education

Department:
Education
REPUBLIC OF SOUTH AFRICA

NATIONAL SENIOR CERTIFICATE

GRADE 12

INFORMATION TECHNOLOGY P1

NOVEMBER 2009

MEMORANDUM

MARKS: 120

The memorandum consists of 32 pages.

GENERAL INFORMATION

- **Pages 2 – 11 contain the Delphi memoranda of possible solutions for QUESTIONS 1 to 3 in programming code.**
- **Pages 12 – 22 contain the Java memoranda of possible solutions for QUESTIONS 1 to 3 in programming code.**
- **Pages 23 – 31 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.**

SECTION A: DELPHI

QUESTION 1: PROGRAMMING AND DATABASE

```
unit Question1_Uxxxx;

interface

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

ype
  TfrmL = class(TForm)
    Panel1: TPanel;
    grdBandB: TDBGrid;
    Panel2: TPanel;
    btnMrFerreira: TButton;
    btnFaltemeyer: TButton;
    tblBandB: TDataSource;
    btnList: TButton;
    btnDiscount: TButton;
    btnCost: TButton;
    BitBtn1: TBitBtn;
    btnEnglish: TButton;
    qryBandB: TADOQuery;
    procedure btnMrFerreiraClick(Sender: TObject);
    procedure btnFaltemeyerClick(Sender: TObject);
    procedure btnDiscountClick(Sender: TObject);
    procedure btnEnglishClick(Sender: TObject);
    procedure btnCostClick(Sender: TObject);
    procedure btnListClick(Sender: TObject);

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

var
  frmL: TfrmL;

implementation

{$R *.dfm}
```

```
// Question 1.1    (5)

procedure TfrmL.btnListClick(Sender: TObject);
begin
  qryBandB.Active := False;
  qryBandB.SQL.Text := 'SELECT *✓ FROM tblClients✓ ORDER BY✓  Surname✓, FName✓';
  qryBandB.Active := true;
end;

//=============================================================================
// Question 1.2    (7)
```

Award marks for ROUND✓(..., 2✓)
instead of format Currency

```
procedure TfrmL.btnMrFerreiraClick(Sender: TObject);
begin
  qryBandB.Active := False;
  qryBandB.SQL.Text := 'SELECT format(✓Sum (SellingPrice) ✓, “Currency”) ✓ AS [Total Due] ✓
from tblOrders✓ WHERE✓  ClientNo = 1✓';
  qryBandB.Active := true;
end;

//=============================================================================
// Question 1.3    (4)

procedure TfrmL.btnEnglishClick(Sender: TObject);
begin
  qryBandB.Active := False;
  qryBandB.SQL.Text := 'DELETE✓ FROM tblClients✓ WHERE Nationality = "English"✓✓';
  qryBandB.ExecSQL;
  messageDlg ('English fans deleted.',mtInformation,[mbOK],0);
end;

//=============================================================================
// Question 1.4    (10)

procedure TfrmL.btnCostClick(Sender: TObject);
begin
  qryBandB.Active := False;
  qryBandB.SQL.Text := 'SELECT Date✓, Category, ✓ SellingPrice✓, SellingPrice✓ -
(SellingPrice/125 *25))✓ AS✓ [Cost] ✓ FROM tblOrders✓ WHERE ✓Clientno = 1✓';
  qryBandB.ExecSQL;
  qryBandB.Active := True;
end;

//=============================================================================
// Question 1.5    (5)

procedure TfrmL.btnDiscountClick(Sender: TObject);
begin
  qryBandB.Active := False;
  qryBandB.SQL.Text := 'UPDATE✓ tblOrders✓ SET SellingPrice✓ = (SellingPrice - 5) ✓WHERE
                                  SellingPrice >= 30✓';
  qryBandB.ExecSQL;
  qryBandB.SQL.Text := 'SELECT * FROM tblOrders';
  qryBandB.Active := true;
end;
```

```
// Question 1.6    (9)

procedure TfrmL.btnFaltemeyerClick(Sender: TObject);
begin
  qrybandB.Active := False;                              //Brackets✓all the fieldnames✓
  qryBandB.SQL.Text := 'INSERT  INTO✓  tblClients✓ (Title, Surname, FName, IDnumber, SA, ' +
                  'Nationality) VALUES✓ ("Mr","Faltemeyer", "Harald", ' +
                  '"'7407185683074", false, "Swedish")'✓;
  qryBandB.ExecSQL;
messageDlg ('Record inserted.',mtInformation,[mbok],0);
end;

end.
```

All the data items with correct data types: double quotes for text fields✓false without quotes for boolean✓ in the correct order to correlate with fieldnames✓

NB: The order of fieldnames does not matter as long as the order is the same in both sets of brackets. Subtract marks for errors e.g. candidate can loose up to 2 marks for errors in data types of data items. Note: Text data items must be in double quotes, Boolean must be either True or false

