**Bonus Material for Section 3**

**Sorting Sizes**

A common option to use with wc is -l which gives you only the line count:

> wc -l sonnets.txt Remember wildcards? How would you find the line count of all .txt files in the directory?

> wc -l \*.txt Another handy shell command is sort. By default, it sorts alphabetically, but you can specify a numerical sort with -n. Try this:

> wc -l \*.txt | sort -n

**Grep the Grep for Processes**

Open up two terminal sessions. In one, do:

> ping google.com In the other, we want to search for the ping process that we just started. Try:

> ps aux | grep ping That should give you the correct process. However, if you're running on a system with lots of other users, you may need to single out a process owned by yourself. So you could grep for your username; then grep that result for the process name:

> ps aux | grep foo | grep ping

Sometimes this chaining together of greps can come in handy. See if you can find a better way to single out processes owned by you (hint: look at options for ps).

Once you've found the pid of the ping process, you can kill it:

> kill <pid> Then the ping process in the other terminal should terminate.

**File Forensics**

Every file is of a particular type. Most of the files we've dealt with so far are what's called plain-text files, that is, normal ASCII characters. Files that are not plain-text are usually called binary files and come in all kinds of formats. Usually you can tell the format by looking at the *file* *extension*, for example .png or .pdf. Command line tools like less and cat and wc are not of much use when dealing with binary files. This little exercise will show you some useful commands for some file detective work.

First go to this link and grab my file from Dropbox. (Do this in Chrome inside your virtual machine.)

<https://www.dropbox.com/s/96q2bvtmtlvxg0w/file_forensics.tar>

(Once the page loads, click ‘Download’ and then ‘Direct Download’.)

The file is now in your Downloads directory—do a listing to see if it’s there:

> ls ~/Downloads

Let’s copy the file from the Downloads directory to your current directory with:

> cp ~/Downloads/file\_forensics.tar .

This file that you downloaded is a *tar file*, as you can tell from the handy extension .tar. A tar file is similar to a zip file--a bunch of files and/or directories have been packaged up and compressed into a single file. Do this command to 'untar' the file:

> tar -xvf file\_forensics.tar

Now you should have a directory called file\_forensics with several different files in it.

Use the cd command to move into this directory (you'll learn about navigating directories in section 4):

> cd file\_forensics Now that you're inside the directory, list the contents:

> ls -l

If you run cat on a binary file, you'll just get a bunch of gobbeldy-gook in your terminal:

> cat TwilightPicture.png

*(This may completely wreck your terminal--you may have to X out and start a new one.)*

There's a command called strings which gets only the printable characters out of a file:

> strings TwilightPicture.png

Still not very helpful in this case. Although if you look at the very beginning...

> strings TwilightPicture.png | head -n5

Hmmm, 'JFIF', 'Exif', maybe on to something.

A much more useful command to determine file type is called (what else?) file. Try the following:

> file ZELDA\_OCARINA\_OF\_TIME\_2.jpg

> file TwilightPicture.png

> file zelda\_quote.docx

So which file is trying to fool you with its bogus extension? Moral of the story--don't always trust the filename extension.

But how does the file command determine the file type? Files actually begin with *magic* *numbers* which specify the particular format.

Let's use the od command to look at actual bytes from the files:

> od -t xC ZELDA\_OCARINA\_OF\_TIME\_2.jpg | head -n1

> od -t xC TwilightPicture.png | head -n1

What's similar about these two? Check out some common magic numbers here:

https://en.wikipedia.org/wiki/Magic\_number\_(programming)

Do you see the magic numbers for the jpeg format in these two files?

Some magic numbers try to be cute--since they use hexadecimal (which has 0-9 plus the letters A-F). For instance, the magic number for the old Microsoft Word document format is D0CF11E which kind of looks like DOCFILE:

> od -t xC zelda\_quote.doc | head -n1

The new Microsoft Word format (docx) is interesting:

> file zelda\_quote.docx

Hmmm...Ubuntu says it’s a MS Word file—but my Mac tells me it’s really a zip file. Let's unzip it:

> unzip zelda\_quote.docx

Now the actual file contents are in word/document.xml:

> file word/document.xml

> cat word/document.xml

Well, you get a bunch of other junk but the Zelda quote is in there somewhere--look for the word 'secret'. Congratulations, you just hacked a Microsoft product!

There's a file there called 'innocent\_little\_script'. What does file tell you about it?

Hmm, since it’s a script, let’s try running it:

> ./innocent\_little\_script

Ha!--that should teach you a lesson. *Never run a script unless it's from a trusted source and you know exactly what it will do*.

Do some detective work on your own and figure out the magic numbers and file types of the 3 'mystery' files.