

# Introduction to Python

## Day One Exercises

Stephanie Spielman

Email: stephanie.spielman@gmail.com

### 1 Bash Exercises

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.*

1. Download today's course materials from the course website. Locate that folder on your computer (either with Finder or with terminal, depending on your preference). It should be called "day1\_materials". Once you find the directory that contains your downloaded materials, determine the directory's *path* using the `pwd` command. Remember this information!
2. Navigate to your home directory (you can type either `cd` or simply `cd` for this), and perform the following tasks:
  - Use the command `mkdir` to create a new directory called "class1". Enter that directory using the command `cd`.
  - Use the command `cp` to copy the directory of today's course materials into current working directory ("class1"). Hint: When copying a folder, then you will need to use the argument `-r`, for example `cp -r <directory to copy, including path> <destination>`
  - Once you have successfully performed the last step, navigate into the "day1\_materials" directory using `cd`. Use the `mv` command to rename the file called "original.txt" to "new.txt". Confirm that the file was renamed with `ls`.
  - Copy the file "new.txt" from its current directory into the directory "class1/", which is one level above the working directory, using `cp`. Confirm that the file has been successfully *copied* (not moved!).
  - Navigate back a directory into "class1/" (using the code `cd ..`) and remove the just-copied file "new.txt" using the `rm` command.
  - Create a new directory called "temp" using the command `mkdir`. Move the file whose current path is "class1/day1\_materials/new.txt" into "temp/" from the "class1/" directory (do not enter the course materials directory or temp!).
  - Now, use `ls` to list the contents of "temp/". There should be a single file in this directory called "new.txt" if the previous step worked. Finally, remove the "temp/" directory with the command `rm -r` (think: why use `-r`?).

## 2 Python 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 If statements

First, define the following variables:

- `a = -4.2`
- `b = 55`
- `animal = "python"`

1. Use an `if` statement to check if the variable `a` is less than 100. If it is true, then print the statement "It is less than 100." Run the code to check if it works.
  - (a) Modify the previous `if` statement to include an `else` component. Inside the `else`, write code to print the statement "It is not less than 100." Run the code to check if it works.
  - (b) Modify the `if/else` to create an `if/elif/else` construct. The `if` should test the `>100` condition and the `elif` should test the `<100` condition. Modify print statements to provide appropriate feedback. Run the code to check if it works.
  - (c) Instead of just printing within the `if` statements, let's define some new variables. In the `if/elif/else` construct, implement the following: If `a` is less than 100, define the variable `"dog = beagle"`. If `a` is greater than 100, define the variable `"dog = spaniel"`. If neither is true, define the variable `"dog = labrador"`. *After* the `if/elif/else` construct, print out the newly created `dog` variable to check that it was assigned correctly.
2. Write an `if/else` statement to check if the variable `b` is even or odd (Hint: the modulus operator `%` returns the remainder of an division. For example, `5%2` is 1, and `5%2` is 0.). Within each `if/else`, print whether the variable was odd or even.
3. Write an `if/else` statement to check if there are more than 10 letters in the variable `animal` (Hint: use the `len()` function!). Within each `if/else`, print an informative message.
4. In Texas, you can be a member of the elite 'top 1%' if you make at least \$423,000 per year. Alternatively, in Hawaii, you can be a member once you start making at least \$279,000 per year! Finally, if you live in New York, you need to earn at least \$506,000 a year to make the cut. Andrew is CEO of Big Money Company, and he earns \$425,000 per year, and Stacey is CEO of Gigantic Money Company with an annual salary of \$700,000. Use `if`-statements to determine, and print, whether Andrew and Stacey would be considered top 1%-ers in Texas, Hawaii, and New York.