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

QUESTION 2: OBJECT-ORIENTED PROGRAMMING

```
unit ExtraItemXXXX;

interface
// Question 2.1.1    (10 / 2) = 5
uses
  SysUtils;
Type
   TExtraItem = class✓
     private✓
      fGuestNum   :String;
      fItemType   :String;  ✓✓
      fCost       :real;
     public✓
      constructor Create(sGNum:String;IType:String;rCost:real);
      function getGuestNumber:String; ✓
      function calculateProfit:real; ✓
      function calculatePrice:real; ✓
      function toString:String; ✓
  end; ✓

//============================================================================
// Question 2.1.2    (8 / 2) = 4


implementation

constructor✓ TExtraItem.Create✓ (sGNum:String; ✓IType:
                                String; ✓rCost:real);✓
begin
   fGuestNum := sGNum; ✓
   fItemType := IType;  ✓
   fCost := rCost; ✓

end;

//============================================================================
// Question 2.1.3    (4 / 2) = 2

function TExtraItem.getGuestNumber✓:String; ✓
begin
    result✓ := fGuestNum; ✓
end;

//============================================================================
// Question 2.1.4    (4 / 2) = 2

function TExtraItem.calculateProfit:real; ✓
begin
    result✓ := fCost * ✓25 /100; ✓
end;

//============================================================================
// Question 2.1.5    (4/ 2) = 2

function TExtraItem.calculatePrice:real; ✓
begin
    result ✓:= calculateProfit✓ + fCost; ✓ // or make profit an instance field.
end;
//============================================================================
```

```
// Question 2.1.6    (8 / 2) = 4

function TExtraItem.toString:String; ✓
begin
    result := fItemType + #9✓ + FloatToStrF✓ (fCost, ffCurrency, 8,2) ✓ +
              #9 + FloatToStrF(calculateProfit, ✓ ffCurrency, 8,2) + #9 +
              FloatToStrF(calculatePrice, ✓ffCurrency, 8,2);
                                          ✓✓ formatting to 2 dec
                        // or make Price an instance field
end;
end.
//============================================================================
unit Quest2_1;  (Main Form)

unit testExtrasItem_U;

interface

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

type
  TfrmQuest2 = class(TForm)
    MainMenu1: TMainMenu;
    OptionA1: TMenuItem;
    OptionB1: TMenuItem;
    Quit1: TMenuItem;
    redOutput: TRichEdit;
    procedure Quit1Click(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure OptionA1Click(Sender: TObject);
    procedure OptionB1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmQuest2: TfrmQuest2;
implementation
//============================================================================
// Question 2.2.1    (28 / 2 = 14)
uses ExtraItemXXXX; ✓                          )
var                                            )
  arrItems :array[1..20] of TExtraItem; ✓      } must be global✓
  iCount    :integer; ✓                        )

{$R *.dfm}

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

procedure TfrmQuest2.FormActivate(Sender: TObject);
var
  TFile                              :TextFile;
  sLine, sGuestNumber, sItemType     :String;
  rCost                              :real;
  iMarkup, iHash                     :integer;
```

```
begin
  if fileExists('Extras.txt') <> true then✓
    begin✓
      ShowMessage('The text file ''Extras.txt'' does not exist'); ✓
      Exit; ✓
    end;
    AssignFile(TFile, 'Extras.txt'); ✓
    Reset(TFile); ✓
    iCount := 0; ✓
    While not eof(TFile) do✓
      begin✓
        inc(iCount); ✓
        readln(TFile, sLine); ✓
        iHash := pos ('#', sLine); ✓
        sGuestNumber := copy(sLine, 1, iHash -1); ✓
        delete(sLine, 1, iHash); ✓
        iHash := pos ('#', sLine);✓
        delete(sLine, 1, iHash); ✓
        iHash := pos ('#', sLine); ✓
        sItemType := copy(sLine, 1, iHash -1);  ✓
        delete(sLine, 1, iHash); ✓
        iHash := pos ('#', sLine);
        rCost := StrToFloat(sLine); ✓


           ✓                        ✓                    ✓
        arrItems[iCount] := TExtraItem.Create(sGuestNumber, sItemType, rCost);
     end;
     CloseFile(TFile); ✓

     redOutput.Paragraph.TabCount := 2;
     redOutput.Paragraph.Tab[0] := 60;
     redOutput.Paragraph.Tab[1] := 100;
     redOutput.Paragraph.Tab[2] := 140;
end;
//============================================================================
// Question 2.2.2  Option:List items     (20 / 2 = 10)

procedure TfrmQuest2.OptionA1Click(Sender: TObject);
var
  K              :integer;
  rTotalAmount   :real;
  sGuest         :String;
begin
  sGuest := InputBox('Guest', 'Enter the guest number', ''); ✓
  rTotalAmount := 0; ✓
  redOutput.Clear;                          ✓                              ✓
  redOutput.Lines.add('Information on extra items for guest number ' + sGuest);
  redOutput.Lines.add('');
  redOutput.Lines.add('Item              Cost           Profit        Price'); ✓
  For K := 1 to iCount do✓
  begin ✓
    if arrItems[K] ✓.getGuestNumber ✓= sGuest✓ then
      begin✓
        redOutput.Lines.add(arrItems[K].toString); ✓
        rTotalAmount := rTotalAmount ✓+ arrItems[K].calculatePrice; ✓
      end;
    end;
```

```
    redOutput.lines.add ('');
    if rTotalAmount = 0 then✓
      ShowMessage('No extra items found for guest number ' + sGuest) ✓
    else✓
      redOutput.Lines.add('The total amount due is '✓ +
                          FloatToStrF(rTotalAmount, ✓ ffCurrency,8,2)); ✓
end;
//============================================================================
```

