Files

Files

- To open a file for reading / writing, we need to know its filename and its file handle

- **Cannot have a file handle without a filename, but can have a filename without a file handle**

Parts of a filename

- Filename is the actual path and name of a file

- Filename may look like:

  C:\Documents and Settings\My Pictures\Sunset.jpg

- When referring to a file handle, filename includes the directory or path to the actual file

File handles (1/2)

- It is a direct link from a computer program to the file on the disk
- It is used to read and write to a file
- It is created on opening a file
- **Does not begin with $**
- Could be any combination of alphanumeric characters, upper and lower case

- Typically used **UPPER CASE**

File handles (2/2)

- Three default file handles
  - STDIN – set to keyboard (default).
  - STDOUT – set to monitor (default).
  - STDERR – set to monitor (default).

The open function

- Opening files

- Syntax

  open (FILE HANDLE, “filename”);

File opening modes (1/2)

- Read mode
  - Read from a file

    open (INFILE, “<filename”);
- Write mode
  - Write to a new file, destroy existing file

    open (OUTFILE, “>filename”);
- Append mode
  - Write to a the end of an existing file

    open (APPFILE, “>>filename”);

File opening modes (2/2)

- Read and write mode
  - Read and write from existing file

    open (RW, “+<filename”);
- Write to a program
  - Send data to a program or command

    open (PIPEOUT, “| filename”);
- Read from program
  - Receive data from a program or command

    open (PIPEIN “filename |”);

The close function

- Closing a file handle

- Syntax

  close (FILEHANDLE);

Reading file into array

```
open (INFILE, “>file1.txt”) || die “$! Cannot open file”;
@file = <INFILE>;

print “@file\n\n”;
close (INFILE);
```

The unlink function

- Deleting a file
- Syntax

  unlink filename;

  unlink filelist;
- Example

  unlink file1.txt, file2.txt;

  unlink <*.txt>;

  – The last unlink function here will delete all files that have a “.txt” extension

The rename function

- Renaming a file

- Syntax

  rename oldname, newname;

- Example

  rename file1.txt, newfile1.txt;

File test operators (1/2)

- Syntax

  operator $filename;

  operator FILEHANDLE;

- If test is true, operator returns 1; else an empty string

- File existence and size tests
  - -e → file exists
  - -s → non-zero size file
  - -z → zero size file

File test operators (2/2)

- File type tests
  - -f → plain file
  - -d → directory
  - -l → symbolic link (shortcut) – unix filesystem only
  - -T → text file
  - -B → binary file

The print function

- Different forms

print FILEHANDLE list;	print to FILEHANDLE the data in list
print FILEHANDLE;	print contents of $_ to FILEHANDLE
print list;	print to default selected filehandle the data in list
print;	print to STDOUT the contents of $_

String functions (1/5)

- **chomp**
  - Removes only a newline character from the end of the string
  - Usage:- lvalue = chomp($scalar);
- **chop**
  - Removes the last character of each element of the input
  - Usage:- lvalue = chop($scalar);
- **chr**
  - Translates a number into a character
  - Usage:- lvalue = chr(number);

String functions (2/5)

- **ord**
  - Converts ASCII characters into numeric values
  - Usage:- **lvalue = ord(exp);**
- **lc**
  - Changes all characters in expresion to lowercase
  - Usage:- **lvalue = lc(exp);**
- **uc**
  - Changes all characters in expresion to uppercase
  - Usage:- **lvalue = uc(exp);**

String functions (3/5)

- **lcfirst**
  - Changes only the first character in expression to lowercase
  - Usage:- **lvalue = lcfirst(exp);**

- **ucfirst**
  - Changes only the first character in expression to uppercase
  - Usage:- **lvalue = ucfirst(exp);**

String functions (4/5)

- **length**
  - Returns number of characters in a string
  - Usage:- **lvalue = length(exp);**

- **substr**
  - Returns or modifies a substring
  - Usage:- **lvalue = substr(exp, startPos, length);**

String functions (5/5)

- **s/ (substitute)**
  - Substitute one string for another string
  - Usage:- **$searchstring =~ s/oldpattern/newpattern/;**

- **tr (translate)**
  - Exchanges each occurrence of characters in the searchstring with its matching characters in the replacement string
  - Usage:- **$input =~ tr/searchstring/replacestring/;**

Exercises

- Write a program to test the type of file.
- Write a program to display the contents of a file.
- Write a program to rename a file.
- Write a program to copy contents of one file to another.
- Write a program to delete lines in a file that matches a pattern given by user.

- Write a program to read in a list of filenames and then display which of the files are readable, writable, and/or executable, and which ones don't exist.

- Write a program that prompts for an input filename, an output filename, a search pattern, and a replacement string from command line, and replaces all occurrences of the search pattern with the replacement string while copying the input file to the output file.

- Write a program to read in a filename from STDIN, then open that file and display its contents with each line preceded by the filename and a colon. [For example, if fred was read in, and the file *fred* consisted of the three lines aaa, bbb, and ccc, you would see fred: aaa, fred: bbb, and fred: ccc]