# Introduction to Programming with Python: Day 1

Jackie Champagne
University of Texas at Austin

June 7, 2021

## 1 Introduction

Welcome to the TAURUS/NSF REU Python workshop! The goal of this workshop is to get you acquainted with basic navigation of the macOS command line terminal as well as some basic Python syntax. The terminal will soon become your home for navigating directories, editing script files, and starting up programs. This document will be a tutorial for some command line basics, and in the last section I will link you over to the beginning of the Python tutorial.

**1. Open a Terminal.** If you're working on a Mac, this is under Applications>Utilities. You can also search for applications using Spotlight (command+space), and it's a good idea to add it to your dock.

**2. Find out where you are.** Whenever you want to know what directory you're working in, type `pwd`. This will print out the path to where you're located; since you just opened a terminal, it should print your home directory, i.e., /Users/yourusername. To see what files and folders are in that directory, type `ls`.

**3. Create a directory.** It's a good idea to stay organized with dedicated folders. To create a directory the same way you'd use "New Folder," use the command `mkdir` followed by the name of a directory. For example, type `mkdir research`.

**4. Navigate to a new directory.** To **c**hange **d**irectories, use the command `cd` followed by the name of the directory. Try navigating to your research directory. Then type `pwd` to confirm you made it to `research`. To go back up one directory, type ".." (without the quotes). To go back to your home directory at any point, simply `cd` will get you there. Navigate back to `research`.

# 2 Using VI

**4. Get started with a text editor.** There are a few command line text editors out there, but I personally stick with vi. Let's practice a few basic vi commands. To create a file, type `vi` followed by a filename, including the extension. For example, `vi test.txt`.

**5. Edit your file.** This should have opened a window for you. To begin inserting text, type `i`. The window should say `--INSERT--` at the bottom. Begin typing something: Hello world! To get out of insert mode, press `ESC`.

**6. Change your text.** You'll notice that you can navigate through your text using the arrows on your keyboard. To change a single character, navigate to the character and press `r` and then the replacement character (but don't go into insert mode). Change the O in hello to a P.

**7. Delete some text.** Get rid of a single character by pressing `x` on the character. Get rid of the P in Hellp.

**8. Insert a new line.** To go directly into insert mode on a blank line, type `o`. Insert a line that says "this is pretty cool!"

**9. Delete some more text.** To get rid of a whole line, type `dd`. Delete "this is pretty cool!"

**10. Save your file.** Always make sure you exit insert mode before saving. To save without quitting, type `:w` and press enter. To quit without saving, type `:q`. To save and quit, type `:wq`. Save and quit this file.

There is a full dictionary of vi functionality here:
http://www.cs.colostate.edu/helpdocs/vi.html.

# 3 Moving/Copying Files

By now you should have one file in your research directory called test.txt. You can copy files directly from the command line, which is highly recommended if you are editing code or modifying data but you want to keep the original.

**11. Copy your file.** To make a copy of your file, type `cp` followed by the name of the file to copy and then the name of the copy. Copy your test.txt to test2.txt.

**12. Rename your copied file.** The command `mv` can be used either to rename a file or move it to a new location. To rename it, type `mv` followed by the current name of the file followed by the new name of the file. If the last argument contains the path to a directory that exists, it will move it there, but if it is not a directory, it will rename the file. Rename test2.txt to blah.txt.

**12. Move your copied file.** To move a file to a new location, type `mv` followed by the name of the file followed by the directory you'd like to move it to. The name of the directory should be the full path, like /Users/jackie/Downloads, but there are a few helpful shortcuts. "[tilde]/" is the shortcut for your home directory, so "~/Downloads" would work. ".." is the shortcut for "one directory above here" like we learned before. Move blah.txt one directory above you.

**13. Delete your files.** To delete a file, type `rm` followed by the filename. Navigate to your home directory and delete blah.txt. Be very careful with `rm` as it cannot be undone!!

**14. Copy a directory.** Navigate back to `research`. To copy or move a full directory rather than a single file, add a space and then `-r` to add a recursive option. You should have already downloaded the workshop materials for today, in a directory called `workshop`, which is probably in your Downloads

folder. Copy it over to your research directory: `cp -r ~/Downloads/workshop .` – the period means your current location, but you could also type out "~/Users/yourname/research".

This should be all you need to know in terms of basic terminal manipulation, but there are lots of other useful skills to use in the bash shell, like creating aliases and bash scripts as well as secure copying from other computers. You'll likely learn that along the way with your research, but I've left some instructions in another document in the workshop tarfile.

# 4 Loading up Python

To use Python from the command line, you can always type `ipython` or `python` to begin a session. I personally don't like using iPython from the command line, because I find notebooks much more helpful for debugging and drafting purposes. You can use a notebook by typing `jupyter notebook`. When you're done with the notebook, press ctrl+C to exit out of the terminal session.

One last useful Terminal thing: the jupyter notebook, and some other programs, will effectively 'kill' a terminal session, meaning you can no longer use this window until the program is closed. You can open new Terminal windows, or keep it available for use by typing & after the name of the program, i.e. `jupyter notebook &`. When using the ampersand, you'll see a number next to the job. When you want to quit the notebook session, you can type `kill [jobnumber]`. Open up the Workshop Day 1 notebook.

Note: the MacOS distribution of Python has historically had some issues with scisoft, which occasionally messes up some Python paths.The computers assigned to you should already be configured with a different distribution called **astroconda** which allows you to manage different virtual environments and has a bunch of astronomy packages already installed. If you type "jupyter notebook" and it throws an error, you may need to reset the python path, which you can do by opening `~/.bash_profile` in a text editor and typing `unset PYTHONPATH`, or we may need to investigate your conda install. If you never need more information about environments, check out some documentation here: https://astroconda.readthedocs.io/en/latest/