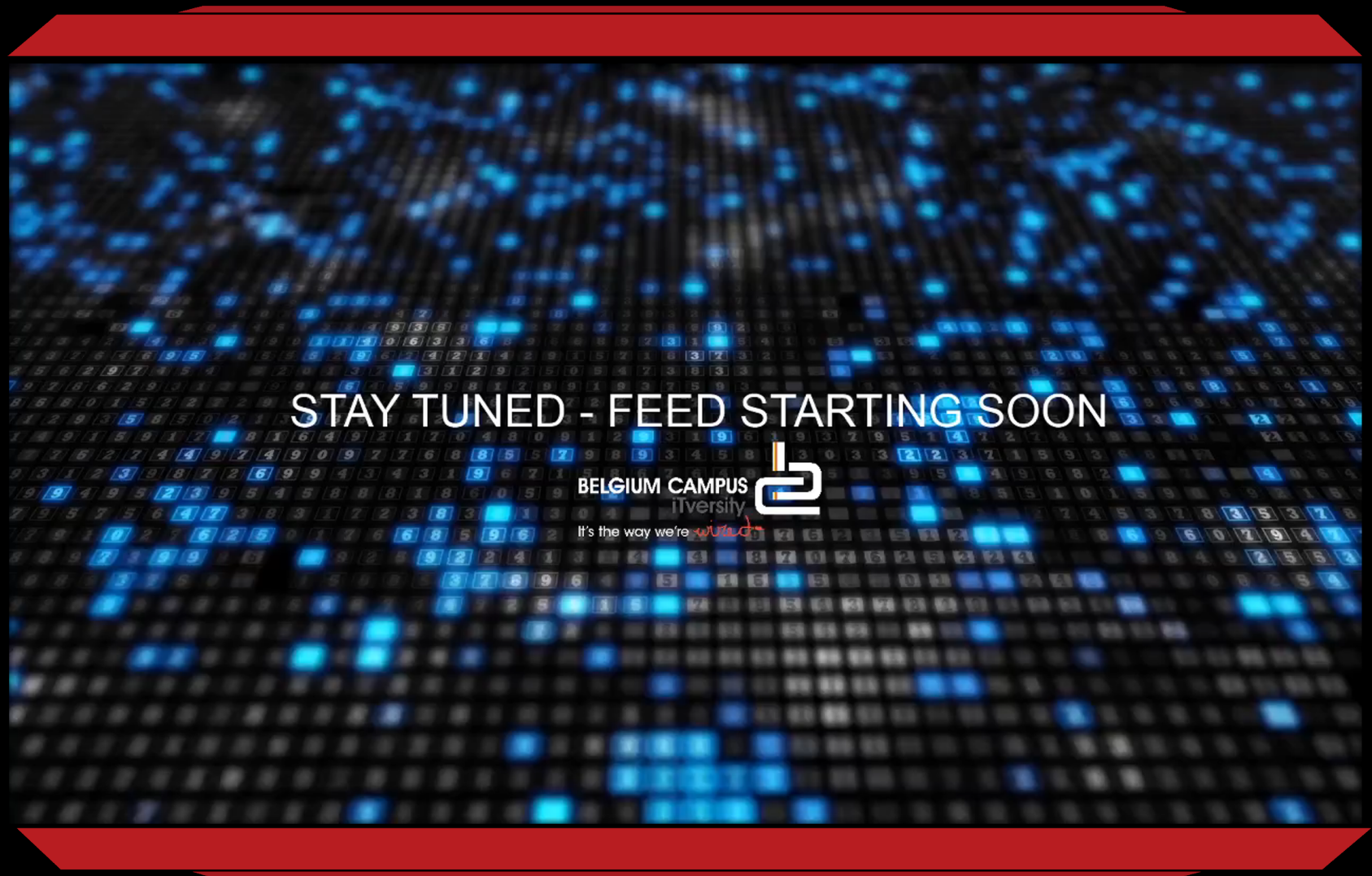

STAY TUNED - FEED STARTING SOON
BELGIUM CAMPUS
iTversity
It's the way we're wired

