education

Department:
Education
REPUBLIC OF SOUTH AFRICA

SENIOR CERTIFICATE EXAMINATION

GRADE 12

INFORMATION TECHNOLOGY P1

FEBRUARY/MARCH 2009

MEMORANDUM

MARKS: 120

The memorandum consists of 37 pages.

General information:

- **Pages 2 – 13 contain the Delphi memoranda of possible solutions for QUESTIONS 1 to 3 in programming code.**
- **Pages 13 – 27 contain the Java memoranda of possible solutions for QUESTIONS 1 to 3 in programming code.**
- **Pages 28 – 37 contain Addenda A to G which includes a cover sheet as well as a marking grid for each question for candidates using either one of the two programming languages.**
- **Copies should be made for each learner to be completed during the marking session.**

SECTION A: DELPHI

QUESTION 1: PROGRAMMING AND DATABASE

```
unit NewsPaperU;

interface

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

type
  TfrmNewspaper = class(TForm)
    Panel1: TPanel;
    dbgAdverts: TDBGrid;
    Panel2: TPanel;
    btnAllAdverts: TButton;
    btnCatMonth: TButton;
    btnUpdateAdverts: TButton;
    btnDisplayClients: TButton;
    btnCountCellphones: TButton;
    btnCatNotPaid: TButton;
    BitBtn1: TBitBtn;
    btnCalcCosts: TButton;
    qryAdverts: TADOQuery;
    DataSource1: TDataSource;
    procedure btnDisplayClientsClick(Sender: TObject);
    procedure btnAllAdvertsClick(Sender: TObject);
    procedure btnCatMonthClick(Sender: TObject);
    procedure btnShortLongAdvertsClick(Sender: TObject);
    procedure btnUpdateAdvertsClick(Sender: TObject);
    procedure btnCatNotPaidClick(Sender: TObject);
    procedure btnCalcCostsClick(Sender: TObject);
    procedure btnCountCellphonesClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmNewspaper: TfrmNewspaper;
```

```
implementation

{$R *.dfm}
                                                       // Question 1.1
procedure TfrmNewspaper.btnDisplayClientsClick(Sender: TObject);
begin
  qryAdverts.Active := False;    ✓         ✓               ✓
  qryAdverts.SQL.Text :=  'SELECT * FROM ClientsTb ORDER BY ClientName' ;
  qryAdverts.Active := true;
end;                                                                (3)
//-------------------------------------------------------------------------
                                                       //Question 1.2
procedure TfrmNewspaper.btnAllAdvertsClick(Sender: TObject);
begin
  qryAdverts.Active := False;              //all the correct fields ✓
  qryAdverts.SQL.Text :=  'SELECT  AdvertID, DatePlaced, WeeksRunning,
           NumWords, Paid,Category, ClientName FROM AdvertsTb,
           ClientsTb ✓WHERE AdvertsTB.ClientID = ClientsTB.ClientID'; ✓

  qryAdverts.Active := true;
end;                                                                (3)
//-------------------------------------------------------------------------
                                                       // Question 1.3
procedure TfrmNewspaper.btnCatMonthClick(Sender: TObject);
var
  monthNum :string;
  cat      :string;
begin
  cat := InputBox('Enter the category ', '', '');      //input ✓
  monthNum := InputBox('Enter the number of the month', '', '');

  qryAdverts.Active := False;
                                                 ✓
  qryAdverts.SQL.Text := 'SELECT  AdvertID, DatePlaced, WeeksRunning ' +
                         'FROM AdvertsTb ' + ✓
                         'WHERE Category = "' + cat +'" ✓ AND
                           MONTH(DatePlaced) ✓ = "' + monthNum +'"  '; ✓
  qryAdverts.ExecSQL;
  qryAdverts.Active := true;

end;                                                                (6)
//-------------------------------------------------------------------------
                                                       // Question 1.4
procedure TfrmNewspaper.btnShortLongAdvertsClick(Sender: TObject);
begin
  qryAdverts.Active := False;                                     ✓
  qryAdverts.SQL.Text:='SELECT AdvertID, ✓NumWords AS [Number of Words],
               ClientName AS [Client Name] ✓FROM AdvertsTb, ClientsTb '+
                'WHERE AdvertsTB.ClientID = ClientsTB.ClientID✓ AND '+
                         '(NumWords > 30 OR✓ NumWords < 15)' ; ✓
  qryAdverts.Active := true;

end;                                                                (6)
//-------------------------------------------------------------------------
                                                       // Question 1.5
procedure TfrmNewspaper.btnUpdateAdvertsClick(Sender: TObject);
```

```
var
  sAdvert :string;
  sNumber :string;
begin                                        // input ✓
  sAdvert := InputBox('Enter the ID of the Advert to be updated','','') ;
  sNumber := InputBox('Enter the number of words to be added?','','');

  qryAdverts.SQL.Text := 'UPDATE AdvertsTb SET ✓ numWords = numWords ✓+
               "'+ sNumber +'"✓ WHERE AdvertID = "' +sAdvert +'"  '; ✓
  qryAdverts.ExecSQL;
  qryAdverts.SQL.Text := 'SELECT AdvertID, numWords AS Words FROM
                                                      AdvertsTb'; ✓
  qryAdverts.Active := true;

end;                                                            (6)
//-------------------------------------------------------------------
                                                      // Question 1.6
procedure TfrmNewspaper.btnCatNotPaidClick(Sender: TObject);
var
  sLetter :string;
begin
  qryAdverts.Active := False;
  sLetter := InputBox('Enter a letter of the alphabet','', ''); ✓
  qryAdverts.SQL.Text := 'SELECT AdvertID, Category, ClientName AS [Name
                            of Client],Paid AS [Has Paid]'✓
                  +'FROM ClientsTb, AdvertsTb '
                  +'WHERE ClientsTb.ClientId = AdvertsTb.ClientId ✓and '
                  + 'AdvertsTb.Category like✓ "' + sLetter + '%"✓  and
                         AdvertsTb.Paid = false'; ✓
  qryAdverts.Active := true;
end;                                                            (6)
//-------------------------------------------------------------------
                                                      // Question 1.7
procedure TfrmNewspaper.btnCalcCostsClick(Sender: TObject);
begin
  qryAdverts.Active := False;         ✓
  qryAdverts.SQL.Text := 'SELECT  AdvertID, DatePlaced, NumWords,
       WeeksRunning, Format(NumWords * WeeksRunning * 0.5, ✓
       "Currency")AS [Cost]✓ FROM AdvertsTb ✓WHERE DatePlaced > ✓
       #16/11/2008#' ; ✓
  qryAdverts.Active := true;

end;                                                            (6)
//-------------------------------------------------------------------
                                                      // Question 1.8
procedure TfrmNewspaper.btnCountCellphonesClick(Sender: TObject);

begin
  qryAdverts.Active := False;

  qryAdverts.SQL.Text := 'SELECT Count(*)✓ AS [Number of clients with
                         cellphone numbers]' ✓ +
                         'FROM ClientsTb WHERE [TelNum] ✓ Like "0%"✓ ';

  qryAdverts.Active := true;
end;                                                            (4)
//-------------------------------------------------------------------
```

```
end.
```

