

Lab 01: Part 1, In Class

Using the Unix Command Line

Instructions

Exercises 1.1 – 1.8 are to be completed in class before you leave. You need to create a directory in your home directory named “lab01” (you must use the exact same name, and remember that Unix is case sensitive, so “Lab01” is not the same name). Inside of the directory “lab01” you will create and edit a file called “lab01-part1.txt”. You should edit the file and type in your answers to the questions 1.1 – 1.8 before the end of class, and ask the instructor to confirm you are done with this part before leaving.

Use this lab time not only to find the answers to the questions, but to practice your skills in using the command line and editor to create directories, files and run commands. The first portion of the lab is to be completed in class so that you may ask the instructor for help and suggestions for any problems you may be experiencing learning the bash shell and command line commands. You may use any of the resources we have discussed to find the answers to the exercises, as well as the textbook and online source materials. It is of course suggested that you try out and test your answer where possible to verify it correctly answers the question.

Exercise 1.1

- The instructor wants you to use a hitherto unknown *command* for manipulating files. How would you get help on this *command*?

Exercise 1.2

- Suppose ls shows you this:

```
Makefile biography.txt data enrolment.txt programs thesis
```

- What argument(s) will make it print the names in reverse, like this:
thesis programs enrolment.txt data biography.txt Makefile

Exercise 1.3

- What does the command `cd ~` do? What about `cd ~hpotter`?

Exercise 1.4

- What command will show you the first 10 lines of a file? The first 25? The last 12?

Exercise 1.5

- `diff` finds and displays the differences between two text files. For example, if you modify `earth.txt` to create a new file `earth2.txt` that contains:

Name: Earth
Period: 365.26 days
Inclination: 0.00 degrees
Eccentricity: 0.02
Satellites: 1

- you can then compare the two files like this:

```
$ diff earth.txt earth2.txt
3c3
< Inclination: 0.00
---
> Inclination: 0.00 degrees
4a5
> Satellites: 1
```

- (The rather cryptic header "3c3" means that line 3 of the first file must be changed to get line 3 of the second; "4a5" means that a line is being added after line 4 of the original file.)
- What flag(s) should you give diff to tell it to ignore changes that just insert or delete blank lines? What if you want to ignore changes in case (i.e., treat lowercase and uppercase letters as the same)?

Exercise 1.6

```
-rwxr-xr-x 1 aturing cambridge 69 Jul 12 09:17 mars.txt
-rwxr-xr-x 1 ghopper usnavy 71 Jul 12 09:15 venus.txt
```

- According to the listing of the data directory above, who can read the file mars.txt? Who can write it (i.e., change its contents or delete it)? When was mars.txt last changed? What command would you run to allow everyone to edit or delete the file?

Exercise 1.7

- Suppose you want to remove all files whose names (not including their extensions) are of length 3, start with the letter a, and have .txt as extension. What command would you use? For example, if the directory contains three files a.txt, abc.txt, and abcd.txt, the command should remove abc.txt, but not the other two files.

Exercise 1.8

- What does `rm *.ch`? What about `rm *.[ch]`?

Lab 01: Part 2, Outside Assignment Using the Unix Command Line

Instructions

Exercises 1.9 and higher are to be completed in your own time out side of class, but are to be finished no later than the beginning of next class period (Tuesday Jan XX). Use a new file to answer the questions for part 2, called "lab01-part2.txt". This file should also be placed in your "lab01" directory.

These questions involve more of the advanced concepts, such as using pipes and I/O redirection, discussed in Lecture 04. To successfully complete some of these, you may need to combine 2 or more commands together using a pipe to obtain the requested results.

Exercises 1.G.* are more advanced questions, and only graduate (Csci 553) students are required to answer those questions.

Exercise 1.9

- Show how you would use the ls and wc commands together, connected by a pipe, to determine how many files are in the current directory. Extra credit if you display ONLY the number of files in the directory (not the number of characters in the file name, or any other information).

Exercise 1.10

- grep is one of the more useful tools in the toolbox. It finds lines in files that match a pattern and prints them out. For example, assume the files earth.txt and venus.txt contain lines like this (copy the files "earth.txt" and "venus.txt" from "/home/csci553/classfiles" into your "lab01" directory, and research the grep command using them):

```
Name: Earth
Period: 365.26 days
Inclination: 0.00
Eccentricity: 0.02
```

- grep can extract lines containing the text "Period" from all the files:

```
$ grep Period *.txt
earth.txt:Period: 365.26 days
venus.txt:Period: 224.70 days
```

- Search strings can use [regular expressions](#), which will be discussed in a [later lecture](#). grep takes many options as well; for example, grep -c /bin/bash /etc/passwd reports how many lines in /etc/passwd (the Unix password file) that contain the string /bin/bash, which in turn tells me how many users are using bash as their shell.

- Suppose all you wanted was a list of the files that contained lines matching a pattern, rather than the matches themselves—what flag or flags would you give to `grep`? What if you wanted the line numbers of matching lines?

Exercise 1.11

- The command `ls data > tmp.txt` writes a listing of the data directory's contents into `tmp.txt`. Anything that was in the file before the command was run is overwritten. What command could you use to append the listing to `tmp.txt` instead?

Exercise 1.12

- Create a directory called “bin” in your home directory. What command would you use in order to add your bin directory to the end of our PATH environment variable in the bash shell.

Exercise 1.G.1

- A colleague asks for your data files. How would you archive them to send as one file using the `tar` command? How could you compress them using either the `gzip` or `bzip` command? The `tar` command actually contains options so that you can do both the archiving and the compression at the same time. First show me how you would use `tar` along to do both the archiving and the compression at the same time. Then, for extra credit, show me how to do the same thing using the `tar` and `bzip/gzip` commands separately combined by a pipe.
- Also discover how to do the reverse of the process, both using `tar` alone and for extra credit by using a pipe and `tar` and `bzip/gzip` separately.

Exercise 1.G.2

- The `find` command allows you to search for files within and below a directory hierarchy that match a particular pattern (such as the name of the file, or the date when the file was created). How would you search for all files under the “/usr/local” directory ending with the `.c` file extension using the `find` command?
- For extra credit, use a pipe and the `xargs` and `grep` command to search all of the `.c` source files you found in the previous step for the text “KNEE”. What was the command you used and which file under the “/usr/local” hierarchy is the only `.c` source file that contains that text?