BELGIUM CAMPUS iTversity

It's the way we're wired

DELPHI – TEXT FILES

V. Pretorius

LESSON OBJECTIVES

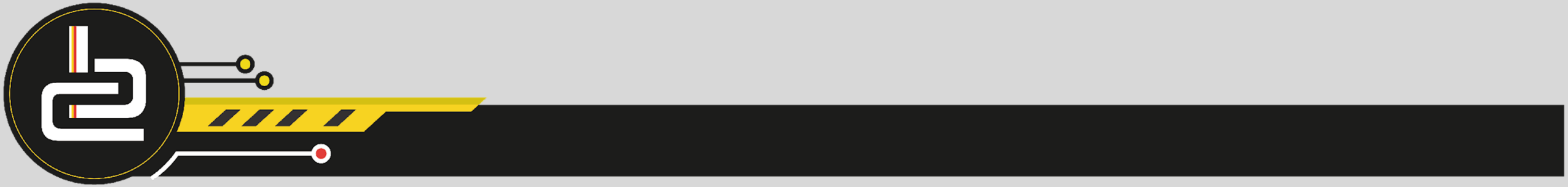
- Use of Arrays, Text files and Text-based reports
- Exception handling and Text files

AIM AND PRE-KNOWLEDGE

Aim is to understand the skills to ..

- ❑ Input and output data using a text file
- ❑ Understand conditional iteration with a text file
- ❑ Produce text-based reports from a text file
- ❑ Solution development using text files

String manipulation functions and procedures is a pre-requisite for this section of work. Eg copy, pos, delete, insert, uppercase and upcase.



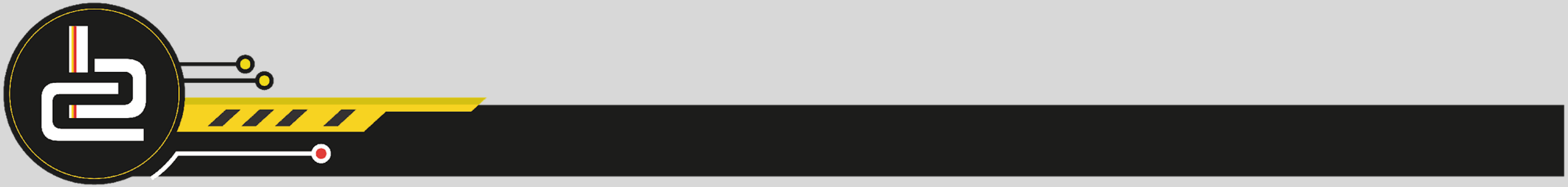
USE OF ARRAYS, TEXT FILES AND TEXT-BASED REPORTS

In order to continue with this slide the basic concepts of an Array must have been studied.

Certain applications need many lines of data to be imported into a data structure eg Array, where the lines can be searched, sorted and reports can be printed.

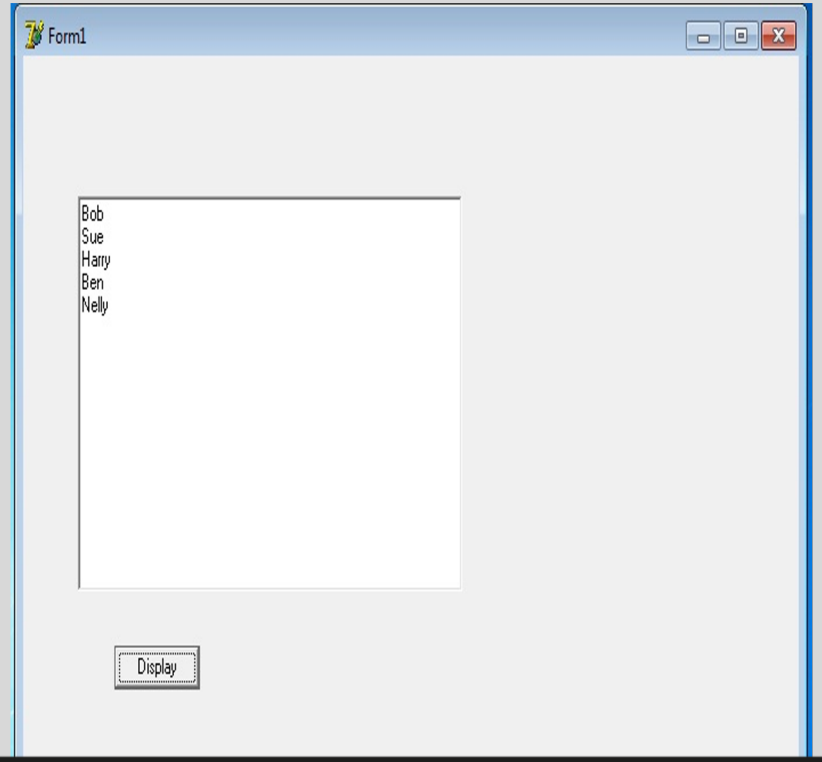
Algorithm used to input data from an existing text file into an array:

1. **Declare the variables Eg VAR MyFile : TextFile;**
   arrFriends:Array[1..10] of string;
   iIndex : integer;
2. **Link the text variable to the external file**
3. **Test if the file exists, if not exit / terminate the project**
4. **If the file exists open the file to read**



ARRAYS AND TEXT FILES CONTINUED ...