QUESTION 2: OBJECT-ORIENTED PROGRAMMING

```
unit Advert_Uxxxx;

interface
uses SysUtils;
Type
// Q 2.1.1    (4 / 2) = 2
       TAdvert = class (TObject) ✓
         private
            fAdvertText  :string; ✓
            fClientName  :string; ✓
         public
          constructor create(aAdvert : string; aName :string);
          function countWords:integer;
          function isLowPolluter:boolean;
          function calcCost:real;
          function toString:string;
          function getClientName:string;
          function getAdvertText:string;
        end; ✓

implementation
//================================================================================
// Q 2.1.2    (4 / 2) = 2

constructor TAdvert.create(aAdvert : string✓; aName :string✓);
begin
  fAdvertText := aAdvert; ✓
  fClientName := aName; ✓
end;

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

// Q 2.1.3    (8 / 2) = 4

function TAdvert.toString:string;
var
  Discount :string;
begin

  if isLowPolluter then✓
    Discount := 'Discount'      ⎫
  Else                          ⎬                 ✓
    Discount := 'No Discount';  ⎭
  Result✓ := 'Client: ' + fClientName✓ + #9 + ' Word Count: '✓ +
      IntToStr(countWords) ✓ + #9 + Discount ✓+ #9 + FloatToStrF(calculateCost,
                                                      ffCurrency, 6,2) ✓;
end;

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

// Q 2.1.4    (10 / 2) = 5

function tAdvert.countWords:integer; ✓
var
   iSpace, iHalfWords, iFullWords :integer;
   temp, thisWord : string;
begin
```

```
  iHalfWords := 0;
  iFullWords := 0;

  temp := fAdvertText+ ' ' ;
  iSpace := pos(' ', temp); ✓

  While iSpace <> 0 do✓
    Begin
      thisWord := copy(temp,1,iSpace-1); ✓
      if thisWord <> 'LowPolluter' then✓
      begin
        if iSpace <= 5 then✓
          inc(iHalfWords) }    ✓
        else
           inc(FullWords);

        delete(temp,1,iSpace); ✓
        iSpace := pos(' ', temp) ; ✓
      end;   // while

     Result := iFullWords + Trunc(iHalfWords /2 + 0.5); ✓
 end;

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

// Q 2.1.5    (6 / 2) = 3

 function TAdvert.isLowPolluter:boolean; ✓
 var
   iPlace :integer;
 begin
   iPlace := pos('LowPolluter'✓, fAdvertText✓);
   if iPlace > 0 then✓
     isLowPolluter := true✓
   else
     isLowPolluter := false; ✓
 end;

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

// Q 2.1.6    (4 / 2) = 2

 function TAdvert.calculateCost:real;
 var
   K  :integer;
   rTCost :real;
 begin
   rTCost := CountWords * 0.50✓;
   if isLowPolluter then✓
     rTCost := rTCost / 2; ✓
   Result := rTCost; ✓
end;
//===============================================================================
// Q 2.1.7    (2 / 2) = 1

function TAdvert.getClientName:string; ✓
begin
  Result := fClientName; ✓
end;

end.

//================================================================================
unit Quest2_1;  (Main Form)

interface

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

type
  TfrmNewsPaper = class(TForm)
    MainMenu1: TMainMenu;
    Aderts1: TMenuItem;
    ListArray: TMenuItem;
    SumIncome: TMenuItem;
    Quit1: TMenuItem;
    redOutput: TRichEdit;
    LookupClient: TMenuItem;
    procedure Quit1Click(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure ListArrayClick(Sender: TObject);

    procedure SumIncomeClick(Sender: TObject);
    procedure LookupClientClick(Sender: TObject);

  private

  public
    { Public declarations }
  end;

var
  frmNewsPaper: TfrmNewsPaper;

implementation
uses
  Advert_Uxxxx; ✓
var
//==========================================================================
// Q 2.2.1    (16 / 2 = 8)
  arrAdverts  :array[1..100] of TAdvert; ✓
  iCount      :integer;

{$R *.dfm}

procedure TfrmNewsPaper.FormActivate(Sender: TObject);
var
  iHash   :integer;
  oneLine :string;
  TextF   :TextFile;
  AdText  :string;
  ClName  :string;
begin
  AssignFile(TextF, 'Adverts.txt'); ✓
  if FileExists('Advertisements.txt') <> true then✓
  begin
    ShowMessage('File not found'); ✓
    Exit; ✓
  end;
  Reset(TextF); ✓
  iCount := 0; ✓
  While not eof(TextF) do✓
    begin
```

```
      inc(iCount); ✓
      readln(TextF, oneLine); ✓
      iHash := pos('#', oneLine); ✓
      AdText := copy(oneLine, 1, iHash - 1); ✓
      delete(oneLine, 1, iHash);
      ClName := oneLine; ✓
      arrAdverts[iCount] ✓ := TAdvert.create(AdText, ClName); ✓
    end;
    CloseFile(TextF); ✓
end;

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

// Menu Display Information   (6/2 = 3)

```
procedure TfrmNewsPaper.ListArrayClick(Sender: TObject);
var
  K : integer;
begin
  redOutput.Paragraph.TabCount := 5;
  redOutput.Paragraph.Tab[1] := 20;
  redOutput.Paragraph.Tab[2] := 100;
  redOutput.Paragraph.Tab[3] := 150;            ✓✓
  redOutput.Paragraph.Tab[4] := 200;
  redOutput.Paragraph.Tab[5] := 230;
  redOutput.Clear;
  redOutput.Lines.Add('Information on Advertisements'); ✓
  redOutput.Lines.Add(' ');
  For K := 1 to iCount do✓
  begin
    redOutput.Lines.Add(arrAdverts[K] ✓.toString); ✓
   end;
   redOutput.Lines.Add(' ');
end;

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

// Menu Summarise Income   (16 / 2) = 8

```
procedure TfrmNewsPaper.SumIncomeClick(Sender: TObject);
var
  K, halfCount, fullCount : integer;
  rTotal, halfTotal, fullTotal, rCost :real;
begin
  redOutput.Clear;
  redOutput.Lines.Add('Summary of income'); ✓
  redOutput.Lines.Add(' ');
  rTotal := 0;
  halfCount := 0;
  fullCount := 0; ✓✓
  halfTotal := 0;
  fullTotal := 0;
  For K := 1 to iCount do✓
  begin
    rCost :=  arrAdverts[K].calculateCost; ✓
    rTotal := rTotal +  rCost; ✓
    if arrAdverts[K].isLowPolluter then✓
    begin
      halfTotal := halfTotal + rCost; ✓
      halfCount := halfCount + 1; ✓
     end
    else✓
      begin
       fullTotal := fullTotal + rCost; ✓
       fullCount := fullCount + 1; ✓
      end;

   end;
   redOutput.Lines.Add(IntToStr(iCount) ✓ + ' advertisements in total worth ' +
FloatToStrF(rTotal, ffCurrency, 6, 2)); ✓
   redOutput.Lines.Add(' ');
   redOutput.Lines.Add(IntToStr(hCount) +' half-priced advertisements worth ' +
FloatToStrF(hTotal, ffCurrency, 6, 2)); ✓
   redOutput.Lines.Add(' ');
   redOutput.Lines.Add(IntToStr(fCount) +' full-priced advertisements worth ' +
FloatToStrF(fTotal, ffCurrency, 6, 2)); ✓
end;

//================================================================================
// Menu Search for client (10/2) = 5

procedure TfrmNewsPaper.LookupClientClick(Sender: TObject);
var
  sName :string;
  k :integer;
  Found:boolean;
begin
  sName := InputBox('Type in name of Client','',''); ✓
  redOutput.Clear;
  Found := false; ✓
  For K := 1 to iCount do✓
    begin
      if uppercase (sName)✓ = uppercase(arrAdverts[K].getClientName) ✓ then
        begin
          redOutput.Lines.Add(arrAdverts[k].toString✓);
          Found := true; ✓
        end;
     end;
   if not(Found) then✓
     redOutput.Lines.Add(sName ✓+ ' is not on the list'); ✓
end;
end.
//================================================================================
procedure TfrmNewsPaper.Quit1Click(Sender: TObject);
begin
 Application.Terminate;
end;
```

