**Day 1: Command Line & Vim**

**I. Command Line Tutorial**

The command line is a text interface for your computer. It's a program that takes in commands, which it passes on to the computer's operating system to run.

From the command line, you can navigate through files and folders on your computer, just as you would with Finder on Mac OS or Windows Explorer on Windows. The difference is that the command line is fully text-based.

The advantage of using the command line is its power. You can run programs, write scripts to automate common tasks, and combine simple commands to handle difficult tasks - making it an important programming tool.

The tutorial below is for is for unix-based systems such as Linux and Mac OS X. For Windows users, you will need to Google equivalent commands. You can do most of this by going to Programs – Anaconda – Anaconda Prompt.

**First things first, download the Unit8Student folder and put it in your Documents Folder.**

**Opening your terminal**

On a Mac, go to Applications – Utilities – Terminal.

Once it’s open, you may want to pin it to your home screen toolbar since it’s so useful.

**Important Commands**

**1. pwd, as in “print working directory”**

Type pwd and then enter into your terminal. This should list your current working directory. Mine says “/Users/shareshianl” and yours probably says something similar. If you ever want to know which folder (aka directory) that you are currently in, you can type pwd.

**2. ls (lowercase L, lowercase S, as in “list”)**

Type ls and then enter. This lists all of the files in your current directory.

If you type “ls –a” this will list all of the files in your directory, including hidden ones. (If you scroll through, you’ll probably see files that start with a period. These are hidden files that you wouldn’t see from typing “ls” alone.

**3. cd (change directory)**

“cd” changes directories. Move into your Documents folder by typing “cd Documents”. Then type “ls” and you should see your Unit8Student folder. Type “cd Unit8Student” to move into that folder. Then, type “ls” to see the contents of that folder.

**4. mkdir (make directory)**

“mkdir” will make a new directory (aka folder) with the name that you specify. Now that we are inside Unit8Student, let’s make a new folder called “terminal\_practice” (Note: You don’t want folder names to contain spaces so we’ll use the underscore instead.”) Type “mkdir terminal\_practice”

Type “ls” again and now you should see a new folder called terminal\_practice located there.

Just because you’ve made the directory doesn’t mean that you are inside it. Type “cd terminal\_practice” to now move inside that folder.

Type “ls” again and you’ll see that this is currently an empty directory.

**5. cd ..**

Let’s say you wanted to navigated back up a directory into Unit8Student. Type “cd ..” and then enter.

Now type “pwd” and you’ll see that you are back in Unit8Student. But actually, we want to be back in terminal\_practice so type cd terminal\_practice once again to get there.

**6. touch**

“touch” makes new files. Make a new file called “helloworld.py” by typing “touch helloworld.py”

Type “ls” and you should now see the helloworld.py file there.

**6. cp (copy)**

Let’s say we wanted to copy this helloworld.py file into our Documents folder. I’ll type

“cp helloworld.py ~/Documents” (Note: The format will always be “cp filename newlocation”) The tilde denotes our home directory, which is the first directory that you were put in when you opened up the terminal. It is one directory above Documents and Desktop.

**7. rm (remove)**

We don’t actually want the helloworld.py file so let’s remove it. Type “rm helloworld.py”. If you type “ls” again you’ll notice the file is gone.

Now let’s remove the terminal\_practice directory itself. Type “cd ..” to navigate back up to the Unit8Student directory. If you type “ls”, you should see the terminal\_practice directory. To remove directories, you need an extra option that specifies “r” for recursive, which deletes everything in the directory. Type “rm -r terminal\_practice”. If you type “ls”, you should no longer see this directory.

8. **Up arrow**

Press the up arrow several times. Note that it lists the past commands that you’ve used. You can always navigate up to repeat a command instead of typing it all in.

**9. Tab key**

Let’s say we want to navigate into the Day1CommandLine folder contained within the Unit8Student directory. Instead of typing the whole directory name, just type “cd Day1” (no space after Day1) and then **press the tab key**. You should now see the entire command listed, and you now press enter.

10. **cat**

Now that you are within the Day1CommandLine, type “ls” to view the contents of this folder. You should see a hi.txt file. You can view the contents of the file by typing “cat hi.txt”.

**II. RUNNING PYTHON IN YOUR TERMINAL**

We don’t need Jupyter Notebooks or Anaconda or PyCharm or anything to run a Python program – we can do it directly from our terminal!

1. You should see an examply.py file located within your Day1CommandLine folder. Type “python example.py” and enter your name to see what the function does.

2. What the heck! We just ran a Python program from the command line. To view the contents of the file, type “cat example.py”.

3. **Opening Jupyter Notebooks (or any program!) through your terminal**

We can now open Jupyter Notebooks through your terminal. Type “jupyter notebook” and press enter and give it a few seconds to boot up. (Note: if you still haven’t installed Anaconda on your computer, this won’t work.) To quit a program, type Control-C.

**III. VIM and other text editors**

Inside the Day1CommandLine folder, make a “helloworld.py” file once again by typing “touch helloworld.py”. This file is empty so we’ll want to add stuff to it. There are many options for editing text files – VIM, emacs, vi, Atom, Sublime, etc. VIM is one of the ones that comes pre-installed on a Mac so let’s use that.

**1. Opening vim**

Type “vim helloworld.py”

**2. Insert mode**

We need to be in the insert mode in order to edit the file. Type “i” to do so.

**3. Edit the file**

We can now edit the file. Type the following. Note: Don’t use the tab key- use four spaces to indent.

print(‘Hello, World!’)

for i in range(10):

print(i)

**4. Save the file**

To save the changes, press escape and then type “ :w”. (This writes the file).

**5. Exit vim**

To exit vim, type “:q” (This stands for quit).

6. **Run the program**

Type “python helloworld.py” to run the program.

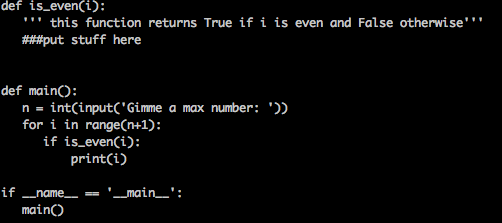
7. **Edit a progam**

Type “vim example.py” to edit the example I gave you. Use the above steps to edit the line name = input("What is your name? ") to the line name = input("What is your first and last name?”). Rerun the program to see your changes.

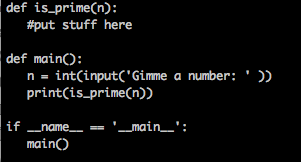
**Terminal Exercises:**

1. Create a new directory within Unit8Student/Day1CommandLine called terminal.

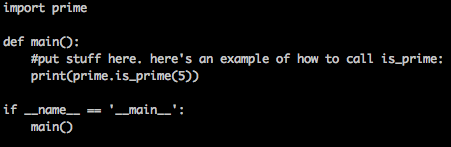
2. Use vim to create a new file within that directory called evens.py that contains an even function. Your program should have the same format as the screenshot below. Use it to print all of the even numbers between 0 and 10 from the command line.



3. Use vim to create a program called prime.py that asks the user for a number and returns True if the number is prime and False if it isn’t. Create a function called is\_prime within this file that helps you, and put the rest within the main function. Then run this program from the command line. Your file should look like this:



4. Use vim to create a program called allprimes.py that asks the user for a number and prints all of the prime numbers between 2 and that number. Use the is\_prime function that you already wrote to help you. Here’s how you’ll import it into allprimes.py:

).