



Province of the  
**EASTERN CAPE**  
EDUCATION

**NATIONAL  
SENIOR CERTIFICATE**

**GRADE 11**

**NOVEMBER 2014**

**INFORMATION TECHNOLOGY P1  
MEMORANDUM**

**MARKS: 150**

---

This memorandum consists of 12 pages.

---

**QUESTION 1: DELPHI DATABASE CONNECTIVITY****Mark Allocation**

---

**SAMPLE DELPHI SOLUTION QUESTION 1**

```
unit Unit1;

interface

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

type
  TForm1 = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    edtName: TEdit;
    edtSur: TEdit;
    Label3: TLabel;
    edtGross: TEdit;
    Label4: TLabel;
    edtSGross: TEdit;
    Label5: TLabel;
    edtMaid: TEdit;
    Label6: TLabel;
    edtFood: TEdit;
    Label7: TLabel;
    edtAcco: TEdit;
    Label8: TLabel;
    edtLoan: TEdit;
    Button1: TButton;
    BitBtn1: TBitBtn;
    Label10: TLabel;
    edtTitle: TEdit;
    redDisplay: TRichEdit;
    Button2: TButton;
    procedure Button1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  totCash: real;

implementation

//uses Unit2;
```

```

{$R *.dfm}
// Function to calculate the Income Tax and Net Salary
function NetSalary(msGross:string):real;
var
  yGross, mrGross:real;
begin
  mrGross:=StrToFloat(msGross);

yGross:=mrGross*12 -67134.00;
  if (yGross>0) and (yGross<165600.00)
  then NetSalary:=mrGross - yGross*0.18/12 else
  if (yGross>165001.00) and (yGross<258750.00)
  then NetSalary:=mrGross - mrGross*0.25 else
  if (yGross>258751.00) and (yGross<358110.00)
  then NetSalary:=mrGross - mrGross*0.30 else
  if (yGross>358111.00) and (yGross<500940.00)
  then NetSalary:=mrGross - mrGross*0.35 else
  if (yGross>500941.00) and (yGross<638600.00)
  then NetSalary:=mrGross - mrGross*0.38
  Else
  if (yGross>638601) then NetSalary:=mrGross - mrGross*0.40;
end;

procedure TForm1.Button1Click(Sender: TObject);
Var
netSal,famSal, totSal,totExp,tloan:real;
begin
  redDisplay.Clear;
  //Set the height of the form
  Form1.ClientHeight:=417;
  //Calculate the salaries
  netSal:=NetSalary(edtGross.Text);
  famSal:=netSal+ NetSalary(edtSGross.Text);
  //Calculate the Total Expenses
  totExp:=strtofloat(edtMaid.Text) + strtofloat(edtAcco.Text)+
strtofloat(edtFood.Text);
  //Calculate the money available for the loan repayment
  totCash:=famSal-totExp;
  tLoan := strtofloat(edtLoan.text);
  mval:=totCash-(tLoan/24);
  //Display
  redDisplay.Lines.Add(edtTitle.Text + ' '+edtName.Text+'
'+edtSur.Text);
  redDisplay.Lines.Add('Your Monthly Net Salary after Tax
deduction is '+ FloattoStr(netSal));
  redDisplay.Lines.Add('Your Familys Monthly Net Salary after Tax
deduction is '+ FloattoStr(famSal));
  redDisplay.Lines.Add('After expenses your income is '+
FloattoStr(totCash));
  if mval >3000 then
    redDisplay.Lines.Add('Monthly Repayment of
'+FloatToStr((tLoan/24))+ 'You Qualify for the Loan')
  Else
    redDisplay.Lines.Add('Monthly Repayment of
'+FloatToStr((tLoan/24))+ ' You do not qualify for the Loan');
end;