QUESTION 3: DELPHI PROGRAMMING

```
unit Charity_U;

interface

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

type
  TForm1 = class(TForm)
    btnGenerateAndDisplay: TButton;
    redDisplay: TRichEdit;
    BitBtn1: TBitBtn;
    procedure FormCreate(Sender: TObject);
    procedure btnGenerateAndDisplayClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

Implementation
```

Q 3.1

```
Type
     CharityArr = Array [1..5] of String;
     GoldRushArr = Array[1..4, 1..4] ✓ of String; ✓                    [2]
//================================================================================
 Var
    Charities : CharityArr;
    CharityName : CharityArr;
    GoldRushGrid : GoldRushArr;

{$R *.dfm}
```

Q 3.2

```
Procedure InitialiseGoldRush;
Var
   Ind, Ind2 : Integer;
Begin
  for ind := 1 to 4 do✓
    for ind2 := 1 to 4 do✓
      begin
        GoldRushGrid [ind, ind2] := ''; ✓
      end;
end;                                                                     [3]
//==============================================================================
procedure TForm1.FormCreate(Sender: TObject);
begin
  InitialiseGoldRush;

end;
```

Q 3.3

```
Procedure GenerateFunding;
Var
    Ind, Ind2 : Integer;
    CharNo, perc : integer;
    tmp : string;
Begin
  randomize;
  for ind := 1 to 4 do✓
    for ind2 := 1 to 4 do✓
      begin
        CharNo := random (5) + 1; ✓
        case Charno of✓
          1 : tmp := 'C';  ⎫
          2 : tmp := 'S';  ⎪
          3 : tmp := 'A';  ⎬✓✓
          4 : tmp := 'R';  ⎪
          5 : tmp := 'W';  ⎭
        end; ✓ // end of case
        perc := random (10) + 6; ✓

        tmp := tmp + inttostr (perc); ✓
        GoldRushGrid [ind, ind2] := tmp; ✓
      end;
end;                                                                    [10]

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

Q 3.4

```
procedure TForm1.btnGenerateAndDisplayClick(Sender: TObject);
Var Ind, Ind2, CInd     : Integer;
    perc, tmp           : string;
    PubTot, GovtTot, GrandTot :real;
begin
  redDisplay.Lines.Clear;
  GenerateFunding;

  redDisplay.Paragraph.TabCount := 4;
  redDisplay.Paragraph.Tab[0] := 65;
  redDisplay.Paragraph.Tab[1] := 65;
  redDisplay.Paragraph.Tab[2] := 65;
  redDisplay.Paragraph.Tab[3] := 65;

  // initialise arrays                                                      ⎫
  Charities[1]:= 'C'; CharityName[1] := 'CHOC';                              ⎪
  Charities[2]:= 'S'; CharityName[2] := 'Cansa';                             ⎬✓✓
  Charities[3]:= 'A'; CharityName[3] := 'Aids Africa';                       ⎪
  Charities[4]:= 'R'; CharityName[4] := 'Red Cross';                         ⎪
  Charities[5]:= 'W'; CharityName[5] := 'Battered Women & Children';         ⎭

  //Display Grid
  redDisplay.Lines.Add(‘Grid with random values’ + #13 + #13);
  for ind := 1 to 4 do✓
    begin
       tmp := '';
        for ind2 := 1 to 4 do✓
          begin
            tmp := tmp + GoldRushGrid[ind,ind2] + #9; ✓

          end;
          redDisplay.Lines.Add(tmp); ✓
         end;
  redDisplay.Lines.Add(' ');
  redDisplay.Lines.Add(' ');

  redDisplay.Lines.Add('Amounts received per charity' + #13); ✓
  redDisplay.Lines.Add('');
  redDisplay.Lines.Add('Total        Public       Govt        Charity' + #13); ✓

  for CInd := 1 to 5 do✓
    begin
      pubTot := 0; ✓
      GovtTot := 0; ✓
      GrandTot := 0;
      for ind := 1 to 4 do✓
        for ind2 := 1 to 4 do✓
          begin
            if GoldRushGrid[ind,ind2] [1] ✓ = Charities[CInd] ✓ then
              begin
                PubTot := PubTot + 50000; ✓
                perc := copy (GoldRushGrid[ind,ind2], 2, 2); ✓
                GovtTot := GovtTot + (50000 * strtoint (perc) / 100); ✓✓

          end;
     end;
     GrandTot := PubTot + GovtTot; ✓

redDisplay.Lines.Add(FloatToStrf(GrandTot, ffCurrency,8,2) + #9 +
FloatToStrf(GovtTot, ffCurrency,8,2) + #9 + FloatToStrf(PubTot, ffCurrency,8,2)
✓+ #9 + CharityName[CInd]); ✓
end;


end;                                                                      [22]
//================================================================================
end.

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

TOTAL SECTION A: 120

SECTION B: JAVA

QUESTION 1: PROGRAMING AND DATABASE

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

public class NewsPaper
{
 Connection conn;
 public NewsPaper ()
 {
 //load the driver
  try
  {
    Class.forName ("sun.jdbc.odbc.JdbcOdbcDriver");
    System.out.println ("Driver successfully loaded");
  }
  catch (ClassNotFoundException c)
  {
    System.out.println ("Unable to load database driver");
  }
  //connect to the database
  try
  {
   //conn = DriverManager.getConnection ("jdbc:odbc:litter.mdb");

   System.out.print("Type in the exact location of your database (FOR
     EXAMPLE - C:/TEST/Newspapers.mdb)");
   BufferedReader inKb = new BufferedReader (new InputStreamReader
                                                         (System.in));

   // String filename = inKb.readLine();
   String filename = "C:/NewsPaperDB.mdb";
   String database = "jdbc:odbc:Driver={Microsoft Access Driver
     (*.mdb)};DBQ=";
   database += filename.trim () + ";DriverID=22;READONLY=true}";
   conn = DriverManager.getConnection (database, "", "");

   System.out.println ("Connection to NewsPaper database successfully
                                                    established");

    }
     catch (Exception e)
     {
       System.out.println ("Unable to connect to the database");
     }
   } //end connect
   //------------------------------------------------------------------
                                                    //Question 1.1
   public void displayAllClientsQuery ()throws SQLException
   {
     System.out.println("\f");
     System.out.println();
     Statement stmt = conn.createStatement ();
```

```
                          ✓          ✓          ✓
    String sql = "SELECT * FROM ClientsTb ORDER BY ClientName" ; (3)
    ResultSet rs = stmt.executeQuery (sql);
    System.out.printf("%-10s%-27s%-20s%-
               15s","ClientID","ClientName","TelNum","Suburb");
    System.out.println();
    System.out.println();
    while (rs.next ())
    {
      String id = rs.getString ("ClientID");
      String sName = rs.getString ("ClientName");
      String sTel = rs.getString ("TelNum");
      String sSuburb = rs.getString ("Suburb");

