Introduction To Files In Pascal

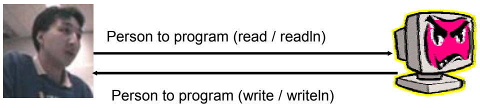
In this section of notes you will learn how to read from and write to files in Pascal.

Why Bother With Files?

- Too much information to input all at once
- The information must be persistent (RAM is volatile)
- Etc.

What You Know About Input And Output

Comes from the user or is displayed to the user



Person to program (read / readln)

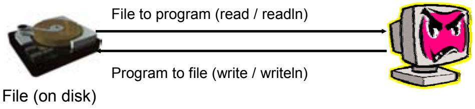
Person to program (write / writeln)

What You Need In Order To Read Information From A File

1. Declare a file variable
2. Open the file
3. A command to read the information

What You Will Learn: Input And Output Using Files

Information is retrieved from and written out to a file (typically on disk)



File to program (read / readln)

Program to file (write / writeln)

File (on disk)

1. Declaring File Variables

Allows the program access to a file

Format:

name of file variable : text;

Example:

letterGrades : text;

2. Opening Files

Prepares the file for reading:

A. Links the file variable with the physical file (references to the file variable are references to the physical file)

B. Positions the file pointer

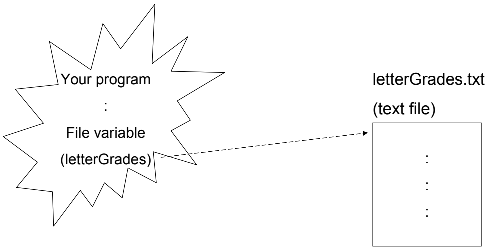
Format:

reset (*name of file variable, location and name of file*);

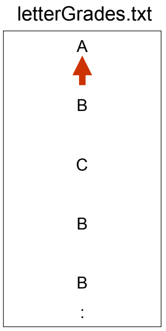
Example:

```
reset (letterGrades, ‘letterGrades.txt’);
```

A. Linking The File Variable With The Physical File



B. Positioning The File Pointer



3. Reading Information From Files

Performed with read or readln

Format:

read (*name of file variable*, variable to store the information);

readln (*name of file variable*, variable to store the information);

Example:

```
readln(letterGrades, letter);
```

3. Reading Information From Files (2)

Typically reading is done within the body of a loop

Format:

```
while NOT EOF (name of file variable) do
begin
    read (name of file variable, variable to store the information);
        OR
    readln (name of file variable, variable to store the information);
end; (* Done reading from input file *)
```

Example:

```
while NOT EOF (letterGrades) do
begin
    readln(letterGrades, letter);
    writeln(letter);
end; (* Loop to read letter grades file *)
```

An Alternative Approach To Reading Files

- Employ a sentinel in the file
- Keep reading from the file until the sentinel value is encountered

Example:

```
var inputFile : text;
var num       : integer;
   :            :
readln (inputFile, num);
while NOT (num = -1) do
begin
    writeln(num);
    readln(inputFile, num);
end; (* Done reading input file *)
```

Reading From Files: Putting It All Together

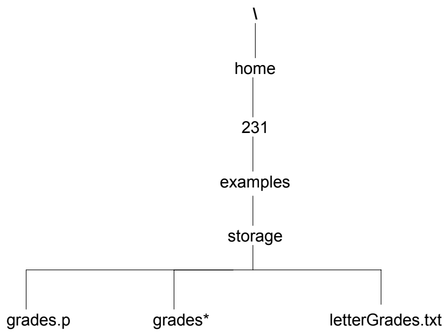
A complete version of this program can be found in Unix under /home/231/examples/storage/grades.p:

```
program grades (output);
begin
   var letterGrades : text;
   var letter           : char;

   (* Open file for reading, confirm file with user. *)
   reset(letterGrades, 'letterGrades.txt');
   writeln('Opening file "letterGrades" for reading.');

   while NOT EOF (letterGrades) do
   begin
        readln(letterGrades, letter);
        writeln(letter);
   end; (* Loop to read letter grades file *)
```

Reading From Files: Putting It All Together (2)

```
   close(letterGrades);
   writeln('Completed reading of file "letterGrades"');
end.
```

View Of Files In Unix



What You Need To Write Information To A File

1. Declare a file variable
2. Open the file
3. A command to write the information

1. Declaring An Output File Variable

- No difference in the declaration of a file variable when writing to a file from the case of reading from a file.

Format:

name of file variable: text;

Example:

```
letterGrades : text;
gradePoints  : text;
```

2. Opening The File

Two methods:

1) Rewriting – erases the old contents of the file (rewrites over what was already there).
2) Appending – retain the old contents of the file (appends the new information at the end).

