

## FORM 5 PASCAL PROGRAMMING

### Unit 10: Sequential File Manipulation

March, 01

#### 10.1 WHEN SHOULD WE USE "FILES"?

- When a large quantities of data are needed to be processed, such as 1000 student records or 500 customer records, it is a good habit to store them in a separate file.

#### 10.2 TWO FILE VARIABLE TYPES IN PASCAL

##### *10.2.1 Typed File (分類檔案)*

- data are stored in binary form;
- every file has a type such as real, integer, char, string, records;
- examples of typed file declaration:

```
type
  StudentRec = record
    Name      : string;
    Age       : integer;
    Height    : real
  end;

var
  Numfile    : file of integer;
  CharFile   : file of char;
  ArrayFile  : file of array[1..10] of real;
  StudentFile : file of StudentRec;
```

##### *10.2.2 Text File (文字檔案)*

- data are stored in character (ASCII) form;
- each file is treated as a series of lines of text and each line is ended by a END-OF-LINE marker;
- examples of text file declaration:

```
var
  FileVariable : text;
```

## 10.3 TEXT FILE HANDLING STATEMENTS

### 10.3.1 *Declaring Files*

- Syntax

```
var
  <file variable list> : text;
```

- Example:

```
var
  InFile, StockFile, OrderFile : text;
```

### 10.3.2 *Using DOS Files*

- We should associate our file variable with the external file name before opening that file;
- Syntax

```
assign( <FileVariable>, <DOS file name> )
```

- Example:

```
assign(InFile, 'data.txt');
assign(StockFile, 'a:stock.dat');
assign(OrderFile, 'c:\data\order.dat');
```

### 10.3.3 *Opening Existing Files*

- Once the association is made, we can open the file for input;
- We can open a file in three different ways: RESET, REWRITE, and APPEND;

	reset	rewrite	append
Meaning	opens an existing file for input (reading)	creates and opens a file for output (writing)	opens an existing text file for output, starting at the end of the file
Syntax	reset(FileVariable);	rewrite(FileVariable);	append(FileVariable);
Example	reset(InFile);	rewrite(NewFile);	append(InFile);

### 10.3.4 Writing Data To Files

- Instead of sending output to the screen, we can output the data items to the file we specifies;
- Syntax

```
writeln (<File variable>, Variable);
```

- Example:

```
writeln (StockFile, MaxValue);;
```

- the current value of the variable MaxVariable will then be outputted to the DOS file associated with StockFile.

### 10.3.5 Reading Data From Files

- Instead of reading data from the keyboard, we can input data from the file we specifies;
- Syntax

```
readln (<File variable>, Variable);
```

- Example:

```
readln (StockFile, Today.HiValue);  
readln (OrderFile, Quantity[i]);
```

### 10.3.6 Closing Files

- Syntax

```
close ( <File variable> );
```

- Example:

```
close (Infile);  
close (StockFile);  
close (OrderFile);
```

### 10.3.7 Detecting End-Of-File and End-Of-Line Signal

- the end-of-file function eof ( ) is used to detect if the end of an opened file has been reached. If so, the Boolean value is set to TRUE;

- similarly, the end-of-line function `eoln` is used to detect if the end of current line of an opened file has been reached. If so, the Boolean value is set to `TRUE`;
- Syntax

```
eof(<FileVariable>)
eoln(<FileVariable>)
```

## 10.4 FILE OPERATION

### 10.4.1 Reading data from a text file

- this program retrieve the marks of a class stored in a sequential text file and display them on the screen:

```
(*****)
(*                                     *)
(* Retrieve Data From A Sequential File *)
(*                                     *)
(*****)

program ReadFile;

const MAX_LEN = 50;
var
  AdmNo, Mark : array[1..MAX_LEN] of integer;
                (* Large enough for any class *)
  Name       : array[1..MAX_LEN] of string[30];
  InFile     : text;
  I          : integer;
  NoOfStudent : integer;
  FileName   : string; (* Name of file in DOS *)
  Class     : string;

begin

  write( 'Enter name of class (e.g. 1A) : ' );
  readln( Class );
  FileName := 'F' + Class + '.DAT';
  (* DAT is the default extension. *)

  (* Prepare file to be read *)
  assign( InFile, FileName );
  reset( InFile );

  (* Get data about students. *)
  I := 0;
  while not eof( InFile ) do
  begin
    I := I + 1;
    readln( InFile, AdmNo[I] );
    readln( InFile, Name[I] );
    readln( Infile, Mark[I] )
  end;

  NoOfStudent := I;

  (* Close file *)
  close( InFile );

  (* Print data on screen. *)
  writeln( 'Admission No', 'Name':20, 'Mark':10 );
  writeln( '*****', '*****':20, '*****':10 );
  for I := 1 to NoOfStudent do
    writeln( AdmNo[I]:12, Name[I]:20, Mark[I]:10 )

end.
```