5. Initialise iIndex.
6. Use a conditional loop that will end when the end of file (boolean) function returns true
    a. Read the line and transfer the data into the array at position iIndex.
    b. Increment iIndex
7. Close the text file, flag the memory space as unused.
8. Display the contents of the array using a loop.



ARRFRIENDS DELPHI EXAMPLE

```
procedure TForm1.BtnDisplayClick(Sender: TObject);
VAR
    TFile : Textfile;                          {1}
    ArrNames : ARRAY[1..10] of string;
    iCount, k: integer;

begin
  Assignfile(Tfile, 'Friends.txt');            {2}

  If fileExists(‘Friends.txt’) <> true         {3}
  then exit
  else RESET(TFile);                           {4}
```

ARRFRIENDS DELPHI EXAMPLE CONTINUED ...

```
iCount := 1;                                {5}

WHILE NOT EOF(TFile) DO                     {6}
  BEGIN
    ReadLn (TFile, ArrNames[iCount]);       {6a}
    inc(iCount);                            {6b}
  END;
Closefile(TFile);                           {7}

For k := 1 to iCount -1 do                  {8}
    RedDisplay.Lines.add(ArrNames[k]);
end;
```

VALUES ASSIGNED BY READING IT FROM A TEXT FILE.

Used when:

working with a huge amount of data,

testing a program with test data,

it will be time consuming to enter all the values into the array

GLOBAL Declaration:

```
{private declarations of the form}
   arrSurname : Array[1..100] of String;
   Count      : Integer;
```

```
Procedure TForm1.btnTextFileToArrayClick(Sender : Tobject);
VAR
   TF : Textfile;
Begin
  AssignFile(TF, 'Data.txt');
  Reset(TF); Count := 0;
  While Not EOF(TF) AND (Count < 100) DO
   begin
     inc(Count, 1);
     Readln(TF, arrSurname[Count]);
   End;//While
   CloseFile(TF);
End;
```

The extra condition is to prevent a range error!

The elements in this array depend on the content of the text file: DATA.txt!

LOOPHOLES WHEN USING TEXT FILES.

Be aware of the following when using a text file:

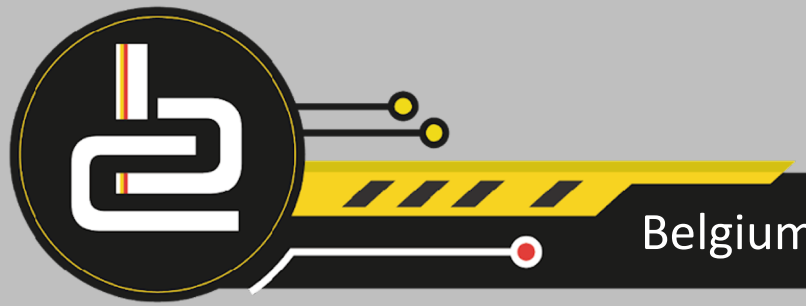
Check if the file exists before continuing with the program.

```
Begin
  IF NOT FileExists('Data.txt')
   then
    begin
      ShowMessage('File not present!');
      //Disable all menu's AND buttons
      Exit;
End;
```

The number of entries in the text file is unknown and may cause a range error if the array is declared with a small upper index.

```
Begin
  While NOT EOF(TF) AND (Count < Max) do
   begin
    //. . .
   end;//while
End;
```

Extra condition to prevent array range error!!



EXERCISE 4

WRITE A DELPHI PROGRAM FOR EACH OF THE FOLLOWING SCENARIOS:

A) Read some numbers from the text file, TESTNUMBERS.TXT, into an array and display a numerically sorted list. Solution

B) Read some names from the text file, TESTNAMES.TXT, into an array and display an alphabetical list. Solution

C.) Read names and corresponding averages from the text file, NAMESAVE.TXT, into 2 arrays. Give the user the option to display: Solution

i) an alphabetical list with corresponding numbers

ii) The top learner according to average Solution

iii) The list of learners in order from best to weakest.

iv) Read a name and an average from the keyboard. Insert this information into the correct alphabetical position in the arrays. Display the new arrays in neat columns.