```

```
procedure TForm1.FormCreate(Sender: TObject);
```

```
Begin
```

```
    //Clear all the edits  
    edtName.Clear;  
    edtSur.Clear;  
    edtTitle.Clear;  
    edtGross.Clear;  
    edtSGross.Clear;  
    edtMaid.Clear;  
    edtFood.Clear;  
    edtAcco.Clear;  
    edtLoan.Clear;
```

```
end;
```

```
end.
```

**[34]**

**QUESTION 2: DELPHI PROGRAMMING****SAMPLE: DELPHI SOLUTION QUESTION 2**

```

unit Q_2;

interface

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

type
  TForm2 = class(TForm)
    redDisplay: TRichEdit;
    Button1: TButton;
    Categories: TButton;
    procedure FormCreate(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure CategoriesClick(Sender: TObject);
  private
    { Private declarations }
    function cdValid(sCode: String; nChar:String): Boolean;
    procedure loadfile;
  public
    { Public declarations }
  end;

var
  Form2: TForm2;
  arrCodes:Array[1..35] of string; ✓
  arrDesc:Array[1..35] of string; ✓
  iNum:integer; ✓

implementation

{$R *.dfm}
//-----
procedure TForm2.Button1Click(Sender: TObject);
begin
  loadfile; ✓
end;

procedure TForm2.CategoriesClick(Sender: TObject);
var
  iCount,sBU : Integer;
  sList , sCateg : String;
begin
  loadfile; ✓
  redDisplay.Clear;
  sBU := 0; ✓
  redDisplay.Lines.Add('List of furniture type represented: '); ✓
  redDisplay.Lines.Add(' ');
  For iCount := 1 to iNum do ✓
    Begin
      sCateg := Copy (arrFCodes[iCount], 1 ,2); ✓
      if sCateg='BU' then ✓
        begin
          sBU:=sBU+1; ✓
        end;
    end;
  end;

```

```

    End;
    redDisplay.lines.add('The number of Bedroom Units is : '
+IntToStr(sBU)); ✓
    redDisplay.lines.add('The number of Bedroom Suites is : '
+IntToStr(iNum-sBU)); ✓
    redDisplay.lines.add('The Total number is : ' +IntToStr(iNum)); ✓

end;

function TForm2.cdValid(sCode: String✓; cdesc:char✓): Boolean; ✓
var k : Integer;
begin
    result := True; ✓
    if sCode[3] <> cdesc ✓then result:=false✓
    else begin
    if length(sCode) <> 7✓
    then Result := false✓
    else begin
        for k := 1 to 5 do✓
            if not(sCode[k] in ['A'..'Z']) ✓ then Result := false; ✓
        for k := 6 to 7 do✓
            if not(sCode[k] in ['0'..'9']) ✓ then Result := false; ✓

        end;
    end; //end else
end;
//-----

procedure TForm2.loadfile; ✓
var TextF :TextFile; ✓
    sName, sLine,sCode : String;
    iCommaPos, iInvalid : Integer;
begin

    if FileExists('furniture.txt') = False then✓
    begin
        MessageDlg('Sorry data file not located - application will now
terminate ', mtError,mbOk] , 0); ✓
        Application.Terminate; ✓
    end;
    iNum := 0; ✓
    iInvalid := 0; ✓
    AssignFile(TextF, 'furniture.txt'); ✓
    Reset(TextF); ✓
    while not eof(TextF) do✓
    begin
        inc(iNum); ✓
        Readln(TextF, sLine); ✓
        iCommaPos := Pos (',' , sLine);
        sName := Copy (sLine , 1 , iCommaPos - 1); ✓✓
        Delete(sLine,1,iCommaPos);
        sCode:=sLine; ✓✓
        arrCodes[iNum] ✓:= sCode;
        arrDesc[iNum] ✓:=sName;
        if CdValid(sCode,sName[1]) ✓ //test for valid
            then begin
redDisplay.Lines.Add(arrDesc[iNum] ✓+' ' +arrCodes[iNum] ✓+ '<----Valid
Code' ); ✓