### 10.4.2 Creating a new file and entering data

- this program reads a list of names and writes it to a disk file:

```
(*****
*)
*)      Writing data to a Data file      *)
*)
*)
*****)
program CreateDataFile;
var
    OutFile : text;
    Name : string[20];
begin

    assign( OutFile, 'Name.txt' );
    rewrite( OutFile );

    write( 'Please enter the first name: ' );
    readln( Name );

    while Name <> 'Nomore' do
        begin
            writeln( OutFile, Name );
            write( 'Please enter the next name: ' );
            readln( Name );
        end;

    close( OutFile );

end.
```

### 10.4.3 Updating a sequential file

- Updating a file includes adding records, removing records, or changing existing records;
- The following illustrates a menu-driven program which updates the mark of a class:

```
(*****
*)
*)      Update Sequential Text File      *)
*)
*)
*****)

program UpdateFile;
const
    DELETED = -1; (* Deletion Indicator *)
    MAX_LEN = 100;
var
    AdmNo, Mark      : array[1..MAX_LEN] of integer;
                        (* Large enough for any class *)
    Name              : array[1..MAX_LEN] of string[30];
    ClassFile         : text;
    NoOfStudent       : integer;
    Choice            : integer; (* For menu selection *)
    Class             : string;

procedure SelectClass;
begin
    write( 'Enter name of class (e.g. 1A) : ' );
    readln( Class );
end; (* of SelectClass *)

procedure ReadData; (* Read data from class file *)
var I : integer;
begin
    (* Prepare file to be read *)
    assign( ClassFile, 'F' + Class + '.DAT' );
    reset( ClassFile );
    (* Get data about students. *)
```

```

I := 0;
while not eof( ClassFile ) do
  begin
    I := I + 1;
    readln( ClassFile, AdmNo[I] );
    readln( ClassFile, Name[I] );
    readln( ClassFile, Mark[I] )
  end;
NoOfStudent := I;
(* Close file *)
close( ClassFile )
end; (* of ReadData *)

procedure Add;
var
  A, I : integer;
  Ans : char;
begin
  writeln( 'Add records' );
  writeln( '*****' );
  repeat
    writeln;
    write( 'Enter admission number : ' );
    readln( A );
    I := 0;
    repeat
      I := I + 1
    until ( I = NoOfStudent ) or ( AdmNo[I] = A );
    if ( AdmNo[I] = A ) then
      writeln( 'Student already in the list found!' )
    else
      begin
        NoOfStudent := NoOfStudent + 1;
        AdmNo[NoOfStudent] := A;
        write( 'Enter name of student : ' );
        readln( Name[NoOfStudent] );
        write( 'Enter student's mark : ' );
        readln( Mark[NoOfStudent] )
      end;
    writeln;
    write( 'Any more (Y/N) ? ' );
    readln( Ans );
  until ( Ans = 'N' ) or ( Ans = 'n' )
end; (* of Add *)

procedure Delete;
var
  A, I : integer;
  Ans : char;
begin
  writeln( 'Delete records' );
  writeln( '*****' );
  repeat
    writeln;
    write( 'Admission no. of student to be deleted : ' );
    readln( A );
    I := 0;
    repeat
      I := I + 1
    until ( I = NoOfStudent ) or ( AdmNo[I] = A );
    if ( AdmNo[I] <> A ) or ( A = Deleted ) then
      writeln( 'Student not found!' )
    else
      AdmNo[I] := DELETED;
    writeln;
    write( 'Any more (Y/N) ? ' );
    readln( Ans );
  until ( Ans = 'N' ) or ( Ans = 'n' )
end; (* of Delete *)

procedure Change;
var
  A, I : integer;
  Ans : char;
begin
  writeln( 'Change records' );

