

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]`?

Exercise 1.9

- What command(s) would you use to find out how many subdirectories there are in the lectures directory?

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:

```
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?