SOLUTION: EXERCISE 4

SECTIONS:
1. Declaration of arrays & File linking
2. Local procedures used for sorting the arrays.
3. Procedure for 4A
4. Procedure for 4B
5. Procedure for 4C

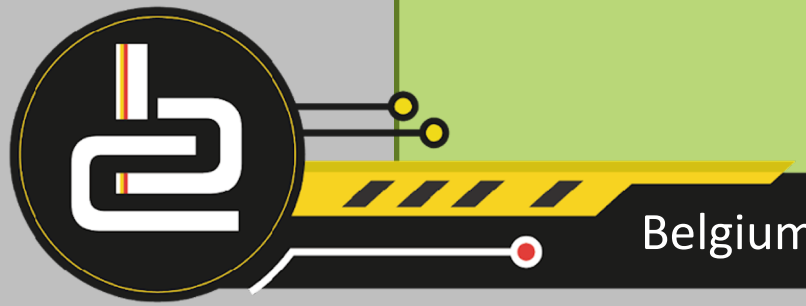
Declaration of Arrays

```
type
  myarray1 = array[1..50] of string;
  myarray2 = array[1..50] of real;

var
  namearray: myarray;
  numberarray: myarray2;
  TF, TF2, TF3 : textfile;
```

Check if files exists and linking them with the Delphi program

```
procedure TForm1.FormActivate(Sender:TObject);
Begin
 assignfile(TF,'TESTNAMES.TXT');
 if Not FileExists('TESTNAMES.TXT')
  then
    begin
      showmessage('File does not exist');
      rewrite(TF);
    end;
  // Use similar AssignFile & IF-statements for
  // the other two text files.
End;
```



SOLUTION: EXERCISE 4

(CONTINUE)

User defined paramaterised procedures for sorting the arrays.

```
procedure sortnames(Var narray:myarray; total:integer);
var i,j   : integer;
    temp : string;
begin
  for i := 1 to total -1 do
    for j:= 1+i to total do
      if  narray[i] > narray[j]
       then
         begin
           temp := narray[i];
           narray[i]:=narray[j];
           narray[j]:= temp;
         end; // swapping the names around
End;
```

Value parameters used to indicate the number of elements in the array.

Reference parameters used to "send back" the sorted array to the calling event.

```
procedure numsort(var narray:myarray2; total:integer);
var i, j, temp: integer;
Begin
  //for swapping the numbers use the same algorithm as above
End;
```

SOLUTION: EXERCISE 4

(CONTINUE)

User defined paramaterised procedures for sorting the arrays.

```
procedure sort2arrays(var onearr:myarray; var twoarr:myarray2; total:integer);
var i,j,temp2:integer;
    temp: string;
begin
  for i := 1 to total -1 do
    for j:= 1+i to total do
      if  namearray[i] > namearray[j]
       then
         begin
          temp := namearray[i];
          namearray[i]:=namearray[j];
          namearray[j]:= temp;

          temp2 := numberarray[i];
          numberarray[i]:=numberarray[j];
          numberarray[j]:= temp2;
          end;
   // swapping the names and numbers around
end;
```

The elements of both arrays must be swapped.

SOLUTION: EXERCISE 4

(CONTINUE)

Question 4.A.

```
procedure TForm1.btnNumClick(Sender: TObject);
var counter, i,j, temp : integer;
 line : string;
begin
 redout.clear;
 reset(TF2);
 counter:=0;
 while not EOF(TF2) do
  // use a while loop when you do not know how many retitions there are
  begin
   inc(counter);
   readln(TF2,line);
   numberarray[counter]:=StrToInt(line);
  end;
  numsort(numberarray,counter);
  redout.Lines.Add('Numerical list of numbers');
        // always display a heading outside the loop
  for i:= 1 to counter do
    redout.lines.Add(IntToStr(numberarray[i]));
        // do not need begin and end as there is only 1 instruction in the loop
  closefile(TF2);
end;
```

Calling the user defined procedure to sort the numbers

SOLUTION: EXERCISE 4

(CONTINUE)

Question 4.B.

