Introduction to Python
Day One Exercises

Stephanie Spielman
Email: stephanie.spielman@gmail.com

1 UNIX Exercises

1. Launch a Terminal session and navigate to your home directory with the **cd** command. Remember, there are three ways to do this:

   ```
   cd
   cd ~
   cd /path/to/home/directory/ # replace your home directory's full path
   ```

   Using the commands **cd**, **pwd**, and **ls** (and **ls -la**), examine the directory structure of your system. Spend a few minutes (no more than 5!!!) figuring out where different files and directories are located so that you understand your file system organization by navigating forward into sub-directories and back into parent directories and listing contents. *The purpose of this task is to become comfortable with your computer's organization.*

2. Navigate to your home directory, and perform the following tasks. After performing a task that copies, moves, or removes a file/directory, enter the command **ls** to see how things have changed.

   (a) Use the command **mkdir** to create a new directory called "blob".

   (b) Use the command **touch** to create a new file called "blerg.txt".

   (c) Use the command **echo** along with the symbol **>** to add the sentence "I'm writing to a file!" to blerg.txt.

   (d) Use the command **mv** to move this file into the directory blob. Then, enter **ls**. What do you notice?

   (e) Navigate into the directory "blob". Make a copy of blerg.txt called "blerg2.txt".

   (f) Use the command **echo** and the symbol **>** to write the line "Another sentence!" to blerg2.txt. Now use **less** to examine the contents of blerg2.txt. What do you notice?

   (g) Make a new copy of the file blerg.txt using the command **cp**, called blerg3.txt. Enter **ls**. What files now exist in this directory?

   (h) *Append* the line "Another sentence!" to blerg3.txt using **echo** and the symbol **»**.

   (i) Navigate back to your home directory (try using the code **cd ..** for this), and use the command **rm -r** to delete the blob directory.

   (j) Use the **rm** command to delete the file blerg.txt.

3. For this exercise, you will use the included file sequences.fasta. This file contains sequences from the Influenza virus PB2 gene. Navigate to the directory where this file is located, and perform the following tasks:

(a) The UNIX command **wc** stands for "word count". This command counts the number of lines, characters, and bytes in a given file. Enter the command **wc sequences.fasta** to display this information. Compare the result with the file size as displayed by **ls**. Do you see any overlapping numbers?

(b) Use **wc** with the argument **-l** to determine just the number of lines in sequences.fasta.

(c) Enter the command **head sequences.fasta** to view the top few 10 lines of this file. Consult the documentation for **head** using the command **man head**, and figure out how to specify a different number of lines. Enter **q** to exit from the man documentation, and use your new knowledge to display the first 16 lines of the file.

(d) Create a new file called "lastseqs.fasta" which contains the *last* 8 lines of sequences.fasta (hint: the command **tail**, which is basically the opposite of **head** should be useful!). For this task, *do not use* **touch**.

2 Python Variables Exercises

You can write Python code in two different ways: directly via the Python interpreter or via a script, which you can then call from the command line. To use the interpreter directly, simply type **python** into your command line. Directly interfacing with the Python interpreter is an excellent way to test out small pieces of code, but it is *not a good way* to develop code. Using scripts, on the other hand, preserves your code in a text file (with the extension .py) so that you always have your Python code saved and accessible.

For these exercises, you can use either the interpreter or a script, although I strongly recommend that you save all code in a script (with lots of comments!) for future reference!!

Most importantly, you should *always print your results after every step you take*. Printing output is the only way to be sure your code has worked properly!

2.1 Working with Lists

1. Define a list variable called animals which contains the following six entries: "monkey", "giraffe", "shark", "caterpillar", "ctenophore", and "squid". Perform the following tasks on this variable:

   (a) Use indexing to create a new list with just the items "monkey", "giraffe", and "shark".

   (b) Use negative indexing to extract "squid" from this list.

   (c) Use the list method **.append()** to add the animal "spider" to the end of the list. Use the **len** function to see the length of the animals list after appending.

   (d) Use indexing to change the second entry in animals to "cat".

   (e) Use the list method **.pop()** to remove the 3rd entry from this list.

   (f) Create a second list called plants which contains the entries "tree", "flower", "bush", and "grass". Next, create another list called organisms, in which the first entry is the animals list and the second entry is the plants list, using this code: **organisms = [plants, animals]**. You have just created a "list of lists", or a nested list. Use the **len()** function to determine the length of this new list. Is this what you expected or not?

(g) Using indexing, replace the word "tree" in organisms with "redwood".

(h) Using indexing, extract the words "grass" and "caterpillar" from the organisms list in order to create this new list: **`["grass", "caterpillar"]`**. Bonus - for this question, try to incorporate the **`.index()`** list method.

2.2 Working with Strings

1. Define two string variables, one with the value "hello" and the other with the value "goodbye". Using concatenation (the + sign), create this new string: "helloGOODBYEHELLOgoodbye" from the two variables defined. Print your result to be sure it is correct.

2. Create a string variable with the contents "abcdefg". From this variable, create a new string in which the "e" has been replaced with "X", i.e. "abcdXfg". Hint: use indexing and concatenation. Print a final statement that says: "My new string is abcdXfg."

3. Define a string variable with the following value: "Hi, my name is Joe. I got a wife and three kids. I work in a button factory." Perform the following tasks on this variable:

   (a) Extract the first 10 characters from the string.

   (b) Count of the number of occurrences of the letter "e" in the string.

   (c) Extract characters 8 - 14 from the string, and save this as a variable. Then, convert this new string to uppercase.

   (d) Convert this string into a list of lower-case words like this: **`["hi", "my", "name", "is", "joe", "i", "got", "a", "wife", "and", "three", "kids", "i", "work", "in", "a", "button", "factory"]`**. For this task, you will need to use the **`.split()`** string method and indexing. Finally, use the **`len()`** function to determine how many words are in this new list.