



Province of the
EASTERN CAPE
EDUCATION

**NASIONALE
SENIOR SERTIFIKAAT**

GRAAD 12

SEPTEMBER 2014

**INLIGTINGSTEKNOLOGIE V1
MEMORANDUM**

PUNTE: 120

Hierdie memorandum bestaan uit 22 bladsye.

VRAAG 1: DELPHI-PROGRAMMERINGPuntetoekenning

Vraag 1 - Merkrubriek				
V	Aspek		Maks. punte	Leerder se punte
1.1	1.1.1	Function CCValidate(creditcardnumber:string):boolean;	2	
	1.1.2	CleanCardNumber:=""; digit:=0; // Remove any non numeric value for I := 1 to Length(CreditCardNumber) do ✓ Begin if CreditCardNumber[i] in DigitsAllowed then ✓ ✓ CleanCardNumber:= CleanCardNumber + CreditCardNumber[i]; ✓ ✓ End;	5	
	1.1.3	// Check for valid card maximum length number if (Length(CleanCardNumber)>MaxCCSize) then ✓ Begin Result:= True; Exit; End; //Check for valid card minimum length number if (Length(CleanCardNumber)<MinCCSize) then ✓ Begin Result:= False; ✓ Exit; End;	3	
	1.1.4	Carrying out the Luhn Algorithm, the student may use any method as long as they accomplish the task, iterations, loops and decision making, rearranging techniques must be applied.	11	
	1.1.5	procedure TForm1.Button1Click(Sender: TObject); begin If CCValidate(Edit1.text)=true ✓ ✓ then Showmessage('Your Card is Valid') ✓ Else Showmessage('Card Number is Invalid'); ✓ end;	4	

1.2	1.2.1	Type TArr=Array[1..10] integer; Var Form1:TForm1; ListA, ListB: TArr; ✓ ----- Begin Randomize; For i:= 1 to 10 do ✓	4	
		Begin ListA[i]:= random(10)+10; ✓ Richedit1.Lines.Add(IntToStr(ListA[i])); ✓		
	1.2.2	Var K,X, Y,iCntP :integer; Dup : Boolean; begin iCntP := 0; For X := 1 to 10 do ✓ begin Y := X + 1; Dup := False; While (Dup = False) AND (Y <= 10) DO ✓ begin If ListA[X] = ListA[Y] then DUp := True; ✓ y := Y + 1; end; If Not(Dup) ✓ then begin iCntP := iCntP + 1; ListB[iCntP] := ListA[X]; ✓ end; end; For K:=1 to iCntP do d.Lines.Add(IntToStr(ListB[K])); ✓ end;	6	
			[35]	

VOORBEELD: DELPHI-OPLOSSING VRAAG 1

```
unit Q_1;

interface

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

type
  TForm1 = class(TForm)
    Edit1: TEdit;
    Button1: TButton;
    lblDisplay: TLabel;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

function CCValidate(CreditCardNumber:string):boolean;

Const
  DigitsAllowed = ['0'..'9'];
  MaxCCSize = 19;
  MinCCSize = 13;

var
  i : integer;
  CleanCardNumber: string;
  digit: Integer;
  CheckSum: Integer; { Holds the value of the operation }
  Flag: Boolean; { used to indicate when ready }
  Counter: Integer; { index counter }
  PartNumber: string; { used to extract each digit of number }
  Number: Integer; { used to convert each digit to integer}

Begin
  CleanCardNumber:="";
  digit:=0;

  // Remove any non numeric value
  for I := 1 to Length(CreditCardNumber) do
  Begin
    if CreditCardNumber[i] in DigitsAllowed then
      CleanCardNumber:= CleanCardNumber + CreditCardNumber[i];
    End;
```

```
// Check for valid card maximum length number
if (Length(CleanCardNumber)>MaxCCSize) then
Begin
Result:= False;
Exit;
End;
//Check for valid card minimum length number
if (Length(CleanCardNumber)<MinCCSize) then
Begin
Result:= False;
Exit;
End;

// get the starting value for our counter
Counter := Length(CleanCardNumber);
Checksum := 0;
PartNumber := '';
Number := 0;
Flag := true;

while (Counter >= 1) do
begin
// get the current digit
PartNumber := Copy(CleanCardNumber, Counter, 1);
Number := StrToInt(PartNumber); // convert to integer
if (Flag) then // only do every other digit
begin
Number := Number * 2;
if (Number >= 10) then
Number := Number - 9;
end;
Checksum := Checksum + Number;

Flag := not (Flag);

Counter := Counter - 1;
end;

result := ((Checksum mod 10) = 0);

End;

procedure TForm1.Button1Click(Sender: TObject);
begin
If CCValidate(Edit1.text)=true
then
Showmessage('Your Card is Valid' )
Else
Showmessage('Card Number is Invalid');

end;

procedure TForm1.GenerateClick(Sender: TObject);
var
i:integer;
```

```
begin
Randomize;
for i := 1 to 10 do
begin
  ListA[i]:= random(10)+10;
  Richedit1.Lines.Add(IntToStr(ListA[i]));
end;
end;

procedure TForm1.DisplayClick(Sender: TObject);
Var
K,X, Y,iCntP :integer;
Dup : Boolean;
begin

iCntP := 0;
For X := 1 to 10 do
begin
  Y := X + 1;
  Dup := False;
  While (Dup = False) AND ( Y <= 10) DO
  begin
    If ListA[X] = ListA[Y] then DUp := True;
    y := Y + 1;
  end;
  If Not(Dup)
  then begin
    iCntP := iCntP + 1;
    ListB[iCntP] := ListA[X];
  end;
end;

For K:=1 to iCntP do
  d.Lines.Add(IntToStr(ListB[K]));

end;

end.
```