```

```
        end
    else✓
    begin
        inc(iInvalid); ✓
        redDisplay.Lines.Add(arrFDesc[iNum]+'
'+arrFCodes[iNum]+' <---
Invalid Code' ); ✓✓
        end; //end if
    end;
    redDisplay.Lines.Add(' ');
    redDisplay.Lines.Add('Number of invalid Codes: ' +
IntToStr(iInvalid)); ✓
    redDisplay.Lines.Add('Number of valid Codes: ' + IntToStr(iNum-
iInvalid)); ✓
    CloseFile(TextF); ✓
end;

procedure TForm2.FormCreate(Sender: TObject);
begin
    redDisplay.Clear;

end;

end.
```

**QUESTION 3: DELPHI PROGRAMMING****Mark Allocation**

---

**SAMPLE DELPHI SOLUTION QUESTION 3**

```
unit Q3_u;

interface

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

type
  TForm1 = class(TForm)
    edtName: TEdit;
    edtSurname: TEdit;
    edtDob: TEdit;
    lblName: TLabel;
    lblSurname: TLabel;
    lblDob: TLabel;
    btnGenerate: TButton;
    lblPassword: TLabel;
    procedure btnGenerateClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.btnGenerateClick(Sender: TObject);

Var
  cName, cSurname, v1, v2: string;
  cDob, k, i: integer;
begin
  cName:= edtName.Text; ✓
  cSurname:=edtSurname.Text; ✓
  if cName='' then
    ShowMessage('Enter Name Please');
  if cSurname='' then
    ShowMessage('Enter Surname Please');
  if edtDob.Text ='' then
    ShowMessage('Enter Date of Birth Please');
  Exit;}
```



```
cDob:=StrToInt(edtDob.Text); ✓  
  
v1:='';  
v2:='';  
  for i := 1 to length(cName) ✓do  
    if cName[i] In ['a','e','i','o','u','A','E','I','O','U'] ✓✓  
      then v1:=v1+cName[i]; ✓✓  
  
  for i := 1 to length(cSurname) ✓ do  
    cSurname[i] In ['a','e','i','o','u','A','E','I','O','U'] ✓✓  
      then v2:=v2+cSurname[i]; ✓✓  
  Randomize; ✓  
  k:=cDob✓+2000✓+random(99)+1; ✓  
  
  lblPassword.Caption:= v1[1] ✓+v2[1] ✓ + IntToStr(k) ✓;  
  
end;
```