QUESTION 3: DELPHI PROGRAMMING

```
unit Question_U;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ComCtrls, Menus;

type
  TForm1 = class(TForm)
    MainMenu1: TMainMenu;
    Convert: TMenuItem;
    Display: TMenuItem;
    redDisplay: TRichEdit;
    Quit1: TMenuItem;
    procedure FormCreate(Sender: TObject);
    procedure ConvertClick(Sender: TObject);
    procedure DisplayClick(Sender: TObject);
    procedure Quit1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

Const Max = 20;

Type
    PhoneArr = Array [1..Max] of string;

Var
    arrPhoneNos, MyArr : PhoneArr;
    NoCnt, NewCnt      : integer;

{$R *.dfm}

procedure TForm1.FormCreate(Sender: TObject);
begin
  arrPhoneNos [1] := '086NewHill';
  arrPhoneNos [2] := '086DialBar';
  arrPhoneNos [3] := '086BayView';
  arrPhoneNos [4] := '086KyaSand';
  arrPhoneNos [5] := '086SowetoN';
  arrPhoneNos [6] := '086CasaSol';
  arrPhoneNos [7] := '086TheHavn';
  arrPhoneNos [8] := '086GetFood';
  arrPhoneNos [9] := '086ThaiPlc';
  arrPhoneNos [10] := '086Cleaner';
  arrPhoneNos [11] := '086CasaRok';
  arrPhoneNos [12] := '086RixTaxi';
  arrPhoneNos [13] := '086AirTime';
  arrPhoneNos [14] := '086DialBed';
  arrPhoneNos [15] := '086DialCar';
  arrPhoneNos [16] := '086DialHlp';
  arrPhoneNos [17] := '086KyaRosa';
  arrPhoneNos [18] := '086BaySand';
  arrPhoneNos [19] := '086Cater4U';
```

```
  arrPhoneNos [20] := '0861to1Air';

  noCnt := 20;
end;
function Convert (sNumber : string) : string;
var
    ind    : integer;
    number : string;

Begin
  number := '';
  for ind := 1 to length (sNumber) do
  case upcase (sNumber[ind]) of
    'A'..'C' : number := number + '2';
    'D'..'F' : number := number + '3';
    'G'..'I' : number := number + '4';
    'J'..'L' : number := number + '5';
    'M'..'O' : number := number + '6';
    'P'..'S' : number := number + '7';
    'T'..'V' : number := number + '8';
    'W'..'Z' : number := number + '9';
  else number := number + sNumber[ind]
  end;
  insert(' ', number, 4);
  insert(' ', number, 8);
  result := number;
end;

procedure TForm1.ConvertClick(Sender: TObject);
var
    Ind    : integer;
    number : string;
begin
  redDisplay.clear;
  redDisplay.Lines.Add('Original Number' + #9 + 'Converted Number');
  for ind := 1 to Max do
    begin
      number := arrPhoneNos[ind];
      arrPhoneNos[ind] := Convert(arrPhoneNos[ind]);
      redDisplay.Lines.add (number + #9 + arrPhoneNos[ind]);
    end;
end;
```

Question 3.1
Initialise variable newString ✓
Upcase string✓
Loop✓ to length of string✓
If test lower boundary✓charAt (or substring)✓ and ✓ test upper boundary ✓then or case statement
Add new character to string✓
Repeat for all possibilities✓✓
NB Case needs an else to cope with numerical digits. For leaving number digits unchanged✓✓
[13]

Question 3.1
Inserting spaces✓✓✓✓
- with loop through string OR
- by replacing characters

[4]

Question 3.1
✓ **Heading** ✓ **Looping**
✓ **Display original number**
✓ **Call method to convert number**
✓ **Display converted number** **[5]**

```
procedure TForm1.DisplayClick(Sender: TObject);
Var
   ind, dups, ind2  : integer;
   dup              : boolean;
begin
  redDisplay.lines.Clear;
  redDisplay.lines.Add('Duplicates');

  dups := 0;
  for ind := 1 to noCnt - 1 do
  begin
     dup := false;
     ind2 := ind + 1;
     while (ind2 < max) and (dup = false) do
     begin
       if arrPhoneNos[ind] = arrPhoneNos[ind2] then
       begin
         dup := true;
         redDisplay.Lines.add(arrPhoneNos[ind]);
         inc(dups);
       end
       else
         inc(ind2);


      end; // while

  end; // for

  if dups = 0 then
    redDisplay.lines.add ('No duplicates found!')
  else
    redDisplay.lines.add ('The number of duplicates : ' + IntToStr(dups))

end;

procedure TForm1.Quit1Click(Sender: TObject);
begin
  Close;
end;

end.
```

Question 3.2
✓ Initialise counter
✓ Outer Loop
✓✓ Initialize inner counter, boolean
✓✓ Inner loop – while with Boolean and counter

✓✓✓If found, display, inc dups else✓
✓✓ inc counter
[12]

Question 3.2

✓ If no duplicates, display message✓ else display number of duplicates✓
[3]

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

SECTION B: JAVA

QUESTION 1: PROGRAMMING AND DATABASE

testBandB.java

```
import java.io.*;
import java.sql.*;
import java.util.Scanner;
public class testBandB
{
    public static void main (String[] args) throws SQLException,IOException
   {
    Scanner inKb = new Scanner(System.in);
   // OR BufferedReader inKb = new BufferedReader (new InputStreamReader
   //                                          (System.in));
      BandB DB = new BandB();

      System.out.println();
      char choice = ' ';
      do
      {
         System.out.println("        MENU");
         System.out.println();
         System.out.println("    A - List");
         System.out.println("    B - Mr Ferreira");
         System.out.println("    C - English");
         System.out.println("    D - Cost");
         System.out.println("    E - Discount");
         System.out.println("    F - Faltemeyer");
         System.out.println(" ");
         System.out.println("    Q - Quit");
         System.out.print("    Your Choice - ");
         choice = inKb.nextLine().toUpperCase().charAt(0);
         // OR  choice = inKb.readLine().toUpperCase().charAt(0);
         System.out.println(" ");
         String sql = "";
         switch(choice)
         {
             case 'A':{
//Question 1.1 (5)
                        sql = "SELECT *✓FROM tblClients✓ORDER BY✓
                        Surname,✓ FName";✓
                       DB.list(sql);
                       break;
                     }
             case 'B':{
// Question 1.2 (7)
                        sql="SELECT ROUND(✓Sum(SellingPrice) ✓,2)✓, AS
                        [Total Due] ✓FROM tblOrders✓ WHERE✓ ClientNo =
                        1✓";
                       DB.mrFerreira(sql);
                       break;
                     }
             case 'C':{
// Question 1.3 (4)
                        sql = "DELETE ✓FROM tblClients✓ WHERE Nationality =
                        'English' ";✓✓
                       DB.English(sql);
                       break;
                     }
```