VRAAG 2: DELPHI OOP-PROGRAMMERING

Vraag 2 - Merkrubriek				
V	Aspek		Maks. punte	Leerder se punte
2.1	2.1.1	Defining of the class TDriver TDriver = Class(TObject) ✓	1	
	2.1.2	Private fields fName:string; ✓ fCountry:String; ✓ fTeam:string; ✓ fTime:real; ✓	4	
	2.1.3	Defining the parameterised constructor correctly: constructor TDriver.Create(sName, sCountry, sTeam,sTime: string); ✓✓ Receiving values from the parameters and equating them to the private fields: fName:=sName; ✓ fCountry:=sCountry; ✓ fTeam:=sTeam; ✓ fTime:=sTime; ✓ Adding the appropriate Uses statement in the interface section of the program uses SysUtils, StdCtrls, ComCtrls; ✓	7	
	2.1.4	Declaring The method correctly function TDriver.getfName: string; ✓ Getting its value correctly Result:=copy(fName,1, pos(' ',fName)-1) ✓✓	3	
	2.1.5	Declaring The method correctly, can use either string or real as long as learner uses it appropriately in the program function TDriver.getAverage: string; ✓ Might use an internal variable to calculate the average sAverage,aTime:real; Calculation and result done correctly rounding to nearest whole number aTime:=3600* strtoint(copy(fTime,1,1))+60*strtoint(copy(fTime,3,2))+strtofloat(copy(fTime,6,6)); ✓ sAverage:=(58*5.95*1000)/aTime; ✓ Result:='The Average speed of '+fName+ '✓ is '+floattostr(sAverage) +'m/s'; ✓	5	

2.1.6	Declaring The method correctly <pre>function TDriver.toString: string; ✓</pre> Getting its value correctly <pre>shrs:=strtoint(copy(fTime,1,1)); ✓ smins:=strtoint(copy(fTime,3,2)); ✓ sseconds:=strtofloat(copy(fTime,6,6)); ✓ result := fName + #9+fCountry+#9+fTeam+#9+inttostr(shrs)+':'+in ttostr(smins)+'":'+floattostr(sseconds); ✓✓</pre>	6	
2.2.1	<pre>uses Datam; ✓ AUDatam.ADOTaugp3.Active:=True; ✓</pre>	2	
2.3.1	<u>Code on the Button</u> <pre>uses Q2_aurgpunit; ✓ Q2_aurgp2.Showmodal; ✓</pre> <u>On the Form</u> <pre>AUDatam.ADOTaugp3.Active:=True; ✓ AUDatam.ADOTaugp3.Sort:='Time'; ✓</pre>	4	
2.3.2	<u>Reading the database</u> <pre>AUDatam.ADOTaugp3.Open; ✓ AUDatam.ADOTaugp3.First; ✓</pre> <u>CREATING THE OBJECT</u> <pre>Winner:=Tdriver.Create ✓ ✓ (AUDatam.ADOTaugp3['Name'], ✓ AUDatam.ADOTaugp3['Country'], ✓ AUDatam.ADOTaugp3['Team'], ✓ AUDatam.ADOTaugp3['Time']); ✓ showmessage(Winner.toString); ✓✓</pre>	10	
2.4.1	<pre>Uses Q2_aurgp3unit; ✓ Q2_aurgp3unit;.showmodal; ✓</pre>	2	
2.4.2	Placing all the components required	3	