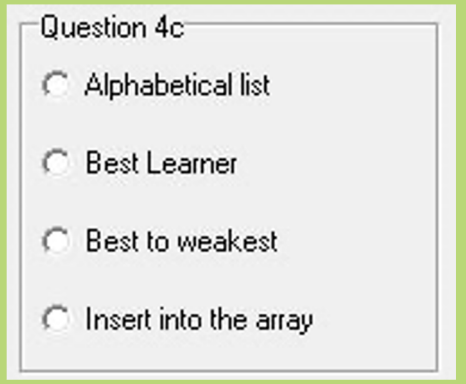
```
procedure TForm1.btnnameClick(Sender: TObject);
var counter, i : integer;
 temp : string;
begin
  redout.clear;
  reset(TF);
  counter:=0;
  while not EOF(TF) do
  // use a while loop when you do not know how many retitions there are
  begin
    inc(counter);
    readln(TF,namearray[counter]);
  end;
  sortnames(namearray,counter);
  redout.Lines.Add('Names in alphabetical order');
        // always display a heading outside the loop
  for i:= 1 to counter do
    redout.lines.Add(namearray[i]);
  // do not need begin and end as there is only 1 instruction in the loop
  closefile(TF);
end;
```

Calling the user defined procedure to sort the names.

SOLUTION: EXERCISE 4

(CONTINUE)

Question 4.C.



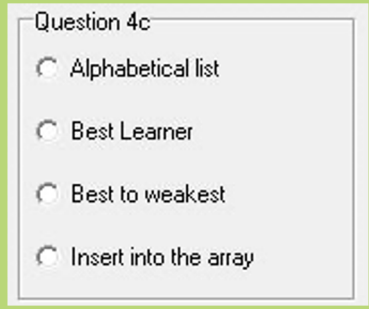

```
procedure TForm1.rgpClick(Sender: TObject);
var inum, counter, i,j,temp2 : integer;
 temp,line,sname : string;
begin
  redout.clear;
  reset(TF3);
  counter:=0;
  while not EOF(TF3) do
  // use a while loop when you do not know how many repetitions
there are
  begin
    inc(counter);
    readln(TF3,line);
    namearray[counter]:= copy(line,1,pos('#',line)-1);
    delete(line,1,pos('#',line));
    numberarray[counter]:=StrToInt(line);
  end;

  // Radiogroup's options continue on next three slides . . . .
  closefile(TF3);
End;
```

Reading an unknown number of elements into the two arrays.

SOLUTION: EXERCISE 4

(CONTINUE)



Question 4.C.

Option > Alpabetical list

```
case rgp.ItemIndex of
  0: begin
       sort2arrays(namearray,numberarray,counter);
       redout.Lines.Add('Names in alphabetical order');
       for i:= 1 to counter do
         redout.lines.Add(namearray[i]+#9+IntToStr(numberarray[i]));
     end;  //end no 0
```

Option > Best learner

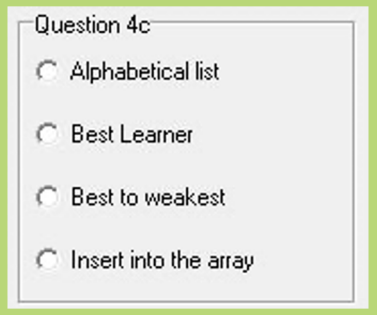
```
  1: begin
      for i := 1 to counter -1 do
       for j:= 1+i to counter do
       if  numberarray[i] < numberarray[j]
        then
         begin
           temp := namearray[i];
           namearray[i]:=namearray[j];
           namearray[j]:= temp;
           temp2 := numberarray[i];
           numberarray[i]:=numberarray[j];
           numberarray[j]:= temp2;
         end; // swapping the names and numbers around
      redout.Lines.Add('Best learner');
      redout.lines.Add(namearray[1]+#9+IntToStr(numberarray[1]));
      end;   //end no1
//continue on next slide . . . .
```

SOLUTION: EXERCISE 4

(CONTINUE)

Question 4.C.

Option > Best to weakest

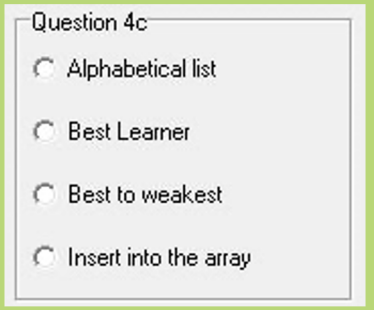



```
2: begin
        for i := 1 to counter -1 do
          for j:= 1+i to counter do
            if  numberarray[i] < numberarray[j]
             then
               begin
                 temp := namearray[i];
                 namearray[i]:=namearray[j];
                 namearray[j]:= temp;
                 temp2 := numberarray[i];
                 numberarray[i]:=numberarray[j];
                 numberarray[j]:= temp2;
               end; // swapping the names and numbers around
         redout.Lines.Add('Best to weakest');
         for i:= 1 to counter do
           redout.lines.Add(namearray[i]+#9+IntToStr(numberarray[i]));
        end;          //end no2


//continue on next slide . . . .
```

SOLUTION: EXERCISE 4

(CONTINUE)



Question 4.C.