Award marks for format✓(..., "Currency"✓) instead of format ROUND(….,2)

```
                    case 'D':{
// Question 1.4  (10)
                                sql = "SELECT Date✓, Category, ✓ SellingPrice✓,
                                (SellingPrice✓ - ((SellingPrice/125*25✓)) AS✓
                                [Cost] ✓ FROM tblOrders✓ WHERE✓ Clientno = 1";✓
                                DB.Cost(sql);
                                break;
                             }
                    case 'E':{
// Question 1.5   (5)
                                sql = "UPDATE✓ tblOrders✓ SET SellingPrice✓ =
                                (SellingPrice - 5) ✓ WHERE SellingPrice >= 30";✓

                                DB.discount(sql);
                                break;
                             }
                    case 'F':{
//Question 1.6    (9)          Brackets✓  around all the field names✓
                                sql = "INSERT INTO✓ tblClients✓ (Title, Surname,
                                FName, IDNumber, SA,'Nationality) VALUES✓
                                ('Mr','Faltemeyer','Harald'," +
                                "'7407185683074', false, 'Swedish')";✓

                                DB.faltemeyer(sql);
                                break;
                             }

            }
        }
        while (choice != 'Q');

        DB.disconnect();
        System.out.println("Done");
    }
}
```

All the data items with correct data types: double quotes for text fields✓false without quotes for boolean✓ in the correct order to correlate with fieldnames✓

NB: The order of fieldnames does not matter as long as the order is the same in both sets of brackets. Subtract marks for errors e.g. candidate can loose up to 2 marks for errors in data types of data items. Note: Text data items must be in double quotes, Boolean must be either True or False

BandB.java

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

 public class BandB
{
   Connection conn;


   public BandB ()
   {
   //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:BandB.mdb");

            //System.out.print("Type in the exact location of your database (FOR
EXAMPLE - C:/TEST/BandB.mdb)");
            // For input from keyboard....
            //Scanner = inKb = new Scanner(System.in);
            // String filename - inKb.nextLine();
            // OR BufferedReader inKb = new BufferedReader (new
InputStreamReader (System.in));
            // String filename = inKb.readLine();
            // OR code the JChooser component to look up the file

            String filename = "BandB.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 BandB database successfully
established");

         }
             catch (Exception e)
            {
               System.out.println ("Unable to connect to the database");
            }
      } //end connect


      public void list (String sql)throws SQLException
      {
         Statement stmt = conn.createStatement ();

         ResultSet rs = stmt.executeQuery (sql);
         System.out.printf("%-10s%-8s%-12s%-12s%-15s%-8s%-12s","ClientNo",
"Title", "Surname","FName", "IDNumber", "SA", "Nationality");
         System.out.println();

System.out.println("==========================================================
==========================");

            while (rs.next ())
            {
            String clientNo = rs.getString ("ClientNo");
            String title = rs.getString ("Title");
            String surname = rs.getString ("Surname");
            String name = rs.getString ("FName");
            String id = rs.getString ("IDNumber");
            String sa = rs.getString ("SA");
            String SA = "";
            if (sa.equals("1"))
              SA = "True";
            else
              SA = "False";
            String nationality = rs.getString ("Nationality");
```

```
            System.out.printf("%-10s%-8s%-12s%-12s%-15s%-8s%-12s",clientNo,
title,surname,name,id,SA,nationality);
           System.out.println();

         }
         System.out.println(" ");
         stmt.close ();
      }

//================================================================================
==============================
       public void mrFerreira (String sql)throws SQLException
      {
         Statement stmt = conn.createStatement ();


         ResultSet rs = stmt.executeQuery (sql);
         System.out.printf("Total Due");
         System.out.println();
         System.out.println("==================");
         while (rs.next ())
         {

           String total = rs.getString ("Total Due");
           System.out.printf("R%-10s", (total ) );
           System.out.println();

         }
         System.out.println(" ");
         stmt.close ();
      } //select


      public void English(String sql) throws SQLException
      {
        Statement stmt = conn.createStatement ();

            stmt.executeUpdate(sql);

        System.out.println (" Deleted ");

        stmt.close();
      }

      public void Cost (String sql)throws SQLException
      {
         System.out.println("\f");
         System.out.println();

         Statement stmt = conn.createStatement ();


         ResultSet rs = stmt.executeQuery (sql);
         System.out.printf("%-25s%-17s%%-15s%-10s","Date","Category",
                                                "SellingPrice","Cost");
         System.out.println();

      System.out.println("=================================================
      =========");
```

```
         while (rs.next ())
         {
            String date = rs.getString ("Date");
            String sDate = date.substring(0,10);
            String category = rs.getString ("Category");
            double sellingprice = Double.parseDouble(rs.getString
                                                ("SellingPrice"));
            String cost = rs.getString ("Cost");

            System.out.printf("%-25s%-15s%-10s%-10s",sDate,category,
                                                sellingprice, cost);
            System.out.println();
          }
         System.out.println(" ");
         stmt.close ();
      }

    public void discount (String sql) throws SQLException
    {
        Statement stmt = conn.createStatement ();
         int numRows = stmt.executeUpdate (sql);

         sql = "SELECT *  FROM tblOrders";
         ResultSet rs = stmt.executeQuery (sql);
         System.out.printf("%-10s%-10s%-20s%-20s%-10s","OrderNo", "ClientNo",
"Date","Category", "SellingPrice");
         System.out.println();