**[20]**

**QUESTION 4: DELPHI PROGRAMMING AND DATABASE****Mark Allocation**

---

**SAMPLE DELPHI SOLUTION QUESTION 4**

```
unit Q_4;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics,
  Controls, Forms,
  Dialogs, Grids, DBGrids, DMQ4, StdCtrls, Buttons;

type
  TForm2 = class(TForm)
    DBGrid1: TDBGrid;
    BtnA: TButton;
    BtnB: TButton;
    BtnC: TButton;
    BtnD: TButton;
    BtnE: TButton;
    BtnF: TButton;
    BitBtn1: TBitBtn;
    procedure BtnAClick(Sender: TObject);
    procedure BtnBClick(Sender: TObject);
    procedure BtnCClick(Sender: TObject);
    procedure BtnDClick(Sender: TObject);
    procedure BtnEClick(Sender: TObject);
    procedure BtnFClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form2: TForm2;

implementation

{$R *.dfm}

procedure TForm2.BtnAClick(Sender: TObject);
begin
  DataForm.AdoTable1.Sort := 'Brand DESC', Size ASC';
end;

procedure TForm2.BtnBClick(Sender: TObject);
begin
  with DataForm.AdoTable1 do
  begin
    Filter := 'Brand = ''LG''';
    Filtered := True;
  end;
end;
```

```

    end;
    ShowMessage('Types of Televisions by LG: ✓ ' +
IntToStr(DataForm.AdoTable1.RecordCount✓));
    DataForm.AdoTable1.Filtered := False;
end;

procedure TForm2.BtnCClick(Sender: TObject);
var iSum : Integer;
begin
    iSum := 0; ✓
    with DataForm do
        begin
            AdoTable1.Open; ✓
            AdoTable1.First; ✓
            while not AdoTable1.Eof do✓
                begin
                    iSum := iSum + AdoTable1['Stock']; ✓
                    AdoTable1.Next; ✓
                end;
            end;
            ShowMessage('Number of all Televisions in Stock: ' +
IntToStr(iSum)); ✓
        end;
    end;

procedure TForm2.BtnDClick(Sender: TObject);
var
Tbrand:string;
cost:real;
begin
cost:=0; ✓
Tbrand:=inputbox('Search','Enter Brand',''); ✓
with DataForm do
    begin
        AdoTable1.Open; ✓
        AdoTable1.First; ✓
        while not Adotable1.Eof do✓
            begin
                if AdoTable1['Brand']=Tbrand then✓
                    cost:=cost + (AdoTable1['Price']*
adoTable1['Stock']); ✓
                    AdoTable1.Next; ✓
                end;
            end;//while
        if cost =0 then✓

            ShowMessage('That type of Television does not exist: Check you
spelling and try again' ) ✓
            else✓
                ShowMessage('Total Value of Televisions : R' FloatToStr(cost));✓
                DataForm.AdoTable1.Filtered := False;
        end;
    end;

procedure TForm2.BtnEClick(Sender: TObject);
begin
    with DataForm do
        begin

```

```
    AdoTable1.Insert; ✓
    AdoTable1['Price']:=1924.00; ✓
    AdoTable1['Brand']:= 'LG'; ✓
    AdoTable1['Size']:=14; ✓
    AdoTable1['TypeT']:= 'Monitor'; ✓
    AdoTable1['Stock']:=2; ✓

    AdoTable1.Post; ✓
end;

end;

procedure TForm2.BtnFClick(Sender: TObject);
var
  NPost:integer;
begin
  with DataForm do
  begin
    AdoTable1.First; ✓
    AdoTable1.Next;
    while Not AdoTable1.Eof do ✓
    begin
      if (AdoTable1['Brand']='LG') ✓ And (AdoTable1['Size']=14)
      then

        AdoTable1.Delete ✓

      else
        AdoTable1.Next; ✓
    end;
  end;
end;

end;

end.
```

**[38]****TOTAL: 150**

```

AdoTabel1.Insert; ✓
AdoTabel1['Price']:=1924.00; ✓
AdoTabel1['Brand']:= 'LG'; ✓
AdoTabel1['Size']:=14; ✓
AdoTabel1['Type']:= 'Monitor'; ✓
AdoTabel1['Stock']:=2; ✓
AdoTabel1.Post; ✓
end;

procedure TForm2.BtnClick(Sender: TObject);
var
  Npost:integer;
begin
  with DataForm do
    begin
      with DataForm do
        begin
          AdoTabel1.First; ✓
          AdoTabel1.Next;
          while Not AdoTabel1.EOF do ✓
            begin
              if (AdoTabel1['Brand']='LG') ✓ And (AdoTabel1['Size']=14)
                then
                  AdoTabel1.Delete; ✓
            else
              AdoTabel1.Next; ✓
            end;
          end;
        end;
      end;
    end;
  end;
end;

```

[38]

