Tasks

In Ubuntu, login and launch the command prompt (aka Terminal). The keyboard shortcut is <CTRL><ALT><T>. Or, you can search for "Terminal" in “Software”.

Do the following tasks.

The "unix>" part of the line should not be typed in, and it will not look the same on your computer. Rather, it might look something like this: "username@computername:/current-directory$", i.e. a combination of your username, computer name, and the current directory you are in. Thus, it will be constantly changing as you work! To simplify this lab, this is written as "unix>"

1. List files: ls

When you first login, your current working directory is your home directory. Your home directory has the same name as your user-name, for example, *vivek*, and it is where your personal files and subdirectories are saved. To find out what is in your home directory, use the ls command. (ls is short for "list") There may be no files visible in your home directory, in which case, nothing will print but the command prompt again.

List contents of your current working directory:

```
unix> ls
```

The ls command does not show hidden files by default. Hidden files have file names that begin with a dot (.). To list all files in your home directory including hidden files, use the -a option with the ls command.

List all contents of your current working directory, including hidden files:

```
unix> ls -a
```

2. Make directory: mkdir

To make a directory called ecpe170 inside your current working directory, type

```
unix> mkdir ecpe170
```

To see the directory you have just created, type

```
unix> ls
```

3. Change to a different directory: cd

The command cd directory means change the current working directory to 'directory'. The current working directory may be thought of as the directory you are in, i.e. your current position in the filesystem. To change to the directory you have just made, type

```
unix>  cd ecpe170
```

To go up one level in the directory tree, use two dots:

```
unix>  cd ..
```

At this point you should be back in your home directory.

To jump back to your home directory (regardless of how deep in the file tree you happen to be), use either of the two following commands:

```
unix>  cd
```

or

```
unix>  cd ~
```

In Unix systems, the tilde (~) character represents your home directory.

4. Show the current directory: pwd

Pathnames enable you to work out where you are in relation to the whole file system. For example, to find out the absolute pathname of your current directory, use:

```
unix>  pwd
```

The full pathname will look something like /home/jshafer which means that jshafer (your home directory) is inside the directory home (the directory containing all user accounts).

5. **Copy file: cp**

Before doing these commands, enter the ecpe170 directory, and create two empty files using the touch command:

```
unix>  cd ecpe170
unix>  touch file1
```

To make a copy of file1 and give it the name file2 (in the current working directory), use:

```
unix>  cp file1 file2
```

Now you should have two files: file1 and file2.

Note that you can use relative or absolute paths to identify files. For example, to copy file1 from the current working directory and save it as the name file2 in one directory up from the current directory, use:

```
unix>  cp file1 ../file2
```

6. Move file: mv
To move (or rename) file1 into file2, use:

```
unix>  mv file1 file2
```

7. Delete file: rm
To delete ("remove") a file, use:

```
unix>  rm file2
```

Caution! The rm command is very powerful. Once you remove a file, recovering that file can be very difficult, if not impossible. The command line does not provide the safety of "are you sure?" questions like the GUI interface does. Further, no "unremove" or "undelete" command. If you're paranoid about deleting something accidentally, use this instead:

```
unix>  rm -i file2
```

8. Delete ("remove") directory: rmdir

```
unix>  rmdir directoryname
```

Note that the directory must be empty first!

9. Clear the terminal window: clear

```
unix>  clear
```

10. View a (text) file on screen: less

```
unix>  less /var/log/syslog
```

The command less writes the contents of syslog (located in the /var/log/ directory) onto the screen a page at a time.

- Press the [space-bar] if you want to see another page
- Press the [arrow keys up/down] if you want to scroll
- Press [q] if you want to quit reading.

11. Read documentation on a command: man

```
unix>  man command-name
```

For example, use man ls to view the manual for the ls utility. These are colloquially referred to as "man pages".

- Press [q] if you want to quit reading

12. Search document for a matching command: apropos

When you are not sure of the exact name of a command, use apropos to search the "man pages" for a command matching a keyword. For example, to search for programs those have something to do with "sort", use:

```
unix>  apropos sort
```

Note that this is the same as using the "-k" option with man:

```
unix>  man -k sort
```

13. Run a command as the administrative ("root") user: sudo

Sudo allows you (a regular user) to run a command as an Administrative ("root") user by typing your password. Everything that comes after sudo is the command to run as an Administrative user. Administrative access is required to install software and read/move/delete files not owned by your current user.

```
unix>  sudo <<command-to-run-as-administrator>>
```

For example, the apt-get utility that we used to install software packages must be run as the root user.

$ sudo apt update ---> To update software packages
$ sudo apt install build-essential ---> To install software packages
$ sudo apt-get install manpages-dev ---> To install software packages

14. D**ownload a file from the Internet: `wget`**

The wget utility can download files (of any kind) via HTTP:

```
unix>  wget http://www.somesite.com/somefile.dat
```

unix> wget https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.17.2.tar.xz

15. Edit a text file: `gedit`

Run gedit to create an untitled document:

```
unix>  gedit
```

16. Run gedit to create the document myfile.txt

```
unix>  gedit myfile.txt
```

Place the following text in myfile.txt and save it

Neo: What are you trying to tell me? That I can dodge bullets?
Morpheus: No, Neo. I'm trying to tell you that when you're ready, you won't have to.

17. Count the number of characters, words, or lines in a file: `wc`

```
unix>  wc myfile.txt
```

What do the first three numbers in the output of the `wc` utility mean?

18. Redirect input and output: < and >

Many programs take their input from "standard input" (i.e. the console keyboard) and display results to "standard output" (i.e. the console monitor). However, you can change this, so either the input, output, or both are files. This works even if the program was never written to expect a file!

Change the input to a program from *standard input* to a *file*: **<**

```
unix>  sort < myfile.txt
```

Change the output of a program from *standard output* to a *file*: **>**

```
unix>  man wc > docs_for_wc_program.txt
unix>  less docs_for_wc_program.txt
```

Change the output of a program from standard output to a file, appending to the end: **>>**
(You should have two copies of the docs in the file now!):

```
unix>  man wc >> docs_for_wc_program.txt
unix>  less docs_for_wc_program.txt
```

19. $ gcc --version ---> gcc is a compiler for C programming language. Using this command version of gcc id displayed.

$cat myfile1.txt ---> This command will display the content of myfile1.txt