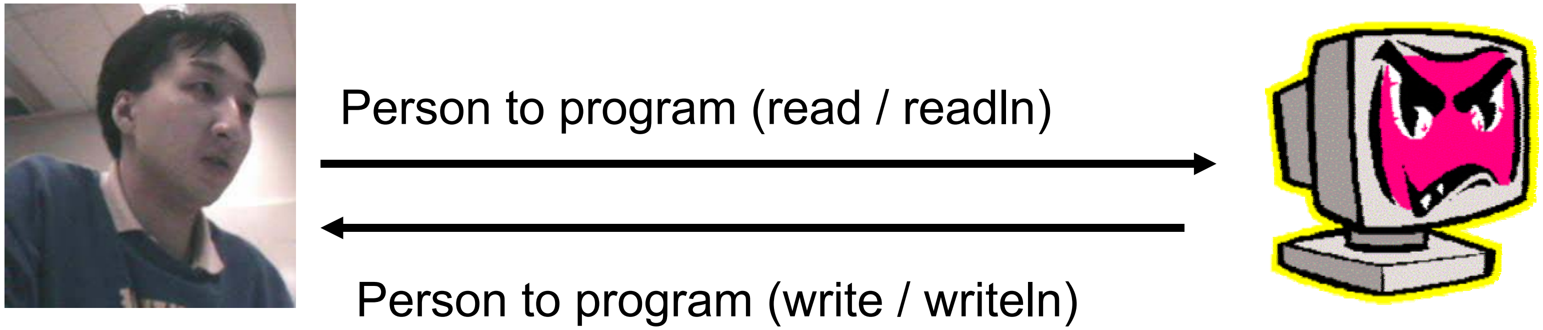
Introduction To Files In Pascal

You will learn how to read from and write to text files in Pascal.

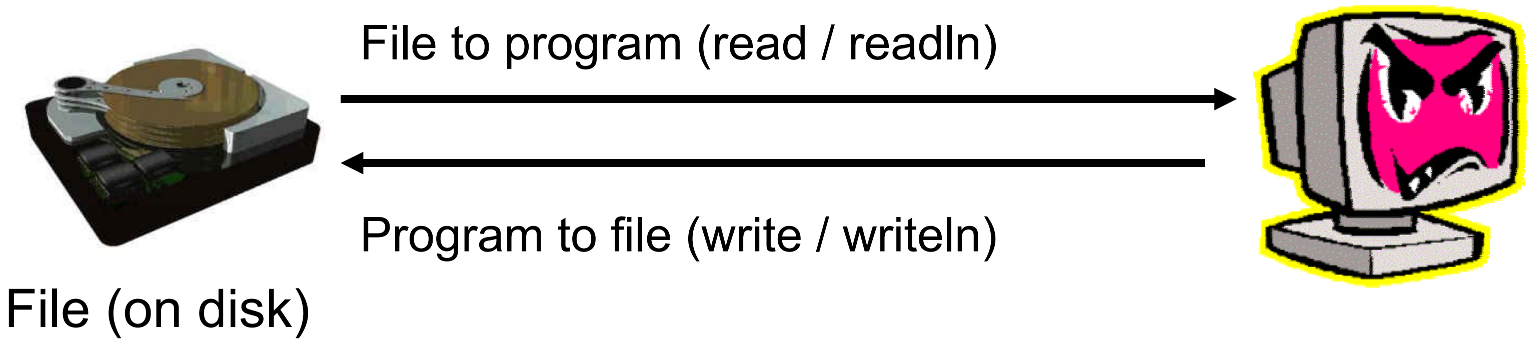
What You Know About Input And Output

Comes from the user or is displayed to the user


Person to program (read / readln)
Person to program (write / writeln)

What You 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)

Why Bother With Files?

Too much information to input all at once

The information must be persistent

Etc.

What You Need To Read Information From A File

Indicate that you are going to read from a file

Declare a file variable

Open the file

A command to read the information

Indicating What File You Are Reading From

Syntax:

program *name* (*name of input file*[1]);

Example:

program grades (output, letterGrades);

1 The name of input file must correspond to an actual file in the same directory that the executable program resides

Declaring File Variables

A variable that is associated with a file on disk

Syntax:

name of file[2] : text;

Example:

letterGrades : text;

2 The name of file variable must correspond to an actual file in the same directory that the executable program resides

Opening Files

Purpose:

Prepares the file for reading (positions the file pointer)

Syntax:

reset (*name of input file*[3]);

Example:

```
reset(letterGrades);
```

3 The name of file being opened must correspond to an actual file in the same directory that the executable program resides

Reading Information From Files

Performed with read or readln

Syntax:

read (*name of input file*[4], variable(s));

readln (*name of input file*[4], variable(s));

Example:

readln(letterGrades, letter);

4 The name of file being read from must correspond to an actual file in the same directory that the executable program resides

Reading Information From Files (2)

Typically reading is done within the body of a loop

Syntax:

```
while NOT EOF (name of input file⁵) do
begin
    read (name of input file⁵, variable(s));
    readln (name of input file⁵, variable(s));
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 *)
```

5 The name of the input file must correspond to an actual file in the same directory that the executable program resides

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;
    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