      System.out.printf("%-10s%-27s%-20s%-15s",id,sName,sTel,sSuburb);
      System.out.println();
     }
     System.out.println(" ");
     stmt.close ();
     }
  //----------------------------------------------------------------------
                                                          // Question 1.2
  public void displayAllAdvertsQuery ()throws SQLException
  {
    System.out.println("\f");
    System.out.println();

    Statement stmt = conn.createStatement ();
                                             //all the correct fields ✓
    String sql = "SELECT  AdvertID, DatePlaced, WeeksRunning, NumWords,
               Paid,Category, ClientName FROM AdvertsTb, ClientsTb " +
                "WHERE ✓ AdvertsTB.ClientID = ClientsTB.ClientID";✓
                                                                  (3)
     ResultSet rs = stmt.executeQuery (sql);
    System.out.printf("%5s%15s%15s%15s%10s%20s%20s","AdvertID","Date
          Placed","Weeks Running","Num of words", "Paid",
                                   "Category","Client Name" );
    System.out.println();
    System.out.println();
    while (rs.next ())
    {
     String id = rs.getString ("AdvertID");
     String sDate = rs.getString ("DatePlaced");
     sDate = sDate.substring(0,10);
     String weeks = rs.getString("WeeksRunning");
     String numberW = rs.getString ("NumWords");
     String sPaid = rs.getString ("Paid");
     if (sPaid.equals("1"))
          sPaid = "True";
     else sPaid = "False";
     String sCat = rs.getString ("Category");
     String sName = rs.getString ("ClientName");

     System.out.printf("%-12s%-17s%-15s%-15s%-16s%-16s%-
               20s",id,sDate,weeks,numberW,sPaid,sCat,sName);
     System.out.println();
     }
     System.out.println(" ");
```

```
      stmt.close ();
      }
//------------------------------------------------------------------------
                                                 // Question 1.3
 public void selectCatMonthQuery ()throws SQLException
 {
   System.out.println("\f");
   System.out.println();

   Statement stmt = conn.createStatement ();
                                                      //input ✓
   String numMonth = JOptionPane.showInputDialog("Type in the number of
                                                 the month ");

   String cat = JOptionPane.showInputDialog("Type in the category ");
                                              ✓
   String sql = "SELECT  AdvertID, DatePlaced, WeeksRunning FROM
                AdvertsTb ✓WHERE Category = '" + cat + "'✓ AND
                MONTH(DatePlaced) ✓ = '" + numMonth + "' ✓  ";
   ResultSet rs = stmt.executeQuery (sql);                     (6)
   System.out.printf("%5s%15s%15s","AdvertID","Date Placed","Weeks
                                                      Running");
   System.out.println();
   System.out.println();
   while (rs.next ())
   {
     String id = rs.getString ("AdvertID");
     String sDate = rs.getString ("DatePlaced");
     sDate = sDate.substring(0,10);
     String weeks = rs.getString ("WeeksRunning");

     System.out.printf("%5s%17s%10s",id,sDate,weeks);
     System.out.println();

    }
    System.out.println(" ");
    stmt.close ();
  }// Select Gauteng
 //------------------------------------------------------------------------
                                                      // Question 1.4
  public void selectShortLongAdverts ()throws SQLException
  {
   System.out.println("\f");
   System.out.println();

   Statement stmt = conn.createStatement ();
                                                      ✓
   String sql = "SELECT AdvertID, ✓ NumWords AS [Number of Words],
           ClientName AS [Client Name] ✓ FROM AdvertsTb, ClientsTb "+
          "WHERE AdvertsTB.ClientID = ClientsTB.ClientID✓ AND
                (NumWords > 30 OR✓ NumWords < 15)" ; ✓
   ResultSet rs = stmt.executeQuery (sql);                     (6)
   System.out.printf("%-20s%-20s%-15s","AdvertID","Number of
                                   Words","Client Name");
   System.out.println();
   System.out.println();
   while (rs.next ())
   {
      String id = rs.getString ("AdvertID");
      String numberW = rs.getString ("Number of Words");
      String name = rs.getString ("Client Name");

      System.out.printf("%-20s%-20s%-15s",id,numberW,name);
      System.out.println();
     }
    System.out.println(" ");
    stmt.close ();
   } // Select short and long adverts
 //------------------------------------------------------------------------
                                                        // Question 1.5
   public void updateAdverts() throws SQLException
   {
     System.out.println("\f");
     System.out.println();

     Statement stmt = conn.createStatement ();
                                                   // input ✓
     String sAdvert = JOptionPane.showInputDialog("Enter the ID of the
                                                   advertisement ");

     String number = JOptionPane.showInputDialog("Enter the number of
                                              words to be added ");

     String sql = "UPDATE AdvertsTb SET✓ numWords = numWords✓ + " +
                 number✓ +  " WHERE AdvertID = '" + sAdvert + "'";✓

     int numRows = stmt.executeUpdate (sql);

     sql = "SELECT AdvertID, numWords AS Words FROM AdvertsTb"; ✓
                                                                    (6)
     ResultSet rs = stmt.executeQuery (sql);

     System.out.printf("%-15s%-20s","AdvertID","Number of Words");
     System.out.println();
     System.out.println();
     while (rs.next ())
     {
       String id = rs.getString ("AdvertID");
       String numberW = rs.getString ("Words");

       System.out.printf("%-15s%-20s",id,numberW);
       System.out.println();
     }
     System.out.println();
     stmt.close();
     }// update
    //---------------------------------------------------------------------
                                                   // Question 1.6


    public void selectNotPaidAdverts()throws SQLException
    {
      System.out.println("\f");
      System.out.println();
      Statement stmt = conn.createStatement ();

      String sLetter = JOptionPane.showInputDialog("Enter a letter of the
                                                    alphabet ");✓

      String sql = "SELECT AdvertID, Category, ClientName AS [Name of
                                     Client], Paid AS [Has Paid] ✓
                  FROM ClientsTb, AdvertsTb
                  WHERE ClientsTb.ClientId =  AdvertsTb.ClientId ✓
                  AND AdvertsTb.Category like✓ '" + sLetter + "%'✓  AND
                  AdvertsTb.Paid = false";✓
                                                                   (6)
      ResultSet rs = stmt.executeQuery (sql);
      System.out.printf("%-10s%-15s%-20s%-20s","AdvertID", "Category",
                              "Name of Client", "Has Paid");
      System.out.println();
      System.out.println();
      while (rs.next ())
      {
        String id = rs.getString ("AdvertID");
        String sCat = rs.getString ("Category");
        String sName = rs.getString ("Name of Client");
         String hasPaid = rs.getString("Has Paid");
        if (hasPaid.equals("0"))
            hasPaid = "False";
        else hasPaid = "True";

        System.out.printf("%-10s%-15s%-20s%-20s",id,sCat,sName,sPaid);
        System.out.println();
      }
      System.out.println(" ");
      stmt.close ();
    }
  //-----------------------------------------------------------------
                                                          //Question 1.7
   public void calcAdvertCost()throws SQLException
   {
     System.out.println("\f");
     System.out.println();

