ORDER BY NumOfGuests DESC✓ | **3** | |
| **1.2** | SELECT Teacher ✓ FROM tblStalls✓ WHERE NumOfGuests >= 100✓ AND StallID like 'RC%'✓ | **4** | |
| **1.3** | SELECT StallName,NumOfGuests✓,round✓(NumOfGuests * 1.25,0) ✓ AS ServingsToPrepare ✓ FROM tblStalls✓ | **5** | |
| **1.4** | System.out.println("Enter an item");<br>String input = kb.nextLine();✓<br><br>SELECT Sum (Amount) AS Total✓ FROM tblDonations✓ WHERE Item Like '"+ input +"'✓ AND Received = false"✓ ***//* or** NOT Received | **5** | |
| **1.5** | SELECT Item, Sum(Amount) ✓ AS AmountReceived ✓ FROM tblStalls, tblDonations✓ WHERE tblStalls.StallID = tblDonations.StallID✓ AND Class Like '12B' ✓AND Received ✓ GROUP BY Item✓ | **7** | |
| **1.6** | UPDATE tblStalls ✓ SET NumOfGuests✓ = NumOfGuests✓+ NumOfGuests * 0.05 ✓ ***//*** OR 5/100 WHERE Class like '%A'✓ | **5** | |
| **1.7** | INSERT INTO tblDonations ✓<br>(StallID, Item, Amount,Received) ✓ ***//*No Autonumber field**<br>VALUES ✓ ('HC77', 'Other expenses', 200, true)✓ ***//* All the values** | **4** | |
| | **TOTAL:** | **33** | |

## ADDENDUM E

## QUESTION 2: JAVA – OBJECT-ORIENTED PROGRAMMING

| CENTRE NUMBER:…… | EXAMINATION NUMBER: …… |
|---|---|

**QUESTION 2: JAVA – MARKING GRID**

| QUESTION | ASPECT | MAX. MARKS | LEARNER'S MARKS |
|---|---|---|---|
| **2.1** | | | |
| **2.1.1** | Declare private (1) fields, 3 string type fields (1) and 2 double type fields(1), correct field names(1), remove comment symbols(1) | **5** | |
| **2.1.2** | **calculateProfit:**<br>return type double(1) calculation correct(1), return value (1) | **3** | |
| **2.1.3** | **findTerm:**<br>Copy month from date(1), use switch / if-statements for term 1 test lower and upper limit (2) Test month value for each term and return (3), 0 if month not in any of the valid terms(1) | **7** | |
| **2.1.4** | **constructNameString:**<br>Get position space, (1) Get position comma(1), Copy last name(1), copy first name(1), copy first initial(1) loop through first name (1), Test each character for space(1), copy character following space(1), add character to initials(1), add last name to initials and return(1) | **10** | |
| **2.2** | | | |
| **2.2.1** | Test if file exists(2), display message if not(1)<br>Open file(1) to read from(1), loop reading lines from file(1), inside loop(1), increment counter(1), read first line(1), use the position of the colon to copy the name(2), use the position of the word ' on' to copy the name of the organiser(2), copy the date(1), read the second line to variable(1), read the third line to variable(1), create object(1) using correct variables(1), assign object to array(1), close file outside loop(1) | **20/2=10** | |
| **2.2.2** | Display heading(1), Inside loop(1)<br>Call the toString method (1), of the object in the array(1) | **4/2=2** | |
| **2.2.3** | Display heading(1), Inside loop(1), call constructNameString of the element in array(1), call the calculateProfit method of the object in the array(1) | **4** | |
| **2.2.4** | Initialise variables(1), Loop(1),call findTerm method(1), as part of switch / if(1), for term 1 call the calculateProfit method(1), add value to total1(1), repeat code for term2(1), repeat code for term3(1),<br>neatly display totals (1) | **8** | |
| | **TOTAL:** | **49** | |

## ADDENDUM F

## QUESTION 3: JAVA – PROGRAMMING

| CENTRE NUMBER: …… | EXAMINATION NUMBER: …… |
|---|---|

**QUESTION 3: JAVA – MARKING GRID**

| QUESTION | ASPECT | MAX. MARKS | LEARNER'S MARKS |
|---|---|---|---|
| **3.1** | (1) Initialise points to 0.<br>(1) Run a loop to 5 for input<br>(1) Input choice<br>(3) Test input for valid range<br>If valid, loop to look up the choice<br>(1) Initialise boolean variable<br>(1) Initialise counter<br>(1) Correct conditions in loop<br>(1) Find position of the ":"<br>(1) Copy the number<br>(1) Convert the number to int<br>(1) Copy the points<br>(1) Check if choice = number<br>(1) Set boolean value to true<br>(1) Convert points to int<br>(1) Add points to total<br>(1) Store user choice in new array<br>(4) Write code to delete the entry selected by the user<br>(1) Increment value in the while loop.<br>(3) Check if a choice is repeated, deduct 5 from the total points. | **26** | |
| **3.2** | (1) Change points to 0 if points is negative<br>(1) Initialise String prize.<br>(1) Check if points in 0 – 100 range and assign correct prize<br>(1) Check if points in 101 – 200 range and assign correct prize<br>(1) Check if points > 200 and assign correct prize<br>(1) return prize<br>(2) Display contents of original array after user choices.<br>(2) Display all 5 user choices<br>(1) Display total points<br>(1) Display prize | **12** | |
| | **TOTAL:** | **38** | |