Option > Insert into the array

```
3: begin
    sname:=inputbox('Name','type in a name','');
    inum:=strtoint(inputbox('Average','Type in an average','0.0'));
    sort2arrays(namearray,numberarray,counter);
    // inserting into a sorted array
    for i := 1 to counter do
    begin
      if namearray[i] > sname then
      begin
        for j:= counter downto i do
        begin
          namearray[j+1]  := namearray[j];
          numberarray[j+1]:= numberarray[j];
         end; //for j
        namearray[i]   := sname;
        numberarray[i] := inum;
        break;
      end; //for if
   end; //inserting a name into the correct alphabetical position in an array
   redout.Lines.Add('Names in alphabetical order');
   for i := 1 to counter+1 do
      redout.lines.Add(namearray[i]+#9+IntToStr(numberarray[i]));
 end;  // no 3
end;  //end case
```

Sort the two arrays in order to add the new values into the correct position.

Move all elements up to make place for the new element.

The new elements is inserted into the correct position.

When the element is inserted the For loop may be ended.

VALUES ASSIGNED BY READING IT FROM A TEXT FILE.

Used when:

working with a huge amount of data,

testing a program with test data,

it will be time consuming to enter all the values into the array

GLOBAL Declaration:

```
{private declarations of the form}
   arrSurname : Array[1..100] of String;
   Count      : Integer;
```

```
Procedure TForm1.btnTextFileToArrayClick(Sender : Tobject);
VAR
   TF : Textfile;
Begin
  AssignFile(TF, 'Data.txt');
  Reset(TF); Count := 0;
  While Not EOF(TF) AND (Count < 100) DO
   begin
     inc(Count, 1);
     Readln(TF, arrSurname[Count]);
   End;//While
   CloseFile(TF);
End;
```

The extra condition is to prevent a range error!

The elements in this array depend on the content of the text file: DATA.txt!

LOOPHOLES WHEN USING TEXT FILES.

Be aware of the following when using a text file:

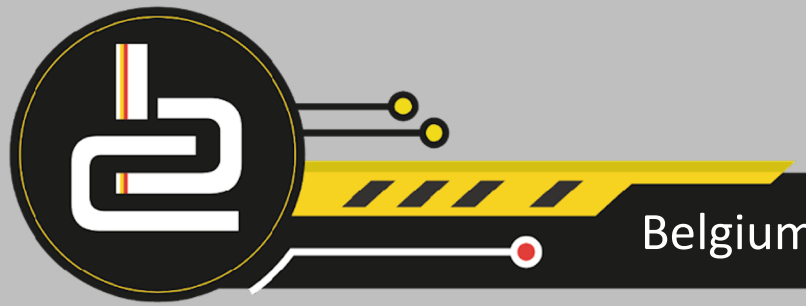
Check if the file exists before continuing with the program.

```
Begin
  IF NOT FileExists('Data.txt')
   then
    begin
      ShowMessage('File not present!');
      //Disable all menu's AND buttons
      Exit;
End;
```

The number of entries in the text file is unknown and may cause a range error if the array is declared with a small upper index.

```
Begin
  While NOT EOF(TF) AND (Count < Max) do
   begin
    //. . .
   end;//while
End;
```

Extra condition to prevent array range error!!



EXERCISE 5

WRITE A DELPHI PROGRAM FOR THE FOLLOWING SCENARIOS:

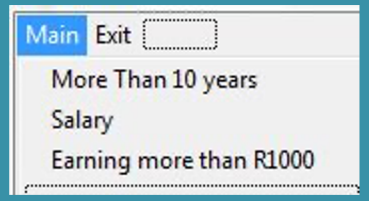
Make provision to read the contents of a text file, WORKERS.TXT.

The information in the text file is as follows:

Name#date of starting#salary category#hours worked

e.g Peter Smith #13/01/2008#A#41

The fields must be separated so that information can be displayed, and calculations can be done.



Make use of the given program and complete all the menu options.

Menu option 1 – Display the names of those workers who have been working for more than 10 years

Menu Option 2 – Display the Salary for each worker – determine the salary as follows:

- Category A – R 430 per hour
- Category B – R 456.33 per hour
- Category C – R 532.09 per hour

Display the name, salary, a column for tax at 25% and the take home pay.