TOTAL: 150

```

end;
ShowMessage('Types of Televisions by LG: ' +
  IntToStr(DataForm.Adotabel.RecordCount) +
  IntToStr(DataForm.Adotabel.Filtered := False);
end;
procedure TForm2.BtnClick(Sender: TObject);
var iSum : Integer;
begin
  iSum := 0;
  with DataForm do
    begin
      Adotabel.Open;
      Adotabel.First;
      while not Adotabel.Eof do
        begin
          iSum := iSum + Adotabel['Stock'];
          Adotabel.Next;
        end;
      end;
      ShowMessage('Number of all Televisions in Stock: ' +
        IntToStr(iSum));
    end;
  end;
end;
procedure TForm2.BtnClick(Sender: TObject);
var
  Tbrand:string;
  cost:real;
begin
  cost:=0;
  with DataForm do
    begin
      Tbrand:=inputbox('Search','Enter Brand','');
      cost:=0;
      while not Adotabel.Eof do
        begin
          Adotabel.Open;
          Adotabel.First;
          while not Adotabel.Eof do
            begin
              if Adotabel['Brand']='Tbrand' then
                cost:=cost + (Adotabel['Price'] *
                  Adotabel['Stock']);
              Adotabel.Next;
            end;
          end;
          if cost = 0 then
            ShowMessage('That type of Television does not exist: Check you
              spelling and try again' )
            else
              ShowMessage('Total Value of Televisions : R' FloatToStr(cost));
          DataForm.Adotabel.Filtered := False;
        end;
      end;
    end;
  end;
end;
procedure TForm2.BtnClick(Sender: TObject);
begin
  with DataForm do
    begin
      procedure TForm2.BtnClick(Sender: TObject);
      begin
        begin
          with DataForm do
            begin
              Tbrand:=inputbox('Search','Enter Brand','');
              cost:=0;
              while not Adotabel.Eof do
                begin
                  Adotabel.Open;
                  Adotabel.First;
                  while not Adotabel.Eof do
                    begin
                      if Adotabel['Brand']='Tbrand' then
                        cost:=cost + (Adotabel['Price'] *
                          Adotabel['Stock']);
                      Adotabel.Next;
                    end;
                  end;
                  if cost = 0 then
                    ShowMessage('That type of Television does not exist: Check you
                      spelling and try again' )
                    else
                      ShowMessage('Total Value of Televisions : R' FloatToStr(cost));
                  DataForm.Adotabel.Filtered := False;
                end;
              end;
            end;
          end;
        end;
      end;
    end;
  end;
end;

```



```
cDob:=StrToInt(edtDob.Text);
v1:='';
v2:='';
for i:=1 to length(cName)
  if cName[i] in ['a','e','t','o','u','A','E','I','O','U']
  then v1:=v1+cName[i];
for i:=1 to length(cSurname)
  do
  for i:=1 to length(cSurname)
  do
  if cSurname[i] in ['a','e','t','o','u','A','E','I','O','U']
  then v2:=v2+cSurname[i];
Randomize;
k:=cDob+2000+random(99)+1;
lblPassword.Caption:=v1[1]+v2[1]+IntToStr(k);
end;
```

[20]



## VRAAG 3: DELPHI PROGRAMMERING

## Mark Allocation

## VOORBEELD DELPHI OPLOSSING VRAAG 3

```

unit Q3_u;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics,
  Controls, Forms,
  Dialogs, StdCtrls;
type
  TForm1 = class(TForm)
    edtName: TEdit;
    edtSurname: TEdit;
    edtDob: TEdit;
    lblName: TLabel;
    lblSurname: TLabel;
    lblDob: TLabel;
    btnGenerate: TButton;
    lblPassword: TLabel;
  private
    procedure btnGenerateClick(Sender: TObject);
  public
    { Private declarations }
    { Public declarations }
end;
var
  Form1: TForm1;
implementation
{$R *.dfm}

procedure TForm1.btnGenerateClick(Sender: TObject);
  Var
    cName, cSurname, v1, v2: string;
    cDob, k, i: integer;
  begin
    cName:= edtName.Text;
    cSurname:=edtSurname.Text;
    {if cName='' then
      ShowMessage('Enter Name Please');
    if cSurname='' then
      ShowMessage('Enter Surname Please');
    if edtDob.Text = '' then
      ShowMessage('Enter Date of Birth Please');
    Exit;}
  end;
end.