     Statement stmt = conn.createStatement ();
                                   ✓
     String sql = "SELECT  AdvertID, DatePlaced, NumWords, WeeksRunning,
                  NumWords * WeeksRunning * 0.5 ✓AS [Cost] ✓ FROM
                  AdvertsTb✓ WHERE DatePlaced >✓ #16/11/2008#" ; ✓
                                                                   (6)
     ResultSet rs = stmt.executeQuery (sql);
     System.out.printf("%5s%15s%15s%15s%8s","AdvertID","Date Placed","No.
                                of Words","Weeks Running", "Cost");
     System.out.println();
     System.out.println();
     while (rs.next ())
     {
       String id = rs.getString ("AdvertID");
       String date = rs.getString ("DatePlaced");
       date = date.substring(0,10);
      String words = rs.getString ("NumWords");
      String weeks = rs.getString ("WeeksRunning");
      double cost = Double.parseDouble(rs.getString("Cost"));

     System.out.printf("%5s%17s%10s%15s%10s%6.2f",id,date,words,weeks,
                                                             "R",cost);
     System.out.println();
    }
    System.out.println(" ");
    stmt.close ();
   }
//-------------------------------------------------------------------------
                                                       // Question 1.8
public void countCellPhones()throws SQLException
  {
    System.out.println("\f");
    System.out.println();

    Statement stmt = conn.createStatement ();
                                                                        ✓
    String sql = "SELECT Count(*)✓AS [Number of clients with cellphone
                 numbers] FROM ClientsTb WHERE [TelNum] ✓Like '0%'" ✓;
    ResultSet rs = stmt.executeQuery(sql);                        (4)
    System.out.println();
    System.out.println();
    while (rs.next ())
    {
      String num = rs.getString ("Total");
      System.out.println("Number of clients with cellphone numbers: " +
                                                                 num);
    }
    System.out.println();
    System.out.println();
    stmt.close ();
   }

   public void disconnect () throws SQLException
   {
       conn.close ();
   }

 }

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

QUESTION 2: OBJECT-ORIENTED PROGRAMMING

Adverts.java

```
import java.util.*;

// Q 2.1.1         (4 / 2) = 2
public class ✓ advertXXXX
{
    private ✓String advertText; ✓
    private String clientName; ✓
//===============================================================================

// Q 2.1.2    (4 / 2) = 2
    public advertXXXX(String at✓, String cn✓)
    {
        advertText = at; ✓
        clientName = cn; ✓
    }

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

// Q 2.1.3    (8 / 2) = 4

    public String toString ()✓
    {
        String tmp = "";
        tmp += clientName + "\t"✓ + "Word Count✓: " + totWordCount()✓ +
               "\t";
        if (isHalfPrice())✓
        {
            tmp += "Discount";✓
        }
        else
        {
            tmp += "No Discount";
        }
        tmp += "\t R" + advertCost ();✓
        return tmp; ✓
    }

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

// Q 2.1.4    (10 / 2) = 5

    public int totWordCount ()✓
    {
       int noFullWords = 0;  ⎫
       int noHalfWords = 0;  ⎬     ✓
       String curWord = "";  ⎭
       Scanner sc = new Scanner(advertText); ✓
       while (sc.hasNext(" ")) ✓
       {
           curWord = sc.next(" ");
           if (! curWord.equalsIgnoreCase("LowPolluter")) {✓
             if (curWord.length() < 5)
             {✓
                 noHalfWords++;✓
             }
             else
             {
                 noFullWords++;
             } /* end of if length < 5 */
           } /* end of if LowPolluter */
       } /* end of while loop */
       // now we check & round half the half words and add it to full word count

       noFullWords += (noHalfWords / 2); ✓
       if (noHalfWords % 2 != 0)
       {
          noFullWords += 1;  ✓
       }
       return noFullWords; ✓
    }

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

// Q 2.1.5   (6 / 2) = 3

```
    public boolean isHalfPrice ()✓
    {
      if (advertText.indexOf("LowPolluter")✓ == -1) ✓
      {
          return false; ✓
      }
      else ✓
      {
          return true; ✓
      }
    }

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

// Q 2.1.6   (4 / 2) = 2

```
    public double calculateCost ()
    {
       if (isHalfPrice ())✓
       {
         return ✓ totWordCount() * 0.5;    ✓
       }
       else
       {
         return totWordCount () * 0.25;  ✓
       }
    }
```

// Q 2.1.7   (2/2 = 1)

```
    public String ✓getClientName ()
   {
        return clientName; ✓
    }

}

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

testAdvert.java

```
import java.io.*;
```

// Q 2.2.1   (16 / 2) = 8

```
import java.io.*;
```

```
public class testAdvert
{
 public static void main (String [] args) throws Exception
 {
  advertXXXX [] arrAdverts = new advertXXXX [100]; ✓
  int adCount = 0; ✓
  File inputFile = new File ("Advertisements.txt");✓
  if (inputFile.exists())✓
  {

    FileReader in = new FileReader (inputFile); ✓
    BufferedReader inF = new BufferedReader (in);

    String line = inF.readLine ();✓ //if the next inf.readLine is in the correct
    while (line != null) ✓                              position)
     {
      String[] part = line.split("#");✓
      String adText = part[0]; ✓
      String cName = part[1]; ✓
      arrAdverts[adCount] ✓= new advertXXXX(adText,cName); ✓
      adCount++;    ✓
      line = inF.readLine ();
     }
    inF.close ();  ✓
  else
  {
    System.out.println("File does not exist"); ✓
    System.exit(0); ✓
  }


   BufferedReader inKb = new BufferedReader (new InputStreamReader (System.in));
   char ch = ' ';
   System.out.println("\f");
   while (ch != 'Q')
    {
      System.out.println("              Menu");
      System.out.println(" ");
      System.out.println("       A - Display Advert Information");
      System.out.println("       B - Summarised Income Data");
      System.out.println("       C - Look up a client");
      System.out.println(" ");
      System.out.println("       Q - QUIT");
      System.out.println(" ");
      System.out.print("       Your choice? :");
      ch = inKb.readLine().toUpperCase().charAt(0);
      switch (ch)
      {
        case 'A':                     //Display Information option ( 6/2 = 3)
                  {
                   System.out.println("\f");✓
                   System.out.println("Information on Advertisements");✓
                   System.out.println(" ");
                   for (int ind = 0; ind < adCount; ind++ ) ✓
                     {
                       System.out.println(arrAdverts[ind] ✓.toString()✓ );
                     }
                   System.out.println("\n\n\n");✓
                   break;
                  }
//================================================================================
        case 'B':                // Summarised Income Data Option ( 16/2 = 8)
```

```
                  {
                    System.out.println("\f");
                    System.out.println("Summary of income");✓
                    int halfTotal = 0;
                    int fullTotal = 0;              ✓✓
                    double totFullPrice = 0;
                    double totHalfPrice = 0;
                    for (int ind = 0; ind < adCount; ind ++) ✓
                    {
                        if (arrAdverts[ind].isLowPolluter())✓
                        {
                            totHalfPrice += arrAdverts[ind].advertCost();✓
                            halfTotal ++; ✓
                        }
                        else✓
                        {
                            totFullPrice += arrAdverts[ind].advertCost();✓
                            fullTotal ++;✓
                        }
                    }
                    int total = halfTotal + fullTotal; ✓
                    double totCost = totFullPrice + totHalfPrice; ✓
                    System.out.println (total ✓+ " advertisements in total worth
                                                          R " + totCost); ✓
                    System.out.println (halfTotal + " half-priced
                            advertisements in total worth R " + totHalfPrice);✓
                    System.out.println (fullTotal + " full-priced
                            advertisements in total worth R " + totFullPrice);✓
                     System.out.println("\n\n\n");
                    break;
                  }
//================================================================================
               case 'C':                 // Look up a client Option (10/2 = 5)
                  {
                     System.out.println("\f");
                     System.out.print("Enter the name of the client : ");✓
                     String clientName = inKb.readLine();
                     System.out.println("\f");
                     boolean found = false; ✓

