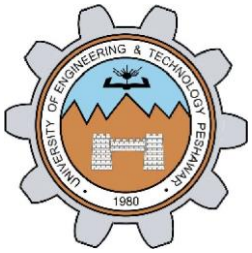
LAB #01
Introduction to the UNIX



Spring 2023

CSE-204L Operating Systems Lab

Submitted by: MUHAMMAD SADEEQ

Registration No.: 21PWCSE2028

Section: C

"On my honor, as a student of the University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work"

Submitted to:

Engr. Madiha Sher

(27 Feb 2023)

Department of Computer systems engineering
University of Engineering and Technology,
Peshawar

Introduction to the UNIX

Part 1: Directory Structure and Manipulation

UNIX has a hierarchical file system. That means that all directories are based upon a root directory (in UNIX it is the / directory). This is illustrated in Figure I.

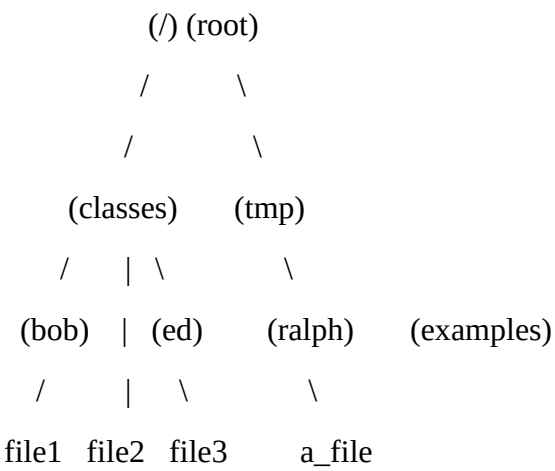


Figure I

A pathname identifies a particular file or directory uniquely. An absolute pathname starts at the root and names each directory along the way to the destination directory or file. An example is (***/tmp/examples/a_file***). This specifies a file called ***a_file*** which exists in the ***examples*** subdirectory. A relative pathname tells how to reach the destination from the current directory. If you were in the ***/tmp*** directory, the path to ***a_file*** would be ***examples/a_file***.

Every user has a home directory in UNIX. It is the directory you start off in when you first log in. In order to specify that directory when referencing it as a file a user can use the ~ key. ***~/files/file*** is a file located in the files subdirectory located in the user's home directory. In order to reference another user's home directory use ***~username*** (an example is ***~bob/file1***).

Other special reference variables are '**.**' and '**..**' The '**.**' reference refers to the current directory so if you are in the ***/tmp*** directory and you want to reference the ***example*** subdirectory you could use ***./examples***. The '**..**' directory refers to the directory to which the current directory is a subdirectory of. If you are in the ***/classes/bob*** directory then **..** refers to the ***/classes*** directory.

Commands to use on directories and files:

cd ***arg***

The change directory command. When taken with no arguments it means to change directory to the users home directory. To change to the ***examples*** directory you would use **cd** ***/tmp/examples***. This accepts all wildcards and special commands like '**.**' and '**..**'

mkdir ***name***
Creates a directory in the spot specified by ***name***. If you wish to make a directory called ***my_dir*** in your current directory then the command would be **mkdir** ***my_dir***.

rmdir ***name***
Removes a directory in the spot specified by name.

NOTE:

Directory and file permissions are very important to UNIX security because of the multiple users on the file system. Each file and directory has an owner, belongs to a user group, and contains permission bits. In order to see the permission bits of all files in a directory a user can type **ls -l** or to get a specific file a user can type **ls -l file**. An example is

% ls -l ***/classes/bob/file1***
Permission bits are displayed in the form: **-rwxr-xr-- 1 bob projx 538 May 24 10:12 /classes/bob/file1**

The first field (in this case -) indicates that it is a file. If it were a directory, it would be indicated with a **d**. The next three fields are the permission bits for the owner, the next three are for the group and the third three are for the rest of the world. These three groups specify whether the user has read, write, or execute/search privileges. In this case the user has all privileges. The group **projx** can read and execute the file and the world can just read the file. The owner of a directory or file can change the permissions on a file by using the **chmod** command. All users should read how to change permissions by typing in **% man chmod**

This gives you the UNIX manual page which explains the **chmod** command. You can use the man command to get info on most unix and c commands. The syntax is **man** ***command***.