System.out.println("==========================================================
==========================");
            while (rs.next ())
            {
            String OrderNo = rs.getString ("OrderNo");
            String ClientNo = rs.getString ("ClientNo");
            String date = rs.getString ("Date");
            String sDate = date.substring(0,10);
            String category = rs.getString ("Category");
            double sellingPrice = Double.parseDouble(rs.getString
("SellingPrice"));

            System.out.printf("%-10s%-10s%-20s%-20sR%-10s",OrderNo,
ClientNo,sDate,category,sellingPrice);
            System.out.println();
        }
            System.out.println();

        System.out.println (" Records updated");

        stmt.close ();
    }
      public void faltemeyer (String sql)throws SQLException
      {
         System.out.println("\f");
         System.out.println();
         Statement stmt = conn.createStatement ();
         stmt.executeUpdate (sql);
         System.out.println("Record Inserted");
         System.out.println(" ");
         stmt.close ();
      }
```

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

QUESTION 2: OBJECT-ORIENTED PROGRAMMING

ExtraItemXXXX.java

```
// Question 2.1.1         (10 / 2) = 5
public class✓ ExtraItemXXXX         ✓ Same name as file
 {                                  ✓✓✓All methods public
   private String guestNum; ✓       ✓✓ All fields private
   private String itemType; ✓
   private double cost; ✓


//============================================================================
// Question 2.1.2    (8/ 2) = 4
            ✓No reference to class or return type
public ExtraItemXXXX✓ (String GNum, ✓ String IType, ✓ double Cost✓)
   {
     guestNum= GNum; ✓
     itemType = IType;  ✓
     cost = Cost;       ✓


}//============================================================================
// Question 2.1.3    (4/ 2) = 2
  public String ✓getGuestNum()✓
   {
      return✓ guestNum; ✓
   }


//============================================================================
// Question 2.1.4    (4 / 2) = 2
  public double ✓calculateProfit()
   {
     return ✓cost * ✓ (25/100.0) ✓
   }
//============================================================================
// Question 2.1.5   (4 / 2) = 2
public double✓ calculatePrice()
{
    return✓cost ✓+ calculateProfit() ; ✓
}
//============================================================================
// Question 2.1.6    (8 / 2) = 4
public String✓ toString()
   {
String result = itemType + "\t\t R"✓ + cost +✓ "\t\t R" +
Math.round(calculateProfit()✓*100) / 100.0 + "\t\t R" +
Math.round(calculatePrice()*✓100) / 100.0 ;
            ✓✓ for formatting to 2 decimals
      return result; ✓

   }



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

testExtraItem.java

```
// Question 2.2.1    (28 / 2) = 14
import java.io.*;
import java.util.Random;
import java.util.Scanner;
public class testExtraItemXXXX
   {

    public static void main (String [] args) throws Exception
      {
        ExtraItemXXXX✓ [] arrItems = new ExtraItemXXXX [100]; ✓
        int count = 0; ✓
        File inputFile = new File ("Extras.txt");✓
        if (inputFile.exists())✓
        {
        FileReader in = new FileReader (inputFile); ✓
        BufferedReader inF = new BufferedReader (in); ✓

        String line = inF.readLine ();✓

         while (line != null) ✓✓
         {✓
            String[]✓ part = line.split("#");✓✓
            String guestNumber = part[0]; ✓  // Skip part[1] ✓
            String itemType = part[2]; ✓
            double cost = Double.parseDouble(part[3]); ✓✓

                  ✓     ✓     ✓
            arrItems[count] = new ExtraItemXXXX(guestNumber, itemType,cost);
            count++; ✓
            line = inF.readLine ();✓
         }
         inF.close ();✓
    }
   else✓
  {
    System.out.println("File does not exist"); ✓
    System.exit(0); ✓
  }

//============================================================================
Scanner input = new Scanner(System.in);
// OR BufferedReader input = 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 - List Items");
     System.out.println(" ");
     System.out.println("       Q - QUIT");
     System.out.println(" ");
     System.out.print("       Your choice? :");

     ch = input.nextLine().toUpperCase().charAt(0);

     // OR  ch = input.readLine().toUpperCase().charAt(0);
      switch (ch)
```

```
      {
         case 'A': //Question 2.2.2   List items per guest option (20/2 = 10)
         {
          double totalAmount = 0; ✓
          System.out.println("\f");
          System.out.print("Enter the number of the guest: "); ✓
          String number = input.nextLine();
          // OR String number = input.readLine();
          System.out.println("\f");                          ✓
          System.out.println("Information on extra items for guest number "
                                                + number); ✓
          System.out.println(" ");
          System.out.println("Item type     Cost        Profit   Price");✓
          for (int k = 0; k < count; k++ ) ✓
          {✓
           if (arrItems[k] ✓.getGuestNum()✓.equals(number)) ✓
           {✓
             totalAmount = totalAmount + ✓
                  arrItems[k].calculatePrice();✓
             System.out.println(arrItems[k].toString() ); ✓
           }
          }
         System.out.println("");
         if (totalAmount == 0) ✓
         {
           System.out.println("No extra items found for guest number " +
                                                    number); ✓
         }
         else✓
          {                          ✓
           System.out.printf("%sR%4.2f\n","The total amount due is ✓
                                      ",totalAmount); ✓
           System.out.println("\n\n\n");
          }
          break;
       }
//============================================================================
     case 'Q':
      {
          System.exit(0);
      } // case

     }//switch
 }  // while
}
}

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

QUESTION 3: JAVA PROGRAMMING

Object-Oriented version