                     System.out.println("Result of the search: ");
                     for (int ind = 0; ind < adCount; ind++ ) ✓
                     {
                         String cName = arrAdverts[ind].getClientName();✓
                         if (cName.equalsIgnoreCase (clientName)) ✓
                         {
                             found = true; ✓
                             System.out.println(arrAdverts[ind].toString());✓
                         }
                     }
                     if (!(found)) ✓
                     {
                         System.out.println(clientName✓ + " was not found. ");✓
                     }
                     System.out.println("\n\n\n");
                     break;
                  }
//================================================================================
                case 'Q':
                  {
                     System.exit(0);
                  } // case
```

```
            }//switch
         }  // while

    }

 }

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

QUESTION 3: JAVA PROGRAMMING

Object Oriented version (according to the question paper)

```
import java.util.Random;
import java.util.Scanner;

public class testCharityXXXX
{
public static void main(String [] args)                        Q 3.1
{
   String [] arrCharity = {"A", "C", "R", "S", "W"};
   String [][] arr = new String [5][5]; ✓✓
                                                                          [2]
//================================================================================
// initialise array                                            Q. 3.2
for (int r = 0; r<5;r++)✓
{
  for (int c = 0; c<5; c++)✓
  {
      arr[r][c] = "";✓
  }
}                                                                         [3]
//================================================================================
                                                               Q 3.3

Random randomNumbers = new Random();

for (int r = 0; r<4;r++)✓
{
  for (int c = 0; c<4; c++)✓
  {
       int value = randomNumbers.nextInt(5); ✓
       int percentage = randomNumbers.nextInt(10)+6; ✓✓
       String letter = arrCharity[value]; ✓

       arr[r][c] = letter + percentage; ✓

    }
}                                                                         [8]
//================================================================================
// Display
System.out.println("\f");                                      Q 3.4
System.out.println("Grid with random values");
System.out.println(" ");

for (int r = 0; r<4;r++)                           ⎫
{                                                  ⎪
  for (int c = 0; c<4; c++)                        ⎪
  {                                                ⎬      ✓✓✓
      System.out.print(arr[r][c] + "\t");          ⎪
  }                                                ⎭
```

```
    System.out.println("");
}
CharityXXXX [] arrCharities = new CharityXXXX[5]; ✓
System.out.println(" ");
System.out.println("Amounts received per charity");
System.out.println(" ");
System.out.println("Total        Public       Government    Charity");   ✓✓

for (int k = 0; k < 5; k++)✓                                  For each charity
{

  arrCharities[k] = new CharityXXXX();✓                       Create the object
  String letter = arrCharity[k];
  arrCharities[k].calcAmounts(k,arr);
  String charityName = arrCharities[k].getName();

  double gov = arrCharities[k].getEarnings();              ✓✓ Call the methods
                                                                for this charity
  double publicDon = arrCharities[k].getContribution();
  double total = arrCharities[k].getTotal();
  System.out.printf("R%10.2f%sR%10.2f%sR%10.2f%-3s%-50s\n",total,"
                          ",publicDon,"  ",gov," ",charityName); ✓✓ Display
}

}

}
//=============================================================================
```

Object class:

```
import java.util.Random;

public class CharityXXXX
{

  private String cName;
  private String cLetter;
  private double earnings;
  private double total;
  private int contribution;


public void setLetterName(int value)
{
 switch (value) ✓
      {
        case  0:cLetter = "S";
                cName = "CANSA";
                break;
        case  1: cLetter = "W";
               cName = "Battered Women & Children";
               break;
        case  2: cLetter = "A";
               cName = "Aids Africa";                ✓✓
               break;

        case  3 : cLetter = "R";
               cName = "Red Cross";
                break;
        case  4 : cLetter = "C";
               cName = "CHOC";
               break;
```

```
            }

        }

public String getName()
{
 return cName;
}

public void calcAmounts(int value, String [][] arr)
{

 setLetterName(value); ✓
 total = 0;           }
 contribution = 0;    }    ✓  initialize counter variables
 earnings = 0;        }
 for (int r = 0; r<4;r++)  }
 {                         }  ✓
  for (int c = 0; c<4; c++)}
  {
      String let = arr[r][c].substring(0,1); ✓

      int val = arr[r][c].length();✓

      int percentage = Integer.parseInt(arr[r][c].substring(1,val)); ✓

      if (let.equals(cLetter)) ✓
      {
          double gov = Math.round(50000 * percentage/100.0); ✓
          contribution = contribution + 50000 ; ✓
          earnings  = earnings + gov; ✓
        }

    }
  }
 total = earnings + contribution; ✓
}

public double getEarnings()
{
  return earnings;
}

public double getTotal()
{
  return total;
}

public double getContribution()
{
  return contribution;
}

}                                                                      [24]

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

Structured programming Version (not desirable – only a test class)

```
testCharity.java
import java.util.Random;
public class testCharity
{
    public static void main(String[] args)                          Q 3.1
    {
        String [][] arrGoldRush = new String[6][6]; ✓✓                [2]
      //======================================================================
                                                                  Q 3.4

        Charity goldRush = new Charity();✓
        String [] arrCharities = {"A", "C", "R", "S", "W"};✓
        String [] arrNames = {"Aids Africa", "CHOC", "Red Cross", "CANSA",
                              "Battered Woman & Children"};✓


        goldRush.initialiseArray(arrGoldRush);  ✓     // Call methods

        goldRush.assignValues(arrGoldRush);

        System.out.println("\f ");


        // Process and Display array
        System.out.println("Grid with random values ");
        for (int r = 0; r<6;r++)    }
        {                           }
           for (int c = 0; c<6; c++)}  ✓
          {
            System.out.print(arrGoldRush[r][c] + "\t");✓
          }
          System.out.println(" ");✓

        }
      System.out.println(" ");
      System.out.println("Amounts received per charity");
      System.out.println(" ");
      System.out.println("Total        Public      Government      Charity");✓

      String letter = "";
      for (int num = 0; num < 5; num++)✓ // for each charity
        {
         letter = arrCharities[num];      ✓
         String charityName = arrNames[num]; ✓
         double total = 0;             }
         double publicDon = 0;         }    ✓
         double gov = 0;               }
         for (int r = 0; r < 4; r++)      }
         {                                }  ✓
         for (int c = 0; c < 4; c++)      }
          {
              String let = arrGoldRush[r][c].substring(0,1); ✓
              int val = arrGoldRush[r][c].length();✓
              int percentage =
                        Integer.parseInt(arrGoldRush[r][c].substring(1,val)); ✓
              if (let.equals(letter)) ✓
              {
               double govContribution = 50000 * percentage/100.0; ✓
               publicDon = publicDon + 50000 ; ✓
               gov = gov + govContribution; ✓
              }
            }
          }
          total = publicDon + gov; ✓
          System.out.printf("R%-8.2f\tR%-8.2f\tR%-8.2f\t%-
                        45s\n",total,publicDon,gov,charityName); ✓✓
      }
    }
```

```
}                                                                    (22)
//===================================================================
class Charity
{

   String [][] initialiseArray( String [][] arrCharity )    Q 3.2
   {
      for (int r = 0; r < 6; r++)✓
        for (int c = 0; c < 6; c++)✓
          arrCharity[r][c] = " ";✓

       return arrCharity;