2.4.3	<u>Call Procedure load and Procedure heading:</u> begin if ((Button = nbFirst) or (Button = nbNext) ✓✓ or (Button = nbPrior) or (Button = nbLast)) ✓✓ then begin Mdriver:=Tdriver.Create(DBedit2.Text, ✓✓ DBedit3.Text, ✓ DBedit4.Text✓, DBedit5.Text); ✓ lblavgdisp.Caption:=Mdriver.getAverage; ✓✓ end; end;	11	
2.5	<u>Call the procedure load and procedure heading</u> begin var myFile:textfile; DrvID , Name, Country : String; iTotal : Integer; begin AssignFile(myFile,'Drivers.txt'); ✓ Rewrite(myFile); ✓ Writeln(myFile,'DRIVERS AND THEIR COUNTRIES':50); ✓ Writeln(myFile,'=====':50); ✓ Writeln(myFile,''); Writeln(myFile,'DriverID':20, 'Name and Surname':30, 'Country':20); ✓✓ Writeln(myFile,'-----':20, '-----': ':30, '-----':20); Writeln(myFile,''); AUDatam.ADOTaugp3.Open; ✓ AUDatam.ADOTaugp3.First; ✓ while not AUDatam.ADOTaugp3.Eof do✓ begin DrvID:=AUDatam.ADOTaugp3['DRID']; ✓ Name:= AUDatam.ADOTaugp3['Name']; ✓ Country:= AUDatam.ADOTaugp3['Country']; ✓ if Country <>'Great Britain' then✓ Writeln(myFile,DrvID:20,Name:30,Country:20); ✓ AUDatam.ADOTaugp3.Next; ✓ end; CloseFile(myFile); ✓	16	
		[74]	

VOORBEELD: DELPHI-OPLOSSING VRAAG 2

Object Class (Games_U)

```

unit clsDriver;

interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
  Forms,
  Dialogs, StdCtrls, ComCtrls, Buttons;
type
//=====
=====
TDriver = Class(TObject)
  Private
    fName:string;
    fCountry:String;
    fTeam:String;
    fTime:string;
  Public

    Constructor Create(sName,sCountry,sTeam,sTime:string);
    Function toString:string;
    Function getfName:string;
    Function getAverage:string;

End;

implementation

{ TDriver }

//=====
=====
constructor TDriver.Create(sName, sCountry, sTeam,sTime: string);
begin
  fName:=sName;
  fCountry:=sCountry;
  fTeam:=sTeam;
  fTime:=sTime;
end;
//=====
=====
function TDriver.getAverage: string;
Var
  sAverage,aTime:real;
begin
  aTime:=3600*
  strtoint(copy(fTime,1,1))+60*strtoint(copy(fTime,3,2))+strtofloat(copy(fTi
me,6,6));
  sAverage:=(58*5.95*1000)/aTime;
  Result:='The Average speed of '+fName+ ' is '+floattostr(sAverage)
+'m/s';
end;

//=====
=====
function TDriver.getfName: string;
Var
  cpos:integer;
begin
  cpos:=pos(' ',fName);
  Result:=copy(fName,1,cpos-1);

```

end;

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

```
function TDriver.toString: string;
```

```
var
```

```
shrs, smins:integer;
```

```
sseconds:real;
```

```
begin
```

```
    shrs:=strtoint(copy(fTime,1,1));
```

```
    smins:=strtoint(copy(fTime,3,2));
```

```
    sseconds:=strtofloat(copy(fTime,6,6));
```

```
result :=fName
```

```
+ #9+fCountry+#9+fTeam+#9+inttostr(shrs)+':'+inttostr(smins)+'":'+floattost  
r(sseconds);
```

```
end;
```

```
end.
```