Format (rewriting / appending):

rewrite (*name of file variable, location and name of physical file*);
append (*name of file variable, location and name of physical file*);

Example (rewriting / appending):

```
rewrite(gradePoints, ‘gradePoints.txt’);
append(gradePoints, ‘gradePoints.txt’);
```

3. Writing To A File

Format:

write (*name of file variables*, variable(s) and/or strings to write);

writeln (*name of file variables*, variable(s) and/or strings to write);

Example:

writeln(gradePoints, gpa);

Writing To A File: Putting It All Together

A complete version of this program can be found in Unix under:
/home/231/examples/storage/grades2.p

```
program grades (output);
begin
  var letterGrades, gradePoints : text;
  var letter                    : char;
  var gpa                       : integer;

  reset(letterGrades, 'letterGrades.txt');
  rewrite(gradePoints, 'gradePoints.txt');

  writeln('Opening file "letterGrades" for reading.');
  writeln('Opening file "gradePoints" for writing.');
```

Writing To A File: Putting It All Together (2)

```
  while NOT EOF (letterGrades) do
  begin
    readln(letterGrades, letter);
    case (letter) of
      'A'    :
      gpa := 4;

      'B'    :
      gpa := 3;

      'C'    :
      gpa := 2;

      'D'    :
      gpa := 1;

      'F'    : gpa := 0;

      else gpa := -1;
    end; (* case *)
```

Writing To A File: Putting It All Together (3)

```
      writeln(gradePoints, gpa);
  end; (* Loop to read letter grades file *)

  writeln('Finished reading and writing to files.');
  close(letterGrades);
  close(gradePoints);
end.
```

Details Of Write And Writeln For Files: Intuitive View

```
Program statement          Effect on file
rewrite(data,’data.txt’);                    (Open file
                                             "data.txt" and
                                             position file
                           ^                 pointer at
                                             start)

write (data, 'x');         x
                            ^

write(data, 'y');          xy
                             ^
write(data, 'z');          xyz
                              ^

writeln(data);             xyz
                           _
                           ^

write(data,'a');           xyz
                           a
                            ^
```

Details Of Write And Writeln For Files: Actual View

```
Program statement    Effect on file
rewrite(data,’data.txt’);
  (Open file
  "data.txt" and
  position file
                     ^              pointer at
  start)

write (data, 'x');   x
                      ^

write(data, 'y');    xy
                       ^
write(data, 'z');    xyz
                        ^
```

Details Of Read And Readln For Files: Intuitive View[1]

```
Program statement          Effect on file      Effect in program
reset (data, ‘data.txt’);  xyz                 (Open file
                           ^                   "data.txt" and
                                               position file
                                               pointer at
                                               start)
                           a

read(data, ch);            xyz                 Value of ch: 'x'
                            ^
                           a

readln(data, ch);          xyz                 Value of ch: ‘Y’

                           a
                           ^

read(data, ch);            xyz                 Value of ch: 'a'


                           a
                            ^
```

[1] Assume that the code on the previous slide has created the file called "data.txt"

Details Of Read And Readln For Files: Actual View[1]

```
Program statement          Effect on file      Effect in
                                               program
reset (data, ‘data.txt’);  xyz<EOL>a           (Open file
                           ^                   "data.txt" and
                                               position file
                                               pointer at start)

read(data, ch);            xyz<EOL>a           Value of ch: 'x'
                            ^

readln(data, ch);          xyz<EOL>a           Value of ch: 'y’
                                   ^

read(data, ch);            xyz<EOL>a           Value of ch:'a'
                                    ^

read(data,ch);             xyz<EOL>a
                                    ^
```

[1] Assume that the code on the previous slide has created the file called "data.txt"

Details Of Read And Readln For Files: Actual View[1]

```
Program statement          Effect on file      Effect in
                                               program
reset (data, ‘data.txt’);  xyz<EOL>a           (Open file
                           ^                   "data.txt" and
                                               position file
                                               pointer at start)

read(data, ch);            xyz<EOL>a           Value of ch: 'x'
                            ^

readln(data, ch);          xyz<EOL>a           Value of ch: 'y’
                                   ^

read(data, ch);            xyz<EOL>a           Value of ch:'a'
                                    ^

read(data,ch);             xyz<EOL>a           Error: read past
                                    ^          end of file
```

[1] Assume that the code on the previous slide has created the file called "data.txt"

Passing File Variables As Parameters

Must be passed as variable parameters *only*.

Format:

procedure *nameProcedure* (var *nameFile* : text);

Example:

```
procedure fileInputOuput (var letterGrades : text;
                          var gradePoints  : text);
```

You Should Now Know

- How to declare a file variable
- How to open a file for reading
- How to open a file a file for writing (rewrite and append mode)
- How to read (read/readln) from and write (write/writeln) to a file
- The details of information is read from and written to a file
- How to close a file and why it is good practice to do this explicitly
- How to pass file variables as parameters