    }                                                                (3)
//=================================================================

Q 3.3
String [][] assignValues(String [][] arrCharity)
    {
       Random rand = new Random();  ⎫
       for (int r = 0; r < 4; r++)  ⎪
         {                          ⎬   ✓
         for (int c = 0; c< 4; c++) ⎭
          {
              int letNumber = rand.nextInt(5)+1; ✓✓
              String letter = "";
              switch (letNumber) ✓
              {
                  case 1: letter = "C";  ⎫
                            break;       ⎪
                  case 2: letter = "A";  ⎪
                            break;       ⎬    ✓✓
                  case 3: letter = "R";  ⎪
                            break;       ⎪
                  case 4: letter = "S";  ⎪
                            break;       ⎭
                  case 5: letter = "W";
                            break;
               }
              int percentage = rand.nextInt(10) + 6; ✓✓
              arrCharity[r][c] = letter + percentage; ✓
           }
        }

        return arrCharity;
    }
}                                                              (7) + (3)

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

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

ADDENDUM A

GRADE 12 FEBRUARY/MARCH 2009

INFORMATION TECHNOLOGY P1

COVER SHEET

Province: ____

Centre Number: ___________________________________

Examination Number: ______________________________

Programming Language (circle the language used): DELPHI / JAVA

TOTAL MARKS PER QUESTION		
QUESTION	**MARK OUT OF**	**LEARNER'S MARK**
1	**40**	
2	**43**	
3	**37**	
GRAND TOTAL	**120**	

ADDENDUM B

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 * (1) FROM ClientsTb (1) ORDER BY ClientName' (1)	**3**	
1.2	'SELECT AdvertID, DatePlaced, WeeksRunning, NumWords, Paid,Category, ClientName (1) FROM AdvertsTb, ClientsTb (1)WHERE AdvertsTB.ClientID = ClientsTB.ClientID'; (1)	**3**	
1.3	cat and month // input (1) 'SELECT AdvertID, DatePlaced, WeeksRunning (1) FROM AdvertsTb (1) WHERE Category = cat (1) AND MONTH(DatePlaced) (1) = monthNum (1)	**6**	
1.4	'SELECT AdvertID,(1) NumWords AS [Number of Words], ClientName AS [Client Name] (1) FROM AdvertsTb, ClientsTb WHERE AdvertsTB.ClientID = ClientsTB.ClientID (1) AND (NumWords > 30 OR (1) NumWords < 15)' (1)	**6**	
1.5	sAdvert and sNumber // input (1) 'UPDATE AdvertsTb SET(1) numWords = numWords (1)+ sNumber + (1) WHERE AdvertID = "' +sAdvert +"' ' (1) 'SELECT AdvertID, numWords AS Words FROM AdvertsTb'(1)	**6**	
1.6	sLetter // input (1) 'SELECT AdvertID, Category, ClientName AS [Name of Client],Paid AS [Has Paid] (1) FROM ClientsTb, AdvertsTb WHERE ClientsTb.ClientId = AdvertsTb.ClientId (1)and AdvertsTb.Category like(1) "' + sLetter + '%"(1) and advertsTb.Paid = false'; (1)	**6**	
1.7	'SELECT AdvertID, DatePlaced, NumWords, WeeksRunning, Format(NumWords * WeeksRunning * 0.5,(1)"Currency") AS [Cost] (1) FROM AdvertsTb(1) WHERE DatePlaced >(1) #16/11/2008#' (1)	**6**	
1.8	'SELECT Count(*) (1) AS [Number of clients with cellphone numbers] (1) FROM ClientsTb WHERE [TelNum] (1) Like "0%" (1) '	**4**	
	TOTAL:	**40**	

ADDENDUM C – SUPP 2009

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	**Define TAdvert attributes**: fAdvertText String (1), fClientName String (1) **Class** (1) **End** of class (1) **(4/2=2)**	**2**	
2.1.2	**Constructor:** Parameters correct order (1), correct types(1) Assignment of fields (2) **(4/2=2)**	**2**	
2.1.3	**toString method:** Check isLowPolluter (1) Make correct discount message (1) result (1) correctly formatted string: ClientName (1) Word Count (1) word count value (1) Discount message (1) cost as currency (1) **(8/2=4)**	**4**	
2.1.4	**CountWords:** Return integer(1), Assign value to Result / name of function(1), loop (1), extract current word(1) exclude 'LowPolluter' from word count (1), check for half words (1) and inc relevant counter(1) get next word(2), round half words up outside loop(1) **(10/2=5)**	**5**	
2.1.5	**isLowPolluter:** Boolean function (1), check if 'LowPolluter' (1)in advert text (1), Assign to result / name of function (2). **(6/2=3)**	**3**	
2.1.6	**calculateCost:** Check if half price (isLowPolluter called) (1), cost = words * 0.5 (1), correctly compensate for half price (1), return result (1) **(4/2=2)**	**2**	
2.1.7	**getClientName:** Return type String(1), return clientName(1) **(2/2=1)**	**1**	
2.2			
2.2.1	Declare array (1), uses object class (1) assignfile (1), if file does not exists (1), Message(1), exit(1), reset file (1), initialise counter(1),while not eof loop (1), increment counter (1), read line (1), ind hash(1), extract fields (2), instantiate new object in array (1), place in array(1) **(16/2)**	**8**	
Display Information	Set tabs for display (2), display heading (1), loop (1) Call & display toString (1) of the current object in the array(1) **(6/2)**	**3**	
Summarised Income Data	Display heading (1), Initialised counters(2),loop (1), call calculateCost (1), add cost to total (1), check if low polluter (1), correctly increment half price total (1) & counter (1), else(1) correctly increment full price total(1) & counter (1), Display half price heading, count(1) & total (1), display full price heading, count & total (1), display grand total heading, count and value (1) **(16/2= 8)**	**8**	

Look up a client	Get & store client name (1), found flag initialised (1), loop (1), check if search name = client name (2), call toString method(1) change found flag if found (1), if not found (1) display suitable message (2) **(10/2=5)**	**5**	
	TOTAL:	**43**	

ADDENDUM D

QUESTION 3: DELPHI PROGRAMMING

CENTRE NUMBER: ………………………..	**EXAMINATION NUMBER: ……………………….**		
QUESTION 3 DELPHI – MARKING GRID			
QUESTION	**ASPECT**	**MAX. MARKS**	**LEARNER'S MARKS**
3.1	Declare array 2d with 4 x 4 elements (1), of string (1)	**2**	
3.2	**Initialise Array** 2 loops (2), correct values initialise to a space (1)	**3**	
3.3	**Generate random data (funding)** 2 loops nested (2), generate random letter of (C/S/A/R/W) (4), generate random value between 5 & 15 (2), concatenate string (1), assign to array (1)	**10**	
3.4	**Calculate Totals** Initialise arrays with names and letters (or any other variables used to work with names of charities and letters) (2) Display 2dim array: nested loops (2), Display (2) Display heading for totals (1) and subheadings(1) Outer loop for charities(1) initialise counters (2) nested loops for 2 dim array (2), check which charity from array(2), add 50000 to pubTot(1) extract govt contribution (string handling) (1), calculate govt contribution amount (2), add to govt contribution to govTot (1), outside nested loop – add totals(1) and display totals(1) ***NB Learner may use ANY means to ensure correct totals. 20 variables permissible (learner penalises self on time & complexity), as is array of records.***	**22**	
	TOTAL:	**37**	