Main Form

```

unit Q2_aurgp;

interface

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

type
  TfrmAustralia = class(TForm)
    btnDisplay: TButton;
    btnResults: TButton;
    BitBtn1: TBitBtn;
    DBGrid1: TDBGrid;
    btnNavigate: TButton;
    btnSave: TButton;
    procedure btnDisplayClick(Sender: TObject);
    procedure btnResultsClick(Sender: TObject);

    procedure btnNavigateClick(Sender: TObject);
    procedure btnSaveClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }

  end;

var
  frmAustralia: TfrmAustralia;
implementation

uses Q2_aurgpunit, Q2_aurgp3unit;

{$R *.dfm}

//=====
=====
procedure TfrmAustralia.btnDisplayClick(Sender: TObject);
begin
  AUDatam.ADOTaugp3.Active:=True;

end;

//=====
=====
procedure TfrmAustralia.btnResultsClick(Sender: TObject);
begin
  Q2_aurgp2.Showmodal;
end;

procedure TfrmAustralia.btnSaveClick(Sender: TObject);
var
  myFile:textfile;
  DrvID , Name, Country : String;
  iTotat : Integer;
begin
  AssignFile(myFile, 'Divers.txt');
  Rewrite(myFile);

```

```

Writeln(myFile, 'DRIVERS AND THEIR COUNTRIES':50);
Writeln(myFile, '=====':50);
Writeln(myFile, '');
Writeln(myFile, 'DriverID':20, 'Name and Surname':30, 'Country':20);
Writeln(myFile, '-----':20, '-----':30, '-----':20);
Writeln(myFile, '');
  AUDatam.ADOTaugp3.Open;
  AUDatam.ADOTaugp3.First;
  while not AUDatam.ADOTaugp3.Eof do
    begin
      DrvID:=AUDatam.ADOTaugp3['DRID'];
      Name:= AUDatam.ADOTaugp3['Name'];
      Country:= AUDatam.ADOTaugp3['Country'];
      if Driver.Country <>'Great Britain' then
        Writeln(myFile, Driver.Getname + #9 + Driver.getcountry);
      AUDatam.ADOTaugp3.Next;
    end;
  CloseFile(myFile);

end;

//=====
=====
procedure TfrmAustralia.btnNavigateClick(Sender: TObject);
begin
  Q2_aurgp3.Showmodal;
end;

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

end.

```

Form2// Button Display Race Results

```
unit Q2_aurgpunit;

interface

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

type
  TQ2_aurgp2 = class(TForm)
    BitBtn1: TBitBtn;
    DBGrid1: TDBGrid;
    btnWinner: TButton;
    procedure FormActivate(Sender: TObject);
    procedure btnWinnerClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Q2_aurgp2: TQ2_aurgp2;
  Winner: Tdriver;

implementation

{$R *.dfm}

procedure TQ2_aurgp2.btnWinnerClick(Sender: TObject);
begin
  AUDatam.ADOTaugp3.Open;
  AUDatam.ADOTaugp3.First;
  Winner:=Tdriver.Create(AUDatam.ADOTaugp3['Name'],
    AUDatam.ADOTaugp3['Country'],
    AUDatam.ADOTaugp3['Team'],
    AUDatam.ADOTaugp3['Time']);
  showmessage(Winner.toString);

end;

procedure TQ2_aurgp2.FormActivate(Sender: TObject);
begin
  AUDatam.ADOTaugp3.Active:=True;
  AUDatam.ADOTaugp3.Sort:='Time';
end;

end.
```

