This is meant to be super simple and can just be “done” in spare time.

You will make typos. This will lead to debugging. Which is how we learn,

This will teach how some basic bash commands work. These exercises that are super simple. You can ignore anything after #### marks which are meant to be comments.

Step 1: Get into a Unix shell. Director nav

If you have a Mac I believe this is automagic. Or you can use a raspberry Pi, or load a virtual machine and install ubuntu. At some point, you need to access the shell.

```
pwd    # this shows you your current directory. It stands for print working directory.
ls                       ## this shows you what is in your directory
ls -la    ### this will show you what is in your directory, and attributes (d = directory, r = read
                         permissions, w = write permissions x = execute permissions… for different roles)
mkdir bashtest
ls
cd bashtest
pwd                      #your in your directory
cd ..                    # shortcut to move UP a directory
pwd
cd bashtest
```

Step 1: files and output

```
ls                       # nothing there, no files
touch myfile              # this created a file, very useful.
ls
echo hi                  # this just echos, to your console (standard out)
echo hi > myfile         # this redirects it to your file. The single carrot will overwrite
cat myfile               # now your file has a word in it
echo hi again >> myfile  ## the double carrot will append
cat myfile
echo hi overwrite > myfile  ## the single carrot overwrote.

ls -la                   # view permissions on your file. It has read and write, no execute.
echo echo this is a command in my file > myfile    ### this will put this echo command in your file
cat myfile
./myfile                         #try to run your file. It will error out it isn’t executable, yet
chmod 755 myfile         ### this will change permissions. Chmod does that.
ls -la                   ### that little x makes it executable
./myfile                 # it should now run the file
rm myfile                # remove your file
ls                       # validate it is gone
```

```
cd ..                     # go up one directory
rmdir bashtest            # remove your directory
```

Step 3: add a user

```
useradd deleteme    # try to add a user

### It probably fails due to permissisions. Back in the old days people ran around with admin privileges
                          (root), and we broke stuff regularly. So you need to run w/ admin priv, and not
                          everyone has it. You should, if you own your machine. Sudo = super user do
sudo useradd deleteme             # creates user
passwd deleteme                   # this should fail w/ permissions issue
sudo passwd deleteme              # now put it in


groups ## this lists groups. Pick one that seems boring, I am picking games

sudo usermod deleteme -G games   ### we added games group to user deleteme. Usermod modifies the
user

userdel deleteme                  #
sudo userdel deleteme
```

Step 4: add a package

We’re using ubuntu. I think a lot of examples use this in the “wild” or possibly exams, because its free. It uses apt. for package mgmt. I am going to add a random package that isn’t big.

```
apt-get install xpuzzles

### this should fail. You should get used to seeing these failures, they are permissions based. they kind
of look the same in different constructs (above earlier failures, and python, whatever)

sudo apt-get install xpuzzles  ## install package

sudo apt remove xpuzzles   ## remove package
```