

Please complete the following questions. If you need help, feel free to ask questions both from other students and from me. If you cannot finish during the class time then you should finish this lab at home. After you finish, feel free to clean up (*i.e.*, remove) any files or directories that were created, you won't need them in the future. But be careful not to remove any other important files you may have in your home directory.

1. Command review.

- (a) Make a new directory called cse399-lab.

Answer:

```
bash$ mkdir cse399-lab
```

- (b) Change your location to this directory.

Answer:

```
bash$ cd cse399-lab
```

- (c) Copy the directory /home1/m/mcorliss/teaching/cse399/lab1 and all of its subcontents to your current location, naming the directory cse399-lab1.

Answer:

```
bash$ cp -r /home1/m/mcorliss/teaching/cse399/lab1 cse399-lab1
```

- (d) Change your location to this directory.

Answer:

```
bash$ cd cse399-lab1
```

- (e) List only files ending in '.jpg' using the long format (hint: use 'man' to find out how to list files with the long format).

Answer:

```
bash$ ls -l *.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 cse399-1.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 cse399-10.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 cse399-2.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 cse399-3.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 cse399-4.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 cse399-5.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 cse399-6.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 cse399-7.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 cse399-8.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 cse399-9.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 unix-1.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 unix-10.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 unix-2.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 unix-3.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 unix-4.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 unix-5.jpg
```

```
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 unix-6.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 unix-7.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 unix-8.jpg
-rw-r-- 1 mcorliss 30675 0 Jan 19 10:44 unix-9.jpg
```

- (f) Print out all files that start with 'cse399' and end in '.txt'.

Answer:

```
bash$ cat cse399*.txt
dkfldsfdfklds
```

- (g) Remove files starting with 'cse399'.

Answer:

```
bash$ rm cse399*
```

- (h) Rename the file 'unix-9.jpg' to 'foo.jpg'.

Answer:

```
bash$ mv unix-9.jpg foo.jpg
```

- (i) Move the file 'foo.jpg' to tmp/

Answer:

```
bash$ mv foo.jpg tmp/
```

- (j) List the contents of the directory 'tmp/' without changing your location. Also, your command should list all files (including hidden files) except the implied files '.' and '..'.

Answer:

```
bash$ ls -A tmp/
.tmp.txt foo.jpg tmp-1.txt tmp-2.txt tmp-3.txt
```

2. New commands.

- (a) **echo.** Type 'echo "hello, world"' at the prompt. What does 'echo' do?

Answer:

echo prints a line of text to the screen.

- (b) **wc.** To find the size of a file, you can use 'wc' command. Type 'wc large-file.txt'. Notice that it returns three numbers. Use 'man' to find out what each number represents. Use 'man' again to also figure out how to just return the size in lines.

Answer:

```
bash$ wc large-file.txt
18144 16902 396288 large-file.txt
```

wc returns the size of the file in lines (18144), words (16902), and bytes (396288).

To return just the lines:

```
bash$ wc -l large-file.txt
18144 large-file.txt
```

- (c) **cmp.** To compare two files to see if the contents are the same you can use 'cmp'. Type 'cmp large-file.txt unix-1.txt'. Now copy 'large-file.txt' to 'large-file-2.txt'. Compare 'large-file.txt' and 'large-file-2.txt'.

Answer:

```
bash$ cmp large-file.txt unix-1.txt
large-file.txt unix-1.txt differ: char 1, line 1
```

```
bash$ cp large-file.txt large-file-2.txt
bash$ cmp large-file.txt large-file-2.txt
```

- (d) **touch.** Type 'ls new-file.txt'. Type 'touch new-file.txt'. Again type 'ls new-file.txt'. What did 'touch' do? 'touch' has a different effect on existing files than on new files. To see this, we'll touch 'large-file.txt'. But first list 'large-file.txt' using the long format. Now touch 'large-file.txt'. Use 'cmp' to compare 'large-file.txt' and 'large-file-2.txt'. Did the contents of 'large-file.txt' change? Using the long format, list 'large-file.txt'. What did 'touch' do to 'large-file.txt'?

Answer:

'touch new-file.txt' created a new empty file 'new-file.txt' since the file did not exist. 'touch large-file.txt' changed the modification time of the existing file large-file.txt to the current date. It did not change the contents of the file.

- (e) **more.** Type 'cat large-file.txt' at the prompt. Notice that you cannot view the beginning of the file since the shell will not scroll far enough back. The problem with cat is it doesn't allow you to pause as you view the file. To view a file one screenful at a time you can use the 'more' command. Type 'more large-file.txt'. This shows the first part of the file. To see the next screenful, type the space bar. When you get to the bottom of the file, 'more' will quit and return you to the prompt. You can also type 'q' to quit at any other point. Unfortunately, 'more' is pretty restricted, we can't arbitrarily scroll up and down while viewing the file. In the homework, you will use a command that gives you more control.

3. Miscellaneous.

- (a) **stdin.** Type 'cat' at the prompt. Notice that this does not bring you back to the prompt. Type some text and hit return. Repeat. 'cat' can concatenate files, but it can also concatenate input from stdin (*i.e.*, entered from the keyboard).
- (b) **<ctrl>-c.** To get back to the prompt, type '<ctrl>-c' (hold the control button down and at the same time hit the key 'c'). In Unix, pressing <ctrl>-c sends INT (interrupt) signal to the running process (in this case the 'cat' command), which causes the process to halt. This works with other commands as well. We'll see more uses of <ctrl>-c in the upcoming lectures.
- (c) **~.** Print your current location. Type 'cd ~' at the prompt. Print your current location again. Type 'ls ~' at the prompt. Make a directory called 'foo'. Type 'cd ~/foo' at the prompt. Print your current location again. What is '~' equal to?

Answer:

'~' is shorthand for your home directory. When entering Unix commands, it is often nice to think of paths in terms of your home directory. '~' allows you to do this.

- (d) **cd -.** Type 'cd ~' and then 'cd ~/foo'. Print your current location. Now type 'cd -'. Print your current location. Type 'cd -' again. Print your current location. What does 'cd -' do?

Answer:

'cd -' changes your location to the previous location. Using 'cd -', you can jump back and forth between two directories.

- (e) **tab completion.** Type 'cd ~/foo'. Make a new directory called 'goo'. Type 'cd goo'. Print your current working directory. Then type 'cd ', but don't hit enter. Begin typing the absolute path (printed from the last command). After typing each character of the path, hit the tab key repeatedly. What happens? What happens after you have typed 'foo/' of the absolute path and you hit the tab key?

Answer:

The tab key will autocomplete the path. If the next directory or file is unambiguous it will fill it in automatically. If not, it will list all the possible matching files and directories. tab completion can be used with most Unix commands, for instance, any that require a path. tab completion will also work with man. Try typing 'man rmd' at the prompt and then hit tab repeatedly. You get a list of possible matching commands. Now type an 'i' (*i.e.*, 'man rmdi') and hit tab repeatedly. This will expand to 'man rmdir'.