Form 3// Button Navigate

```

unit Q2_aurgp3unit;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
  Forms,
  Dialogs, Datam, ExtCtrls, DBCtrls, StdCtrls, Mask, Buttons, clsDriver;

type
  TQ2_aurgp3 = class(TForm)
    DBNavigator1: TDBNavigator;
    Label1: TLabel;
    DBEdit1: TDBEdit;
    Label2: TLabel;
    DBEdit2: TDBEdit;
    Label3: TLabel;
    DBEdit3: TDBEdit;
    Label4: TLabel;
    DBEdit4: TDBEdit;
    Label5: TLabel;
    DBEdit5: TDBEdit;
    BitBtn1: TBitBtn;
    lblAverage: TLabel;
    lblavgdisp: TLabel;
    procedure FormActivate(Sender: TObject);
    procedure DBNavigator1Click(Sender: TObject; Button: TNavigateBtn);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Q2_aurgp3: TQ2_aurgp3;
  Mdriver:Tdriver;

implementation

{$R *.dfm}

procedure TQ2_aurgp3.DBNavigator1Click(Sender: TObject; Button:
TNavigateBtn);
begin
  if ((Button = nbFirst) or (Button = nbNext)
    or (Button = nbPrior) or (Button = nbLast)) then
    begin
      Mdriver:=Tdriver.Create(DBedit2.Text,
        DBEdit3.Text,
        DBEdit4.Text,
        DBEdit5.Text);
      lblavgdisp.Caption:=Mdriver.getAverage;
    end;
end;

procedure TQ2_aurgp3.FormActivate(Sender: TObject);
begin
  AUDatam.ADOTaugp3.Active:=True;
end;

end.

```

VRAAG 3: DELPHI-PROGRAMMERINGPuntetoekenning

Vraag 3 - Merkrubriek			
V	Aspek	Maks. punte	Leerder se punt
3.1	<p><u>Introduce a counter to count the number of entries in the file</u> <code>k:=0; ✓</code></p> <p><u>Check to see if file exists and exit if it does not exist</u> <code>if FileExists('abrivs.txt') <> true then ✓</code> Begin ShowMessage('File Does not exist'); Exit; ✓ End;</p> <p><u>Assign and reset file</u> AssignFile(Myfile, 'abrivs.txt'); ✓ Reset(Myfile); ✓</p> <p><u>Start a Loop</u> While Not EOF(Myfile) do ✓ inc(k); ✓</p> <p><u>Read line by line</u> Readln(Myfile, SOneline); ✓</p> <p><u>Enter correct values into array</u> ArrAbrevs[1,k] := copy(SOneline, 1, 3); ✓ ArrAbrevs[2,k] := copy(SOneline, 5, length(SOneline)); ✓</p>	10	
3.2	<p><u>Loop to Display in Richedit</u> For z:=1 to k do ✓ Begin</p> <p><u>Display correctly</u> redAbrevs.Lines.add(ArrAbrevs[1,z]+' '+ ArrAbrevs[2,z]); ✓ end;</p> <p><u>Nested loop to display positions in stringgrid</u> For i:=1 to 12 do ✓ begin For j:=1 to 10 do ✓ begin</p> <p><u>ArrPositions displayed correctly</u> strgdPositions.Cells[j,i] := ArrPositions[j,i]; end; ✓ end;</p> <p><u>Loop to display circuits and numbers</u> For i:=1 to 10 do ✓ Begin</p> <p><u>Circuits and Positions displayed correctly</u> strgdPositions.Cells[0,i] := arrCircuits[i]; ✓ strgdPositions.Cells[i,0] := inttostr(i); End; ✓</p>	8	

3.3	<p><u>Variable to count number of podium appearances to be initialised</u> nump:=0; ✓</p> <p><u>Use inputbox correctly to get values</u> iAbrev:=UpperCase(Inputbox('Driver Podium Search', 'Enter Abbreviations of Driver here', '')); ✓✓</p> <p><u>Start loop to search abbreviation</u> For i:=1 to 25 do✓ Begin</p> <p><u>Link Abbreviation to Name</u> if ArrAbrevs[1,i]=iAbrev then iName:=ArrAbrevs[2,i]; ✓✓ End;</p> <p><u>Search podium positions on stringgrid</u> for i := 1 to 10 do ✓ Begin For j:=1 to 3 do Begin if strgdPositions.Cells[j,i]=iAbrev✓ then inc(nump); ✓ End; End; if nump>0 then✓</p> <p><u>Display Podium podium positions using a message box</u> ShowMessage(iName+' has been on the podium '+inttostr(nump)+' times so far') ✓ Else✓ShowMessage(iName+' has not been on the podium so far') ✓</p>	13	
3.4	<p><u>Use of Input box correct to get Abbreviation</u> iAbrev:=UpperCase(Inputbox('Points Calculator', 'Enter Abbreviations of Driver here', '')); ✓✓</p> <p><u>Start loop to count</u> for j := 1 to 10 do✓ Begin for i:= 1 to 10 do✓ if strgdPositions.Cells[i,j]= iAbrev then</p> <p><u>Correct use of Case statement or If Statement</u> Begin Case i of 1:iPoint:=25; 2:iPoint:=18; 3:iPoint:=15; 4:iPoint:=12; 5:iPoint:=10; ✓ 6:iPoint:=8; ✓ 7:iPoint:=6; 8:iPoint:=4; 9:iPoint:=2; 10:iPoint:=1 Else iPoint:=0; ✓ End;</p>	10	

