Tutorial One: Unix and Basic Python

Imad Pasha
Chris Agostino

February 5, 2015

1 Introduction

The situation is a little bit weird this year with the lab computers not yet set up. Because of this, you will likely be completing this tutorial on your own computer. If you own a Mac, the steps below should give you some practice working with UNIX; however, since PC users will have to skip this section, it is optional. (Furthermore it involves making directories on your own computers so we'd have no way to check it anyhow). Feel free to take a look at it if you have a Mac, but otherwise just skip to the Python section.

2 Unix

We went over the basics of moving around within a UNIX system (as well as basic file and directory manipulation) on Monday. Just as a reminder, those lecture slides are posted on the website, and the textbook has lots of examples of UNIX to remind you of the steps. You can also check out the UNIX Cheat Sheet if you get stuck.

2.1 Making Some Directories

Open up terminal on your computer; the easiest way is usually using spotlight to search for it. The first thing we are going to do is make some directories to be organized for this class.

1. Figure out where you are by typing the command to show the current directory. Then change directories to get either to your home (username) or documents directory (where you will create a folder for this class).

2. Create a directory in your home directory (or in your documents, etc) called "ay98", or "python_decal" (or whatever you want).

3. cd into the directory you just made, and create directories for tutorials, homeworks, and your final project

4. Delete the directory you just made for homework, since we decided that homeworks would be extensions of the tutorials and would be turned in together!

5. If you feel comfortable in UNIX move on. If not, practice cd'ing in and out of the folders you've created using the shortcuts we covered in lecture. You can also experiment with making, deleting, and moving some files within your system.

3 Python

In your tutorials directory, go ahead and launch the ipython interpreter (if you are on a mac and in terminal- for windows just open canopy and create a new file). In this section, we will be giving you some things to try out to familiarize yourself with how basic math, variables, and data types work. At the end, we will show you how to create a text file with all the inputs and outputs made during that ipython session, which you'll then turn in on the website. Don't worry if you make mistakes along the way and have to retry calculations- we won't be grading this such that you have to get it right on the first try.

1. Add, subtract, multiply, and divide some integers of your choice together. In your division, choose numbers that illustrate the limitations of integer division.

2. Have python output 2, 3, and 4 squared, then 2, 3, and 4 to the 5th power.

3. Create a variable named "sum" and set it equal to the sum of two 3 digit integers. Print sum.

4. Recast your variable sum as a float, and then make a variable "div" equal to sum divided by 2. Print div.

5. Recast div as a string and print it.

6. Type print div[0] into python and see what happens.

7. Make a variable called "nums" and set it equal to a list containing 42, 46, 54, 65, and 90 as elements.

8. type nums[3] into python and see what happens.

    - In python, typing a variable name alone is the same thing as typing print variable_name.

9. Using the syntax above, set a variable "total" equal to the sum of the first two elements in "nums". Print total. The output you get should be 88. If it isn't, try to figure out what went wrong. Did you get 100? Look closely at which element of "nums" nums[3] seems to refer to.

10. Import numpy as np (in ipython)

11. Create a numpy array of several numbers (anything you want). Print it.

12. Set a variable "zeros" = np.zeros(20) and print it to see what it is

13. Experiment with the functions np.arange(start,stop,step) and np.linspace(start,stop,number of divisions). You can type help(np.arange) if you want to pull up the documentation, or you can google the functions to figure out how they work.

14. Use numpy functions to create an "x-axis" of numbers and y as a sin function of that x array.

15. To see what your function looks like, type 'import matplotlib.pyplot as plt', and then type 'plt.plot(x_arr,y_arr)' (where the variables are your x axis and sin function you made). Finally plt.show() will show your plot. Does it look like a sin? Try using (4*np.pi) as one of your arguments to create your x-axis to be 2 full periods. If you use np.arange you'll notice it looks really choppy, but you can use np.linspace to make it look smoother! Try to get a good looking sine wave.

16. Type %history -n -o -f yourname.tutorial_one.txt

17. Type "exit" to exit the ipython interpreter.

18. In terminal, use the command to list files and check that yourname.tutorial_one.txt has indeed been created.

19. Type "vim yourname.tutorial_one.txt" in terminal, and it will open the text file. Make sure that it has the history of your ipython session written. Then type ":wq" to close vim. Vim can be kinda scary. There's a vim guide online, and you can call one of us over if you have questions.

3.1 Closing Comments

In this tutorial, the main goal was to have you explore the ipython interpreter a bit and see what it is capable of. Most of the things we asked you to do were in the lecture, but others were precursors to what is coming up. Don't worry too much if you didn't quite understand what some of those not-yet-covered commands were doing. We will cover them in time, and repeated exposure is the best way to learn!

As a closing tip on numpy, as we get into working with it more next week: if there is a math sounding function that you think there *should* be in numpy somewhere, it probably is. And the name for it is probably just np.the-obvious-name-of-the-function. Want an exponential? np.exp(). Want a cosine? np.cos(). Want to transpose a matrix? np.transpose(). So about 80% of the time, you can just guess the numpy function you need. For the other twenty, its on to google!

4 Homework

There's no written homework for this week! This tutorial (the history text file you made) is due next Wednesday at 7pm, before the start of the next tutorial. To turn it in, open up firefox on the lab computer, navigate to the decal website, and click the homework tab, then

click the turn in link at the top! (Please enter your name such that we can match it to the class list we have).

Please take some time to skim through the textbook for chapters 3 and 5, on writing basic programs, and plotting! You'll need that for next week.