ADDENDUM E – SUPP 2009

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 *(1) FROM ClientsTb(1) ORDER BY ClientName"(1)	**3**	
1.2	"SELECT AdvertID, DatePlaced, WeeksRunning, NumWords,Paid,Category, ClientName (1)FROM AdvertsTb, ClientsTb WHERE (1) AdvertsTB.ClientID = ClientsTB.ClientID"(1)	**3**	
1.3	Input(1) SELECT AdvertID, DatePlaced, WeeksRunning (1) FROM AdvertsTb(1) WHERE Category = '" + cat + "'(1) AND MONTH(DatePlaced)(1) = '" + numMonth + "' " (1)	**6**	
1.4	"SELECT AdvertID,(1) NumWords AS [Number of Words],(1)ClientName AS [Client Name] (1) FROM AdvertsTb, ClientsTb WHERE AdvertsTB.ClientID = ClientsTB.ClientID (1) AND (NumWords > 30 OR (1) NumWords < 15)" (1)	**6**	
1.5	Input(1)"UPDATE AdvertsTb SET(1) numWords = numWords(1) + " + number(1) + " WHERE AdvertID = '" + sAdvert + "'"(1)	**6**	
1.6	Input(1) "SELECT AdvertID, Category, ClientName AS [Name of Client], Paid AS [Has Paid] (1)FROM ClientsTb, AdvertsTb WHERE ClientsTb.ClientId = AdvertsTb.ClientId (1) AND AdvertsTb.Category like(1) '" + sLetter + "%'(1) AND AdvertsTb.Paid = false";(1)	**6**	
1.7	"SELECT AdvertID, DatePlaced, NumWords, WeeksRunning,(1)NumWords * WeeksRunning * 0.5 (1) AS [Cost](1) FROM AdvertsTb (1) WHERE DatePlaced > (1) #16/11/2008#" (1)	**6**	
1.8	"SELECT Count(*)(1) AS [Number of clients with cellphone numbers] FROM ClientsTb WHERE [TelNum](1) Like '0%'(1)	**4**	
	TOTAL:	**40**	

ADDENDUM F

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	**Define Advert istance fields**:private (1) advertText String (1), clientName String (1) **Class** (1) **(4/2 = 2)**	**2**	
2.1.2	**Constructor:** Parameters correct order (1), correct types(1) Assignment of fields (2) **(4/2 = 2)**	**2**	
2.1.3	**toString method:** Correctly formatted string: ClientName (1) Word Count (1) word count value (1) with tabs(1) Check isLowPolluter (1) Make correct discount message (1) cost (1) return string(1) **(8/2 = 4)**	**4**	
2.1.4	**countWords:** Return integer(1), initialise variables(1) control loop with space(1), loop (1), exclude 'LowPolluter' from word count (1), check for half words (1) and inc relevant counter(1) , round half words up outside loop(1) and inc word counter(1) return value(1) **(10/2 = 5)**	**5**	
2.1.5	**isLowPolluter:** Boolean method (1), check if 'LowPolluter' (1)in advert text (1), return the correct Boolean value(3) **(6/2=3)**	**3**	
2.1.6	**calculateCost:** Check if half price (isLowPolluter called) (1), cost = words * 0.5 (1), correctly compensate for half price (1), return result (1) **(4/2=2)**	**2**	
2.1.7	**getClientName:** Return type String(1), return clientName(1) **(2/2=1)**	**1**	
2.2			
2.2.1	Declare array of objects(1) Initialise counter(1), Create File object(1) if file exists(1) then create FileReader object(1) while loop (1) Read from file (1), Get pos of #(1),Extract name and advert text(2) create object (1) with arguments(1)Inc counter(1), Close file outside loop(1), If file does not exist (1)display message(1) and exit(1) **(16/2= 8)**	**8**	
2.2.2 Display Advert Information:	Clear the output area(1),Heading(1) for loop(1), In loop: call toString method(1) of the object in the array(1) New line(1) **(6/2=3)**	**3**	
Summarised Income Data	Display heading (1), Initialised counters(2),loop (1), call calculateCost (1), add cost to total (1), check if low polluter (1), correctly increment half price total (1) & counter (1), else(1) correctly increment full price total(1) & counter (1), Display half price heading, count(1) & total (1), display full price heading, count& total (1), display grand total heading, count and value (1) **(16/2 = 8)**	**8**	

Look up a client	Get & store client name (1), found flag initialised (1), loop (1), check if search name = client name (4), call toString method(1) change found flag if found (1), if not found (1) display suitable message (1) with clientName(1) **(10 /2 = 5)**	**5**	
	TOTAL:	**43**	

ADDENDUM G

QUESTION 3: JAVA PROGRAMMING (with object class)

CENTRE NUMBER: …………………..	EXAMINATION NUMBER: ………………………………..		
QUESTION 3 JAVA – MARKING GRID			
QUESTION	**ASPECT**	**MAX. MARKS**	**LEARNER'S MARKS**
3.1	Declare array 2d with 4 x 4 elements (1), of string (1)	**2**	
3.2	**Initialise Array** 2 loops (2), correct values initialise to a space (1)	**3**	
3.3	**Generate random data (funding)** 2 loops nested (1), generate random letter of (C/S/A/R/W) (2), generate random value between 5 & 15 (2), concatenate string (1), assign to array (1)	**7**	
3.4	**Process and display** Display 2dim array: nested loops (1), Display (2) Initialise array for letters (1) Create array of objects(1) Display heading and subheadings(2) Outer loop for charities(1) Call methods for this object (2) Display all the values and the name of the Charity(2) **Method: calculate amounts**: Get letter for this charity (3) initialise counters (1) Nested loops for 2 dim array (1), Extract the letter(1) and the value(2) from 2dim array If the letter is this charity (1) Calculate gov amount(1), public amount(1), add to total for this charity(1) outside nested loop – add totals(1) ***NB Learner may use ANY means to ensure correct totals. 20 variables permissible (learner penalises self on time & complexity), as is array of records.***	**25**	
	TOTAL:	**37**	

QUESTION 3: JAVA PROGRAMMING (Only one test class)

CENTRE NUMBER: …………………	EXAMINATION NUMBER: …………………………………		
QUESTION 3 JAVA – MARKING GRID			
QUESTION	**ASPECT**	**MAX. MARKS**	**LEARNERS MARKS**
3.1	Declare array 2d with 4 x 4 elements (1), of string (1)	**2**	
3.2	**Initialise Array** 2 loops (2), correct values initialise to a space (1)	**3**	
3.3	**Generate random data (funding)** 2 loops nested (1), generate random letter of (C/S/A/R/W) (2), generate random value between 5 & 15 (2), Test the letter against the **(3)** concatenate string (1), assign to array (1)	**7 + 3**	
3.4	**Process and display** Display 2dim array: nested loops (1), Display (2) Create object(1) Initialise arrays with names and letters (or any other variables used to work with names of charities and letters) (2) Call methods(1) Display heading and subheadings(1) Outer loop for charities(1) Get the name and letter of this charity(2) initialise counters (1) Nested loops for 2 dim array (1), Extract the letter(1) and the value(1) from 2dim array If the letter is this charity (1) Calculate gov amount(1), public amount(1), add to total for this charity(1) outside nested loop – add totals(1) Display the information(2) ***NB Learner may use ANY means to ensure correct totals. 20 variables permissible (learner penalises self on time & complexity), as is array of records.***	**22**	
	TOTAL:	**37**	