```
import java.util.Scanner;
// OR import java.io.*; for BufferedReader
public class testNumbers
{
  public static void main(String [] args) throws Exception
  {
     int max = 20;
     String [] arrPhoneNos  = new String [max];
      arrPhoneNos [0] = "086NewHill";
      arrPhoneNos [1] = "086DialBar";
      arrPhoneNos [2] = "086BayView";
      arrPhoneNos [3] = "086KyaSand";
      arrPhoneNos [4] = "086SowetoN";
      arrPhoneNos [5] = "086CasaSol";
      arrPhoneNos [6] = "086TheHavn";
      arrPhoneNos [7] = "086GetFood";
      arrPhoneNos [8] = "086ThaiPlc";
      arrPhoneNos [9] = "086Cleaner";
      arrPhoneNos [10] = "086CasaRok";
      arrPhoneNos [11] = "086RixTaxi";
      arrPhoneNos [12] = "086AirTime";
      arrPhoneNos [13] = "086DialBed";
      arrPhoneNos [14] = "086DialCar";
      arrPhoneNos [15] = "086DialHlp";
      arrPhoneNos [16] = "086KyaRosa";
      arrPhoneNos [17] = "086BaySand";
      arrPhoneNos [18] = "086Cater4U";
      arrPhoneNos [19] = "0861to1Air";
      int count = 20;
      Scanner input = new Scanner(System.in);
   // OR BufferedReader input = 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 - Convert");
       System.out.println("       B – Duplicates");
       System.out.println(" ");
       System.out.println("       Q - QUIT");
       System.out.println(" ");
       System.out.print("       Your choice? :");
       ch = input.nextLine().toUpperCase().charAt(0);
       // OR ch = input.readLine().toUpperCase().charAt(0);
       switch (ch)
        {
         case 'A':                                    // Question 3.1
         {
           System.out.println("Original number Converted number");
           for (int ind = 0; ind < count; ind++)
           {
             PhoneNumber number = new PhoneNumber(arrPhoneNos[ind]);
             System.out.print(arrPhoneNos[ind] + "\t");
             arrPhoneNos[ind] = number.Convert (arrPhoneNos[ind]);
             System.out.println(arrPhoneNos[ind]);
           }
          break;
        }
```

Question 3.1
- ✓ **Heading**
- ✓ **Looping**
- ✓ **Display original number**
- ✓ **Call method to convert number**
- ✓ **Display converted number**

[5]

```
     case 'B':
     {
       System.out.println("\nDuplicates");
       int duplicates = findDuplicates(arrPhoneNos);

       if (duplicates ==0)

      System.out.println("No duplicates were found");
       else
           System.out.println("The number of duplicates : " + duplicates);

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

   }//switch

  }  // while

} // main
```

Question 3.2
✓ **Call method to display duplicates (modularity)**
✓ **If no duplicates, display message**
✓ **else display number of duplicates**

Number can be global or local depending on design

[3]

```
 public static int findDuplicates(String [] arrPhoneNos)
 {
  int duplicates = 0;
  boolean found;
  int ind2;
  for ( int ind = 0; ind < 20; ind++)
  {
    ind2 = ind + 1;
    found = false;
    while (ind2 < 20 && !(found))
    {
      if (arrPhoneNos[ind].equals(arrPhoneNos[ind2]))
      {
         found = true;
         System.out.println(arrPhoneNos[ind]);
         duplicates++;
      }
      else
        ind2++;
    } // while
  }  // for
 return duplicates;    // OR display duplicates with if in this method
}                       //make this a void method
}  // class
```

Question 3.2
✓ **Initialise counter**
✓ **Outer Loop**
✓✓ **Initialize inner counter, boolean**
✓✓ **Inner loop – while with Boolean and counter**

✓✓✓ **If found, display, inc dups**
else ✓
✓✓ **inc counter**

[12]

```
public class PhoneNumber
{
private String number;
public PhoneNumber()
{
}

public PhoneNumber(String PhoneNumber)
{
  number = PhoneNumber;
}

public String Convert (String number)
{
  String newString =  "";

  number = number.toUpperCase();
  for (int ind = 0; ind < number.length(); ind++)
  {
   String thisChar = number.substring(ind,ind+1);
    if (number.charAt(ind) >='A' && number.charAt(ind)<= 'C')
       newString = newString + "2";
   else  if (number.charAt(ind) >='D' && number.charAt(ind)<= 'F')
       newString = newString + "3";
   else   if (number.charAt(ind) >='G' && number.charAt(ind)<= 'I')
       newString = newString + "4";
    else    if (number.charAt(ind) >='J' && number.charAt(ind)<= 'L')
       newString = newString + "5";
   else     if (number.charAt(ind) >='M' && number.charAt(ind)<= 'O')
       newString = newString + "6";
   else     if (number.charAt(ind) >='P' && number.charAt(ind)<= 'S')
       newString = newString + "7";
   else     if (number.charAt(ind) >='T' && number.charAt(ind)<= 'V')
       newString = newString + "8";
   else     if (number.charAt(ind) >='W' && number.charAt(ind)<= 'Z')
       newString = newString + "9";
   else newString = newString + thisChar;

