# Lab 2: Shell Commands

Nathan Jarus

January 30, 2016

## Introduction

This lab exercise will give you some experience with navigating around the shell and using IO redirection.

Feel free to consult with your favorite search engine, relevant `man` pages, the lab instructor and assistants, and your fellow labmates when you need help. Just make sure that what you turn in is your own work!

## Problem 0: Preliminaries

1. If you haven't already made yourself a folder for this assignment, go ahead and do that, then change to that directory.
2. Make yourself a text file for writing lab question answers in.
3. Put your name at the top of your answers text file so I know who you are.

## Problem 1: Output redirection

1. Run `echo foo > file.txt`. What does file.txt contain?
2. Run `echo bar > file.txt`. What does file.txt contain now? What do you conclude?
3. Run `echo baz >> file.txt`. What does file.txt contain? What does `>>` do?

(In case you're wondering, there is a reason I put this first in the lab.)

## Problem 2: `ls` and friends

1. Run `ls -al *`. Describe the output. What do you conclude about what `*` matches by default?
2. Run `ls -al .*`. What does it output, and why?

3. Why should you NEVER run `rm -rf .*`? (I am not kidding. Do not run that command.)

## Problem 3: Intermediate `man` usage

`man` contains information on lots of things other than commands. For example, there is a man page for the `passwd` file.

1. What does `man passwd` do?
2. Consult `man man`. How would you get to the man page for the `passwd` file?
   (Hint: you can do `man 3 printf` to open the `printf` man page in section 3.)
3. What environment variables does `login` set from `/etc/passwd`?

## Problem 4: A series of tubes

Your 'friend' Clayton Price has given you a text file with a story he wrote in it so that you could edit it. While you were editing, you put in a bunch of comments about his writing skills. You realize now that you probably shouldn't show him those comments... Fortunately, you put each comment on its own line, starting with a `#`:

```
My Cow Story
by Clayton Price
# okay, let's see what this nerd has to say
Homer was in a beautiful field.
# what's this, a Simpsons reference already?
There was a cow.
Homer hungrily eyed the cow, drooling excessively at the thought of steak.
# welcome to adverb city, my friend!
The end.
# That's it!?
```

You could just open the file up in an editor and remove the comments by hand, but since you are an enlightened programmer, you realize there is A Better Way.

1. Write a short program that you can pipe your story through to strip out the comments.
   (Hint 1: remember, piping puts command output into STDIN. You also probably want to output some stuff to STDOUT.)
   (Hint 2: use `getline()`, either for character arrays or for strings.)

(Note: no cheating with `grep`, `sed`, or other tools we haven't talked about yet!)

2. What command do you type to use your program to filter out the comments?

3. Run your program without piping anything into it and type some stuff in. What happens? (Hint: use Ctrl+d to send an EOF character.)

After completing this problem, bask in your newfound ability to shave yaks.

## Problem 5: Submitting your homework

That's right, you get points for handing your homework in!

1. Use the `zip` or `tar` commands to make an archive of your folder for this assignment.

2. What command will you use to make the archive?

3. Upload your archive to the assignment on Blackboard.