```

```

writeln( '*****' );
repeat
  writeln;
  write( 'Admission no. of student to be updated : ' );
  readln( A );
  I := 0;
  repeat
    I := I + 1
  until ( I = NoOfStudent ) or ( AdmNo[I] = A );
  if ( AdmNo[I] <> A ) then
    writeln( 'Student not found!' )
  else
    begin
      write( 'Adm. no. = ', AdmNo[I],
            '      changed to : ' );
      readln( AdmNo[I] );
      write( 'Name      = ', Name[I],
            '      changed to : ' );
      readln( Name[I] );
      write( 'Mark      = ', Mark[I],
            '      changed to : ' );
      readln( Mark[I] );
    end;
  writeln;
  write( 'Any more (Y/N) ? ' );
  readln( Ans );
until ( Ans = 'N' ) or ( Ans = 'n' );
end; (* of Change *)

procedure Update;
var
  OldFile : text;
  I : integer;
begin
  assign( ClassFile, 'F' + Class + '.NEW' );
  rewrite( ClassFile );
  for I := 1 to NoOfStudent do
    if AdmNo[I] <> Deleted then
      begin
        writeln( ClassFile, AdmNo[I] );
        writeln( ClassFile, Name[I] );
        writeln( ClassFile, Mark[I] );
      end;
  close( ClassFile );
  writeln( 'File of F.', Class, ' updated.' );
end; (* of Update *)

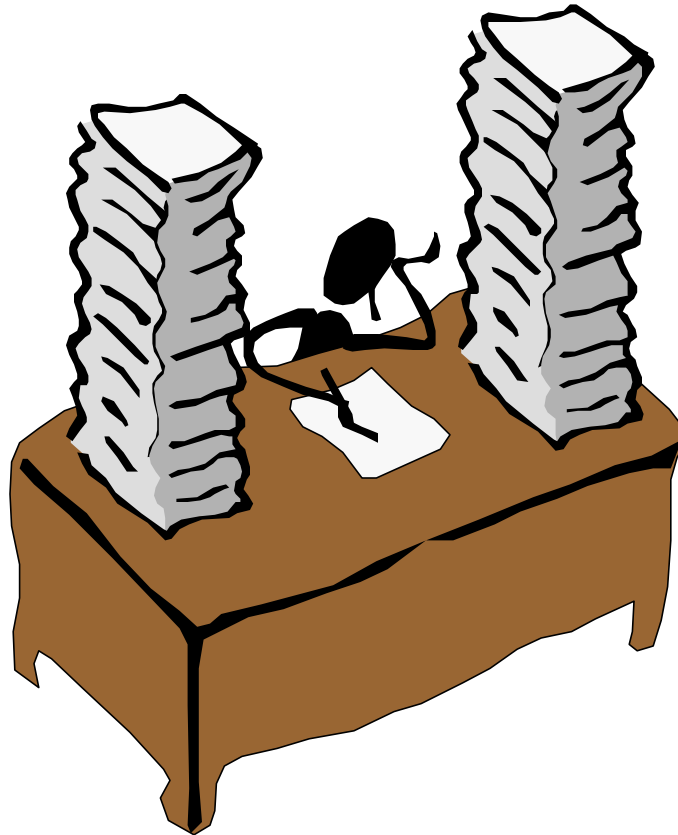
begin (* Main program *)
  SelectClass;
  ReadData;
  (** Main loop and Menu **)
  repeat
    writeln( '1. Add records' );
    writeln( '2. Delete records' );
    writeln( '3. Change records' );
    writeln( '4. Quit' );
    writeln;
    write( 'What do you want (1-4) ? ' );
    readln( Choice );
    case Choice of
      1 : Add;
      2 : Delete;
      3 : Change
    end
  until Choice = 4;
  Update (* Update file *)
end. (* of main program *)

```

## 10.5 BONUS: HOW TO SEND OUTPUT TO PRINTER

- We can send the output to a printer by adding a statement `uses printer` after the program heading;
- Whatever data items we want to output to the printer instead of screen, add `lst` inside the `writeln` bracket;
- Example:

```
program AddTwoNumbers;  
(* output to printer *)  
uses printer;  
var  
  Num1, Num2, Sum : integer;  
begin  
  Num1 := 10;  
  Num2 := 15;  
  Sum := Num1 + Num2;  
  writeln( lst, Sum );  
end.
```



end of unit 10