Questions:

(All questions refer to **Figure I** as the file structure)

1) You are the user bob and are in your home directory, the ***/classes/bob*** directory. Give 5 ways to reference the file ***file3***. You must use Absolute, Relative and the '**.**' and '**..**' reference methods at least once.

Ans)

1) `../ed/file3`
2) `../../classes/ed/file3`
3) `~/../../ed/file3`
4) `/classes/ed/file3`
5) `./../ed/file3`

2) If you are logged into **DCSE** and you want to go to your TA's home directory how would you do it without knowing anything about the DSCE departments file structure? (Your TA's username is ***torfanos***)

Ans) `cd /home/torfanos`

3) If you are user bob, how would you change the permission on ***file1*** to just execute for user and group and no privileges for the world?

Ans) `chmod 110 /classes/bob/file1`

Part 2: File Redirection and Pipes

Standard output is the data that is usually displayed on the screen. We may send standard output to a file by using the **command** > ***file*** syntax. We say that > redirects the standard output. **command** is a single command (or command group) and file is any valid file pathname. If you want to list what you have in your directory and save it to a file called ***dir_list*** you would enter **ls** > ***dir_list***. You can append data to a file using the >> command. To add the list of items in the root directory to your directory list you could use **ls** / >> ***my_dir***.

Input redirection is using a file as the input for a command. The syntax is **command** < ***file***. An example of this is if Bob wants to mail Ed a copy of ***file1***; he would enter **mail ed** < ***file1***.

Sequences of events often require a temporary file to hold intermediate results. For example if Bob entered the commands:

% ls > ***dir_contents***
% mail ed < ***dir_contents***
% rm ***dir_contents*** (deletes the file)

Then Ed would get a copy of Bob's directory. We can eliminate the need for a temporary file by using a pipe to connect the output of **ls** to the input of **mail**.

The pipe symbol is **|** in UNIX. The solution would be **ls | mail ed**. The general format of a pipe is:

command1 | command2 | command3 | ... | command*n*

Questions:

(All questions refer to **Figure I**)

1) The command **wc** counts words, characters or lines. The syntax is:
wc [options] [file(s)]

options:
-l counts # of lines in file
-c counts # of bytes in file
-w counts # of words in file

If ***file2*** is a list of users with more than one on each line, how would Ed (from his home directory) store the number of users in a file called ***num_users***?

Ans) `wc -w ../file2 > num_users`

2) The command **sort -d *file*** displays a file in dictionary order. The command **more** (**more *file***) displays the contents of a file one page at a time.

Show how Bob would display the sorted contents of his home directory one page at a time.

Ans)
```
ls > dir_list;
sort -d dir_list;
more dir_list;
```

Part 3: Wildcards and Special Characters

When describing a file a user may use wildcards to help name more than one file. This is called filename expansion and the *csh* (and *tcsh* which is running on the ee systems and cs systems) supports these characters:

?	Matches any single character
[list]	Matches any character in list.
[lower-upper]	Matches any character in the range between lower and upper (inclusive)
*****	Matches any pattern (including null)

Examples:

% ls *.c	lists all files which end in .c
% ls file*	list all files that start with file (includes file by itself)
% ls program.?	list all files that start with program. and have a one letter suffix.
% ls file[1-2]	list file1 and file2.

Special Characters that the shell interprets first:

\	Dereferences the following character (used to use things with * or ? in the name).

$	Variable Identifier
;	Ends a command

Questions:

These questions are based on a directory which has the following files in it:

file1 file2 file3 file4 afile file.c file.s file.so file.o farm.c farm.co

NOTE: All expansion solutions should be of the format **command *file(s)***. There should only be one argument following command and in order to get multiple files you must use wildcards.

1. How would you list files these files (***file2, file3, file4***)? (HINT use **ls**)

Ans) `ls file[2-4]`

2. How would you remove, using the **rm *file(s)*** command, ***file.c*** and ***file.s*** (note: keep ***file.o***)?

Ans) `rm file.[s,(s),c]`

3. How would you remove any file which contains the word ***file*** in it?

Ans) `rm *file*`

4. How would you list all files that begin with ***f*** and end with a 2 letter suffix?

Ans) `ls f*.??`

If you had a file which was called ***hard?*** , with ***?*** actually being a question mark. How would you reference that file?

Ans) `hard\?`