```

```

end
else
begin
inc(iInvalid);
redisplay.Lines.Add(arrDesc[iNum]+'
+arrCodes[iNum]+' <---
Invalid Code');
end; //end if
end;
redisplay.Lines.Add('');
end;
redisplay.Lines.Add('');
redisplay.Lines.Add('Number of Invalid Codes: ' +
IntToStr(iInvalid));
redisplay.Lines.Add('Number of valid Codes: ' + IntToStr(iNum-
iInvalid));
CloseFile(TextF);
end;
procedure TForm2.FormCreate(Sender: TObject);
begin
redisplay.Clear;
end;
end.

```

[58]

```

End;
redDisplay.Lines.add('The number of Bedroom Units is : '
+IntToStr(SU));
redDisplay.Lines.add('The number of Bedroom Suites is : '
+IntToStr(INum-SBU));
redDisplay.Lines.add('The Total number is : ' +IntToStr(INum));

function TForm2.cdvalid(scode: String; cdesc:char): Boolean;
var k : Integer;
begin
result := True;
if scode[3] <> cdesc then result:=false
else
begin
if length(scode) <> 7
then result := false
else begin
for k := 1 to 5 do
if not(scode[k] in ['A'..'Z']) then result := false;
for k := 6 to 7 do
if not(scode[k] in ['0'..'9']) then result := false;
end;
end; //end else
end;
//-----
procedure TForm2.loadfile;
var TextF: TextFile;
Name, sline, scode : String;
iCommasPos, iInvalid : Integer;
begin
if fileexists('furniture.txt') = False then
begin
MessageDlg('Sorry data file not located - application will now
terminate ', mtError,mbOk, 0);
Application.Terminate;
end;
iNum := 0;
iInvalid := 0;
AssignFile(TextF, 'furniture.txt');
Reset(TextF);
while not eof(TextF) do
begin
inc(iNum);
Readln(TextF, sline);
iCommasPos := Pos(',', sline);
sName := Copy(sline, 1, iCommasPos - 1);
Delete(sline, 1, iCommasPos);
sCode:=sline;
arrCodes[iNum] := scode;
arrDesc[iNum] :=sName;
if cdvalid(scode,sName[1]) //test for valid
then begin
redDisplay.Lines.Add(arrDesc[iNum] + ' ' +arrCodes[iNum] + '<----Valid
Code' );
end;
end;

```

## VRAAG 2: DELPHI-PROGRAMMERING

## VOORBEELD DELPHI OPLOSSING VRAAG 2

```

unit Q_2;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
  Forms,
  Dialogs, StdCtrls, ComCtrls;
type
  TForm2 = class(TForm)
    redisplay: TRichEdit;
    Button1: TButton;
    Categories: TButton;
    procedure FormCreate(Sender: TObject);
    procedure ButtonClick(Sender: TObject);
    procedure CategoriesClick(Sender: TObject);
  private
    { Private declarations }
    function IsValid(scode: String; nchar:String): Boolean;
    procedure LoadFile;
  public
    { Public declarations }
  end;
var
  Form2: TForm2;
  arrCodes: Array[1..35] of string;
  arrDesc: Array[1..35] of string;
  iNum: integer;
implementation
{$R *.dfm}
-----
procedure TForm2.ButtonClick(Sender: TObject);
begin
  LoadFile;
end;
procedure TForm2.CategoriesClick(Sender: TObject);
var
  iCount, SBU : Integer;
  slist , scateg : String;
begin
  LoadFile;
  redisplay.Clear;
  SBU := 0;
  redisplay.Lines.Add('List of furniture type represented: ');
  redisplay.Lines.Add(' ');
  For iCount := 1 to iNum do
  begin
    scateg := Copy (arrCodes[iCount], 1, 2);
    If scateg='BU' then
      begin
        SBU:=SBU+1;
      end;
  end;
end;

```

```
procedure TForm1.FormCreate(Sender: TObject);
```

```
begin
```

```
//clear all the edits
```

```
edtName.Clear;
```

```
edtSur.Clear;
```

```
edtTitle.Clear;
```

```
edtGross.Clear;
```

```
edtGross.Clear;
```

```
edtMaid.Clear;
```

```
edtFood.Clear;
```

```
edtAcco.Clear;
```

```
edtLoan.Clear;
```

```
end;
```

```
end.
```