	<pre> End; <u>Add Total</u> iTotal:=iTotal+iPoint; ✓ End; <u>Link Name to Abbreviation</u> For i:=1 to 25 do Begin if ArrAbrevs[1,i]=iAbrev then iName:=ArrAbrevs[2,i]; ✓ End; <u>Display Points</u> ShowMessage(iName+' has '+inttostr(iTotal)+' points so far'); ✓ end; </pre>		
		[41]	

VOORBEELD: DELPHI-OPLOSSING VRAAG 3

```

unit Results;

interface

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

type
  TfrmSummaries = class(TForm)
    MainMenu1: TMainMenu;
    sRATRT1: TMenuItem;
    Load: TMenuItem;
    Display: TMenuItem;
    Podium: TMenuItem;
    Points: TMenuItem;
    redAbrevs: TRichEdit;
    strgdPositions: TStringGrid;
    Exit1: TMenuItem;
    Label1: TLabel;
    procedure LoadClick(Sender: TObject);
    procedure DisplayClick(Sender: TObject);
    procedure PodiumClick(Sender: TObject);
    procedure PointsClick(Sender: TObject);
    procedure Exit1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmSummaries: TfrmSummaries;
  ArrAbrevs: Array[1..2,1..25] of string;
  ArrPositions: Array[1..10,1..15] of string = (('VET', 'VET', 'HAM',
'ROS', 'ROS', 'ROS', 'VET', 'HAM', 'HAM', 'HAM', 'HAM', 'VET', '',
'', ''),
  ('WEB', 'MAS', 'RAI', 'VET', 'HAM', 'HAM', 'HAM', 'ROS', 'VET',
'VET', 'VET', 'WEB', '', '', '' ),
  ('HAM', 'ALO', 'ALO', 'ALO', 'VET', 'VET', 'BOT', 'VET', 'WEB',
'GRO', 'WEB', 'HUL', '', '', '' ),
  ('MAS', 'HAM', 'ROS', 'MAS', 'RAI', 'WEB', 'ROS', 'WEB', 'RAI',
'ROS', 'ROS', 'MAS', '', '', '' ),
  ('ALO', 'WEB', 'MAS', 'DIR', 'ALO', 'RAI', 'WEB', 'RIC',
'GRO', 'ALO', 'DIR', 'ALO', '', '', '' ),
  ('ROS', 'ROS', 'GRO', 'SUT', 'GRO', 'ALO', 'ALO', 'SUT',
'RIC', 'RAI', 'BUT', 'ROS', '', '', '' ),
  ('RAI', 'BUT', 'RIC', 'WEB', 'WEB', 'PER', 'VER', 'GRO',
'MAS', 'MAS', 'GRO', 'RIC', '', '', '' ),
  ('GRO', 'SUT', 'BUT', 'RAI', 'PER', 'SUT', 'SUT', 'RAI',
'ALO', 'RIC', 'RAI', 'PER', '', '', '' ),
  ('DIR', 'PER', 'VET', 'HAM', 'MAS', 'BUT', 'HUL', 'ALO',
'BUT', 'PER', 'ALO', 'BUT', '', '', '' ),
  ('BUT', 'RAI', 'HUL', 'BUT', 'DIR', 'VER', 'RAI', 'BUT',