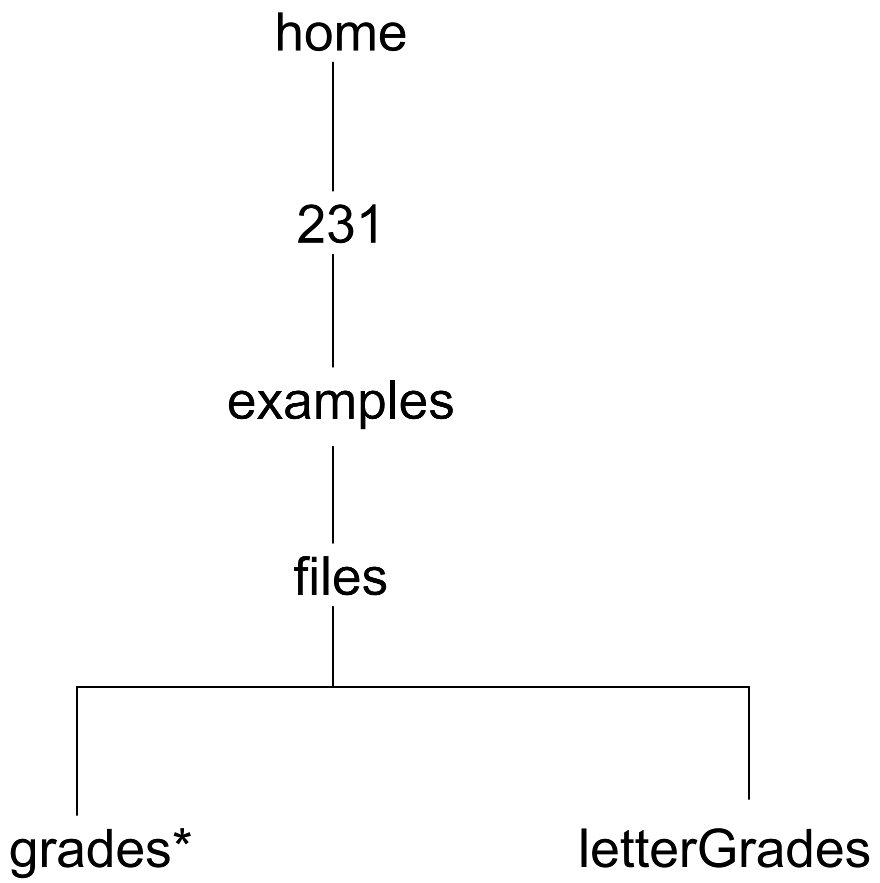
A complete version of this program can be found in Unix under /home/231/examples/files/grades.p

```
program grades (output, letterGrades);
var
  letterGrades : text;
  letter       : char;
begin
   reset(letterGrades);
   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.  (* End of program *)
```

View From Unix



What You Need To Write Information To A File

Indicate that you are going to write to a file

Declare a file variable

Open the file

A command to write the information

Indicating That You Are Writing To A File

Syntax:

program *name* (*name of output file*);

Example:

program grades (output, letterGrades, gradePoints);

Declaring An Output File Variable

No difference between declarations when reading from a file or writing to a file.

Syntax:

```
name of file : text;
```

Example:

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

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).

Syntax (rewriting / appending):

rewrite (*name of file*);   append (*name of file*);

Example (rewriting / appending):

```
rewrite(gradePoints);        append(gradePoints);
```

Writing To A File

Syntax:

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

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

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/files/grades2.p

```
program grades (output, letterGrades, gradePoints);
var
   letterGrades : text;
   gradePoints  : text;
   letter       : char;
   gpa          : integer;
begin
   reset(letterGrades);
   rewrite(gradePoints);
   writeln('Opening file "letterGrades" for reading.');
   writeln('Opening file "gradePoints" for writing.');
while NOT EOF (letterGrades) do
begin
```

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

```
      readln(letterGrades, letter);
      case (letter) of
        'A'      : gpa := 4;
        'B'      : gpa := 3;
        'C'      : gpa := 2;
        'D'      : gpa := 1;
        'F'      : gpa := 0;
        otherwise gpa := -1;
       end; (* case *)
       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);                                 (Open file "data" 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);                                  (Open file "data" and position
                           ^                       file pointer at start)


write (data, 'x');         x
                            ^


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


writeln(data);             xyz<EOL>
                                   ^


write(data,'a');           xyz<EOL>a
                                    ^
```

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

```
Program statement     Effect on file     Effect in program
reset (data);         xyz                (Open file "data" 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"

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

```
Program statement        Effect on file      Effect in program
reset (data);            xyz<EOL>a           (Open file "data" 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"

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

```
Program statement        Effect on file     Effect in program
reset (data);            xyz<EOL>a          (Open file "data" 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 – reading past end of file
                                  ^
```

1 Assume that the code on the previous slide has created the file called "data"

Passing File Variables As Parameters

Must be passed as variable parameters *only*.

Syntax:

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

Example:

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

Summary

You should now know:

- How to read information from a text file with Pascal
- How to write information to a text file with Pascal
- The difference between write and writeln when writing to text files using Pascal
- The difference between read and readln when reading from text files using Pascal