[34]

```
{ $R *.dfm }
// Function to calculate the Income Tax and Net Salary
function NetSalary(msgross:string):real;
var
  YGross, mrgross:real;
begin
  mrgross:=strtofloat(msgross);
  YGross:=mrgross*12/-67134.00;
  if (YGross>0) and (YGross<165600.00) then
    NetSalary:=mrgross - YGross*0.18/12
  else
    if (YGross>165001.00) and (YGross<258750.00) then
      NetSalary:=mrgross - mrgross*0.25
    else
      if (YGross>258751.00) and (YGross<358110.00) then
        NetSalary:=mrgross - mrgross*0.30
      else
        if (YGross>358111.00) and (YGross<500940.00) then
          NetSalary:=mrgross - mrgross*0.35
        else
          if (YGross>500941.00) and (YGross<638600.00) then
            NetSalary:=mrgross - mrgross*0.38
          else
            if (YGross>638601) then
              NetSalary:=mrgross - mrgross*0.40;
            end;
procedure TForm1.Button1Click(Sender: TObject);
var
  NetSal, famSal, totSal, totExp, tloan:real;
begin
  redDisplay.Clear;
  //Set the height of the form
  Form1.ClientHeight:=417;
  //Calculate the salaries
  NetSal:=NetSalary(edtGross.Text);
  famSal:=NetSal+NetSalary(edtGross.Text);
  //Calculate the Total Expenses
  totExp:=strtofloat(edtMaid.Text) + strtofloat(edtAcco.Text) +
  strtofloat(edtFood.Text);
  //Calculate the money available for the loan repayment
  totCash:=famSal-totExp;
  tloan := strtofloat(edtLoan.Text);
  mval:=totCash-(tloan/24);
  //Display
  redDisplay.Lines.Add(edtTitle.Text+' '+edtName.Text+' '+
edtSur.Text);
  redDisplay.Lines.Add('Your Monthly Net Salary after Tax
deduction is '+floattostr(NetSal));
  redDisplay.Lines.Add('Your Family's Monthly Net Salary after Tax
deduction is '+floattostr(famSal));
  redDisplay.Lines.Add('After expenses your income is '+
floattostr(totCash));
  if mval > 3000 then
    redDisplay.Lines.Add('Monthly Repayment of
'+floattostr((tloan/24))+ 'You qualify for the loan')
  else
    redDisplay.Lines.Add('Monthly Repayment of
'+floattostr((tloan/24))+ 'You do not qualify for the loan');
end;
```

## VRAAG 1: DELPHI-DATABASIS KONNEKTIVITEIT

## Punteoekening

## VOORBEELD DELPHI OPLOSSING VRAAG 1

```

unit Unit1;

interface

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

type
  TForm1 = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    edtName: TEdit;
    edtSur: TEdit;
    Label3: TLabel;
    edtGross: TEdit;
    Label4: TLabel;
    edtSGross: TEdit;
    Label5: TLabel;
    edtMaid: TEdit;
    Label6: TLabel;
    edtFood: TEdit;
    Label7: TLabel;
    edtAcco: TEdit;
    Label8: TLabel;
    edtLoan: TEdit;
    Button1: TButton;
    BitBtn1: TBitBtn;
    Label10: TLabel;
    edtTitle: TEdit;
    edtDisplay: TRichEdit;
    Button2: TButton;
    procedure Button1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
  totCash: real;

implementation

//uses Unit2;

```

Hierdie memorandum bestaan uit 12 bladsye.

PUNTE: 150

**INLIGTINGSTEKNOLOGIE V1  
MEMORANDUM**

**NOVEMBER 2014**

**GRAAD 11**

**NASIONALE  
SENIOR SERTIFIKAAT**