```

```
'HUL', 'WEB', 'MAS', 'VER', '', '', ''));
  ArrCircuits: Array[1..12] of string = ('Rolex Australian Grand
  Prix','Petronas Malaysia Grand Prix','UBS Chinese Grand Prix','Gulf
  Air Bahrain Grand Prix','57° Gran Premio de España','71e Grand Prix
  de Monaco','Grand Prix du Canada','Santander British Grand
  Prix','Grosser Preis Santander von Deutschland','Magyar
  Nagydíj','Shell Belgian Grand Prix','84° Gran Premio d'Italia');
  k:integer;
```

```
implementation
```

```
{ $R *.dfm }
```

```
//=====
=====
procedure TfrmSummaries.Exit1Click(Sender: TObject);
begin
  Application.Terminate;
end;
procedure TfrmSummaries.FormCreate(Sender: TObject);
begin
  strgdPositions.ColWidths[0]:=220;
end;

//3.1=====
=====
procedure TfrmSummaries.LoadClick(Sender: TObject);
Var
  Myfile:TextFile;
  i:integer;
  SOnline:string;
begin
  k:=0;
  if FileExists('abrivs.txt')<> true then
    Begin
      ShowMessage('File Does not exist');
      Exit;
    End;
    AssignFile(Myfile,'abrivs.txt' );
    Reset(Myfile);
    While Not EOF(Myfile) do
      Begin
        inc(k);
        Readln( Myfile, SOnline);
        ArrAbrevs[1,k]:=copy(SOnline,1,3);
        ArrAbrevs[2,k]:=copy(SOnline,5,length(SOnline));

        End;
  End;
```

```
End;
```

```
//3.2=====
=====
```

```

procedure TfrmSummaries.DisplayClick(Sender: TObject);
  Var
    z,i,j:integer;
begin
  For z:=1 to k do
    begin
      redAbrevs.Lines.add(ArrAbrevs[1,z]+' '+ ArrAbrevs[2,z]);
    end;

    For i:=1 to 12 do
      begin
        For j:=1 to 10 do
          begin
            strgdPositions.Cells[j,i]:=ArrPositions[j,i];
          end;
        end;
      end;
    For i:=1 to 10 do
      Begin
        strgdPositions.Cells[0,i]:= arrCircuits[i];
        strgdPositions.Cells[i,0]:= inttostr(i);
      End;
    end;

end;

//3.3=====
=====
procedure TfrmSummaries.PodiumClick(Sender: TObject);
  Var
    i,j,nump:integer;
    iAbrev,iName:string;
begin
  nump:=0;
  iAbrev:=UpperCase(Inputbox('Driver Podium Search','Enter
Abbreviations of Driver here',''));

  For i:=1 to 25 do
    Begin
      if ArrAbrevs[1,i]=iAbrev then
        iName:=ArrAbrevs[2,i];
      End;
    end;

for i := 1 to 10 do
  Begin
    For j:=1 to 3 do
      Begin
        if strgdPositions.Cells[j,i]=iAbrev then
          inc(nump);
        End;
      end;
    End;

    if nump>0 then

      ShowMessage(iName+' has been on the podium '+inttostr(nump)+'
times so far')

```

```

Else
  ShowMessage(iName+' has not been on the podium so far')

end;
//3.4=====
=====
procedure TfrmSummaries.PointsClick(Sender: TObject);
Var
  i,j,iPoint,iTotal:integer;
  iName,iAbrev:string;

begin

  iAbrev:=UpperCase(Inputbox('Points Calculator','Enter
Abbreviations of Driver here',''));
  for j := 1 to 10 do
    Begin
      for i:= 1 to 10 do
        if strgdPositions.Cells[i,j]= iAbrev then
          Begin
            Case i of
              1:iPoint:=25;
              2:iPoint:=18;
              3:iPoint:=15;
              4:iPoint:=12;
              5:iPoint:=10;
              6:iPoint:=8;
              7:iPoint:=6;
              8:iPoint:=4;
              9:iPoint:=2;
              10:iPoint:=1
            Else
              iPoint:=0;
            End;
          End;
        iTotal:=iTotal+iPoint;
      End;
      For i:=1 to 25 do
        Begin
          if ArrAbrevs[1,i]=iAbrev then
            iName:=ArrAbrevs[2,i];
          End;
        ShowMessage(iName+' has '+inttostr(iTotal)+' points so far');
      end;

end.

```

TOTAAL: 150