 } // for
  newString = newString.substring(0,3) + " " + newString.substring(3,6) + " " +
                     newString.substring(6,10);
  return newString;
 }
 }
```

Question 3.1
Initialise variable newString ✓
Upcase string✓
Loop✓ to length of string✓
If test lower boundary✓charAt (or substring)✓ and ✓ test upper boundary ✓then or case statement
Add new character to string✓
Repeat for all possibilities✓✓
NB Case needs an else to cope with numerical digits. For leaving number digits unchanged✓✓
[13]

Q 3.1
Inserting spaces✓✓✓✓
- with loop through string OR
- by replacing characters

[4]

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

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

ADDENDUM A

GRADE 12 NOVEMBER 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 – November 2009

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 *✓ FROM tblClients✓ ORDER BY✓ Surname✓, FName✓	**5**	
1.2	SELECT Format(✓,Sum (SellingPrice) ✓, "Currency") ✓, as [Total Due] ✓ FROM tblOrders✓ WHERE✓, ClientNo = 1✓ **OR** SELECT Round(✓,Sum (SellingPrice) ✓, 2) ✓, as [Total Due] ✓ FROM tblOrders✓ WHERE✓, ClientNo = 1✓	**7**	
1.3	Delete✓ FROM tblClients✓ WHERE Nationality = 'English'✓✓	**4**	
1.4	SELECT Date✓, Category, ✓ SellingPrice✓, SellingPrice✓ - (SellingPrice/125 *25))✓ AS✓ [Cost] ✓ FROM tblOrders✓ WHERE ✓Clientno = 1✓	**10**	
1.5	UPDATE✓ tblOrders SET Price✓ = (Price – 5) ✓ WHERE Price > 10✓Show all the fields✓	**5**	
1.6	Brackets✓all the fieldnames✓ INSERT INTO✓ tblClients✓ (Title, Surname, FName, IDnumber, SA, Nationality) values✓ ("Mr", "Faltemeyer", "Harald", "7407185683074", false, "Swedish") in brackets✓ All the data items with correct data types: double quotes for text fields✓false without quotes for boolean✓ in the correct order to correlate with fieldnames✓ NB: The order of fieldnames does not matter as long as the order is the same in both sets of brackets. Subtract marks for errors e.g. a candidate can loose up to 2 marks for errors in data types of data items. Note: Text data items must be in double quotes, Boolean must be either true or false	**9**	
	TOTAL:	**40**	

ADDENDUM C – November 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 TExtraItem**: class(1)private (1)fields declared correctly(2), methods public(1) , the four functions add to public section (4) **End** of class (1) **(10/2=5)**	**5**	
2.1.2	**Constructor:** constructor(1) Name of constructor: Object.Create(1)Parameters correct order (1), correct types(1) parameters(1) Assign to fields (3) **(8/2=4)**	**4**	
2.1.3	**get method:** getGuestNum(1),return String(1) correct field(1), return(1) **(4/2=1)**	**2**	
2.1.4	**Calculate profit:** return real(1), correct formula(2) return(1) **(4/2=2)**	**2**	
2.1.5	**Calculate price:** return real(1), call calcProfit method (1) add fCost(1), assign to result(1) **(4/2=2)**	**2**	
2.1.6	**toString:** Return String (1), Add the guest number to the string (1) convert cost (1), call calculate profit method(2) call calculate price method(1), Format the cost and price fields(2) **(8/2=4)**	**4**	
2.2			
2.2.1	Declare array (1), uses object class (1), declare counter(1) – all three must be global(1), If file does not exists (1), Begin(1), Message(1), exit(1), Assignfile(1) reset file (1), initialiseer counter (1) While not eof(file name) do begin(2), inkrementeer counter (1),read line (1), pos of hash(1), copy number of guest (1), delete(1), pos hash(1) delete to skip name(1), pos hash(1) copy item type(1), delete(1), pos hash &, copy cost & convert(1), instantiate new object(2), place in array(1) closeFile(1) **(28/2 = 14)**	**14**	
2.2.2 List Items	Inputs: Guestnumber (1), initialise total(1), headings(3), for loop(1), begin (1), if & begin(4), call toString(1), call calculatePrice method(1) add to total(1) Outside for, if no items(1) message(1), else(1), message(1)with total(1),formatted to 2 decimal places(1) **(20/2 = 10)**	**10**	
	TOTAL:	**43**	

ADDENDUM D – November 2009

QUESTION 3: DELPHI PROGRAMMING

CENTRE NUMBER: ……..	**EXAMINATION NUMBER: ……**		
QUESTION 3 JAVA – MARKING GRID			
QUESTION	**ASPECT**	**MAX. MARKS**	**LEARNER'S MARKS**
3.1	The learners may solve this as they see fit: Heading(1) Loop (1) Display Original numbers(1) Display new numbers(1) Call method to convert – modularity(1) To convert: • Initialise string • Case conversion • Loop to length of string • Switch / case / if statements • Leave numbers as is • Add spaces in new number	(5) (1) (1) (2) (7) (2) (4) **[22]**	
3.2	MEMO Learners may answer as they see fit. Marks should be allocated for: • Initialise counter • Outer loop • Initialise counter for inner loop and Boolean • Inner loop – while with two conditions – counter and found • Checking if item exists Display, increment counter, Boolean = true • Else inc inner loop counter • Outside loop: If no duplicates, display message else display number of duplicates OR Various options could be followed. Whichever method they followed mark according to the following rubric:	(1) (1) (2) (2) (2) (3) (1) (3) **[15]** **[37]**	

ADDENDUM D – November 2009 (continued)

Level	***Description***	Mark range	
0	*Question not answered*	0	
1	*Learner clearly lost and unable to answer. A few variables declared, possibly an empty loop, messages displayed without justification, etc. Allocate marks at 1 per appropriate item, even if out of context to maximum of 3.*	1 – 3	