Menu option 3 –Display the names only of those workers whose salary is more than R 1000

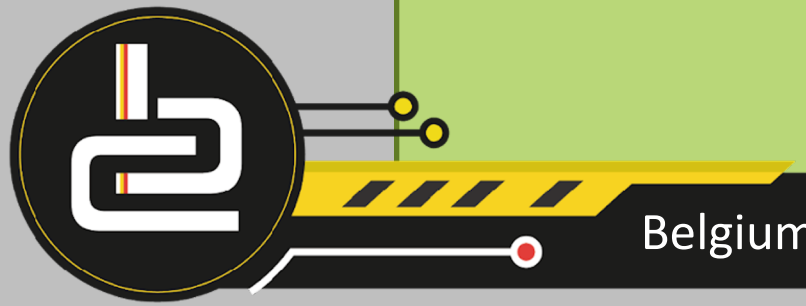
SOLUTION: EXERCISE 5

The content of the text file:

```
Peter#12/01/2008#A#42
John#16/05/2008#B#36
Mary Goodyear#01/11/2009#C#25
Mary Jones#05/05/2010#C#44
```

To split each line of text into the various variables make use of the following:

```
While NOT EOF(TF) DO
   begin
     Readln(TF, sline);
     name := copy(sLine, 1, pos('#', sLine)-1);
     Delete(sLine, 1, pos('#', sLine));
     StartDate := StrToDate(copy(sLine,1 , pos('#', sLine)-1));
     Delete(sLine, 1, pos('#', sLine));
     Cat := sLine[1];
     Delete(sLine, 1, pos('#', sLine));
     Hours := StrToInt(sLine);
   end;
```



SEARCH FOR AN ELEMENT IN AN ARRAY

A search method using a flag.

GLOBAL Declaration:

```
{private declarations of the form}
   arrSurname : Array[1..100] of String;
   Count      : Integer;
```

```
Procedure TForm1.btnSearchArrayClick(Sender : Tobject);
VAR
   a : Integer; SearchFor : String; Found : Boolean;
Begin
  Searchfor := InputBox('Search','Search for?', 'Peters');
  a := 1;    Found := False;
  While NOT Found AND (a <= Count) do
   begin
     if arrSurname[a] = SearchFor
       then Found := True
       else Inc(a, 1);
   end;//while
  IF NOT Found
    then ShowMessage(Searchfor + ' not found in array.');
End;
```

NOTE

You must be able to search with a flag in order to stop the search once the required element is found.

Sometimes the question requires that you search ALL the array elements to compile the report.

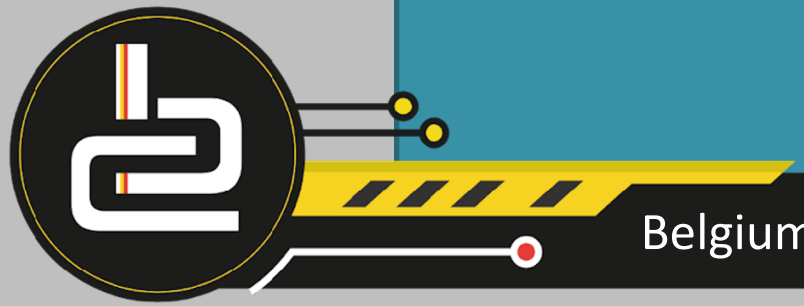
EXERCISE 6

SCENARIO:

The statistics collector is keeping information on the soccer players in the Bafana Bafana team. In each training game that is played over a period of 8 games, the number of errors made by each player is recorded. This will help make decisions about the consistent behaviour on the field of each player. The information is stored in a text file and displayed in a string grid.

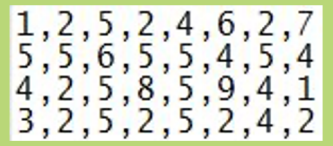
Complete the given Delphi program and follow the instructions.

- Read the information from the text file, BAFANA.TXT, and display it in a string grid.
- The column headings will run from 1 to 8.
- The row labels will be the player's names – also read from a text file, NAMES.TXT.
- Find the total number of errors for each player and display these in the 9th column.
- Find the total number of errors for each match and display these in the next row.



SOLUTION: Exercise 6

Example of the content of the two text files:



Declaration of Arrays

```
type
  myarray= array[1..20,1..10] of string;
var
  TF1,TF2: textfile;
  Bafanarr:myarray;
```

SECTIONS:

1. Declaration of arrays & File linking
2. Reading of data from the textfile into array.
3. Displaying headings and data in grid.
4. Calculating and displaying totals.