2	*Basic structures correct – learner loops appropriately and compares appropriately. Duplicates NOT identified successfully.*	4 – 8	
3	*Code correct but basic syntax errors / simple logic errors prevent it from working. 1 error = 12, 5 or more errors = 9.*	9 – 12	
4	*Code works but inelegant solution for example no modules. All code in one place.*	13 – 14	
5	*Elegant solution, uses good techniques and additional methods*	15	
	TOTAL:	**37**	

ADDENDUM E – November 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 *✓ FROM tblClients✓ ORDER BY✓ Surname✓, FName✓	**5**	
1.2	SELECT ROUND(✓Sum (SellingPrice) ✓, 2) ✓ AS TotalDue ✓ FROM tblOrders✓ WHERE ✓ clientNo = 1✓	**7**	
1.3	DELETE✓ FROM clients✓ WHERE nationality = 'English'✓✓	**4**	
1.4	SELECT Date✓, Category, ✓ SellingPrice✓, (SellingPrice✓ - ((SellingPrice/125*25✓)) AS✓ [Cost] ✓ FROM tblOrders✓ WHERE✓ Clientno = 1	**10**	
1.5	UPDATE✓ tblOrders SET price✓ = (SellingPrice – 5) ✓ WHERE SellingPrice > 10✓ Show all the fields✓	**5**	
1.6	Brackets✓all the fieldnames✓ INSERT INTO✓ tblClients✓ (Title, Surname, FName, IDnumber, SA, Nationality) VALUES✓ ("Mr", "Faltemeyer", "Harald", "7407185683074", false, "Swedish") in brackets✓ All the data items with correct data types: double quotes for text fields✓false without quotes for boolean✓ in the correct order to correlate with fieldnames✓ NB: The order of fieldnames does not matter as long as the order is the same in both sets of brackets. Subtract marks for errors e.g. a candidate can loose up to 2 marks for errors in data types of data items. Note: Text data items must be in double quotes, Boolean must be either true or false	**9**	
	TOTAL:	**40**	

ADDENDUM F – November 2009

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 ExtraItem**: class public(1)Name of class same as file(1) all fields private (2) fields declared correctly(3), all methods public(3) **(10/2 = 5)**	**5**	
2.1.2	**Constructor:** without class type(1) correct name (1), Parameters(1) correct order (1), correct types(1) Assign to fields (3) **(8/2 = 4)**	**4**	
2.1.3	**1 get method:** Correct return type(1) Correct field returned (1), return(1), name of method(1) **(4/2=1)**	**2**	
2.1.4	**Calculate profit:** Correct return type(1), correct formula(2) return(1) **(4/2=2)**	**2**	
2.1.5	**Calculate price:** Correct return type(1), correct formula(2), return(1) **(4/2=2)**	**2**	
2.1.6	**toString:** Return String (1), Add the guest number to the string (1) cost (1), call calculate profit method(2) call calculate price method(1), Format the double fields(2) **(8/2=2)**	**4**	
2.2			
2.2.1	Declare array of objects(2) Initialise counter(1), Create File object(1) if file exists(1) then create FileReader object(2) while loop (2), { (1), Read from file (2), Create split array(2) with # as delimiter(1),Extract guest number from array(1) skip the name(1), extract itemType(1), extract cost (1), create object (2) with arguments(1)Inc counter(1), Close file outside loop(1), If file does not exists (1), {(1), Message(1), exit(1) **(28/2= 14)**	**14**	
2.2.2 List Items	Inputs: Guestnumber (1), initialise total(1), headings(3), for loop(1), begin (1), if & begin(4), call toString(1), call calculatePrice method(1) add to total(1) Outside for, if no items(1) message(1), else(1), message(1)with total(1),formatted(1) **(20/2 = 10)**	**10**	
	TOTAL:	**43**	

ADDENDUM G – November 2009

QUESTION 3: JAVA PROGRAMMING

CENTRE NUMBER: ……..	EXAMINATION NUMBER: ……		
QUESTION 3 JAVA – MARKING GRID			
QUESTION	**ASPECT**	**MAX. MARKS**	**LEARNER'S MARKS**
3.1	The learners may solve this as they see fit: Heading(1) Loop (1) Display Original numbers(1) Display new numbers(1) Call method to convert – modularity(1) To convert: • Initialise string • Case conversion • Loop to length of string • Switch / case / if statements • Leave numbers as is • Add spaces in new number	(5) (1) (1) (2) (7) (2) (4) **[22]**	
3.2	MEMO Learners may answer as they see fit. Marks should be allocated for: • Initialise counter for duplicates • Outer loop • Initialise counter for inner loop and Boolean • Inner loop – while with two conditions – counter and found • Checking if item exists Display, increment counter, Boolean = true • Else inc inner loop counter • Outside loops: If no duplicates, display message Else display number of duplicates OR Various options could be followed. Whichever method they followed mark according to the following rubric:	(1) (1) (2) (2) (2) (3) (1) (3) **[15]** **[37]**	

ADDENDUM G – November 2009 (continued)

Level	***Description***	Mark range	
0	*Question not answered*	0	
1	*Learner clearly lost and unable to answer. A few variables declared, possibly an empty loop, messages displayed without justification, etc. Allocate marks at 1 per appropriate item, even if out of context to maximum of 3.*	1 – 3	
2	*Basic structures correct – learner loops appropriately and compares appropriately. Duplicates NOT identified successfully.*	4 - 8	
3	*Code correct but basic syntax errors / simple logic errors prevent it from working. 1 error = 12, 5 or more errors = 9.*	9– 12	
4	*Code works but inelegant solution for example no modules. All code in one place.*	13 - 14	
5	*Elegant solution, uses good techniques and additional methods*	15	
	TOTAL:	**37**	