SOLUTION: EXERCISE 6

(CONTINUE)

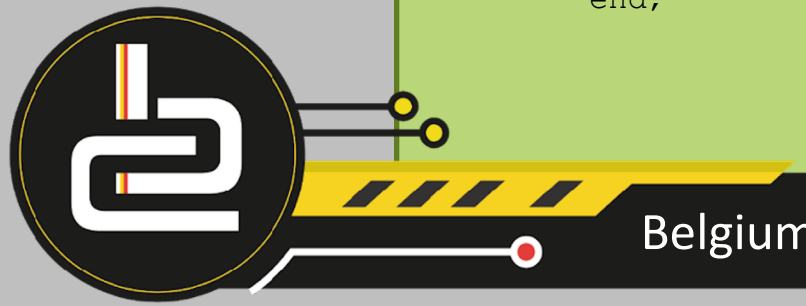
A. Reading of the text file into the 2D array

```
procedure TQuestion6.FormActivate(Sender: TObject);
var row, col : integer;
Begin
  assignfile(TF2,'BAFANA.txt');
  if not fileExists('BAFANA.txt') then exit;
  reset(TF2);
  row:=0;
  while not eof(TF2) do
  begin
    readln(TF2,line); // reads the text file one line at a time
    inc(row);
    for col:= 1 to 7 do
    begin
      bafanarr[row,col]:=copy(line,1,pos(',',line)-1);
      delete(line,1,pos(',',line));
    end;
    bafanarr[row,8]:=line;
    // adds last match data to array
  end;
end;
```

The loop is used to break up the data for each match.

```
1,2,5,2,4,6,2,7
5,5,6,5,5,4,5,4
4,2,5,8,5,9,4,1
3,2,5,2,5,2,4,2
```

The last value in the text line is added because there is no “,” after the last value.



SOLUTION: EXERCISE 6

(CONTINUE)

B. Display the headings & data in the grid

```
procedure TQuestion6.btnReadClick(Sender: TObject);
var row, col: integer;
begin
 stgdisplay.RowCount := 1;
 for col:= 1 to 8 do // displays the game number in the top row
  stgdisplay.Cells[col,0]:= InttoStr(col);
 reset(TF1);
 row:=0;
 while not eof(TF1) do
  begin
    readln(TF1,line);
    inc(row);
    stgdisplay.RowCount := stgDisplay.RowCount+1;
    stgDisplay.Cells[0,row]:=line;
  end;  // displays the names in the first column

  //display the data
  for row := 1 to stgdisplay.RowCount do
     for col := 1 to 8 do
         stgdisplay.Cells[col,row]:= bafanarr[row,col];
end;
```

Initialise the rowcount to 1 (for the heading).

Aaron
Sticks
Mpho
Lucky

The loop is used to add the names of the players from the textfile to the grid.

Note the use of the **RowCount** property of the grid in order to determine how many rows of data needed to be added.

Rowcount starts at 0 (heading), therefor for the data from 1 to count

Note the difference in the indexes for the grid and the array.

SOLUTION: EXERCISE 6

(CONTINUE)

C. Display the totals for individuals

```
procedure TQuestion6.btnCalcrowClick(Sender: TObject);
var coltotal,r:integer;
row,col:integer;
begin
// adds totals for indviduals
stgdisplay.Cells[9,0]:='Errors per person';

r:=stgdisplay.rowcount;
for row := 1 to r-1 do
  begin
    Coltotal:=0;
    for col := 1 to 8 do
      coltotal:=coltotal+strtoInt(bafanarr[row,col]);
    bafanarr[row,9]:= inttoStr(coltotal);
  end;

  for row := 1 to stgdisplay.RowCount do
    stgdisplay.Cells[9,row]:= bafanarr[row,9];
end;
```

Calculations must take place using memory ie 2D array and not using properties of string grid!
Answers are transferred to grid for display!

SOLUTION: EXERCISE 6

(CONTINUE)

D. Display the totals for games

```
procedure TQuestion6.btncalccolClick(Sender: TObject);
var rowtotal,r:integer;
row,col:integer;
begin
// adds totals for each game
  r:=stgdisplay.RowCount;
  stgdisplay.Cells[0,r]:='Errors per Game';
  for col := 1 to 8 do
    begin
        rowtotal:=0;
        for row := 1 to r-1 do
          rowtotal:=rowtotal+strtoInt(bafanarr[row,col]);
       bafanarr[row,col]:=inttostr(rowtotal);
    end;
  stgdisplay.RowCount:=stgdisplay.RowCount+1;
  row:=stgdisplay.RowCount-1;
  for col := 1 to 8 do
    stgdisplay.Cells[col,row]:= bafanarr[row,col];
end;
```

The extra row needed for the totals