UNIX command line

Also called terminal, shell, etc...

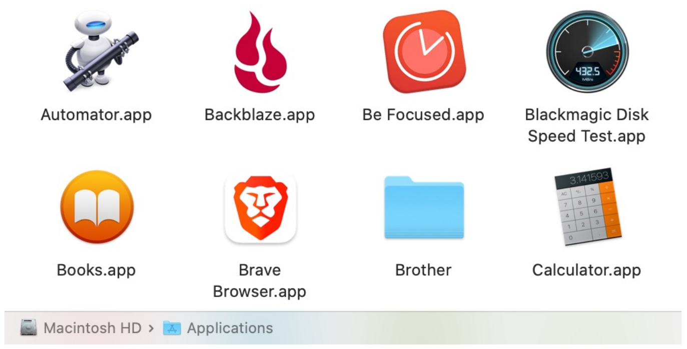
Terence Parr
MSDS program
University of San Francisco

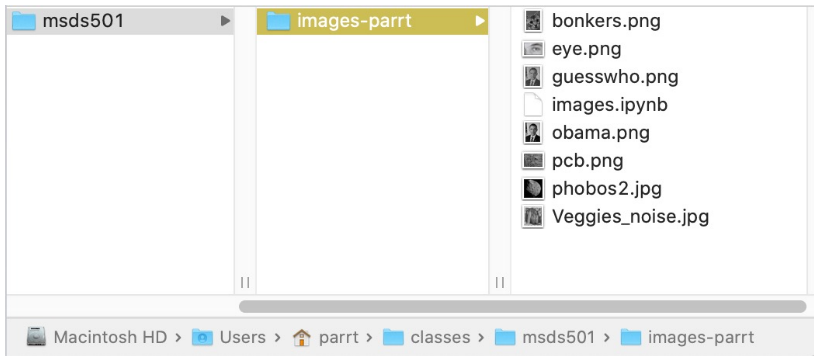
See also [Videos from Philip Guo (pythontutor guy) on terminal]()

UNIVERSITY OF SAN FRANCISCO

Controlling your machine w/o a GUI

- The Mac "Finder" is a graphical way to launch and control programs as well as manipulate files and folders on the disk



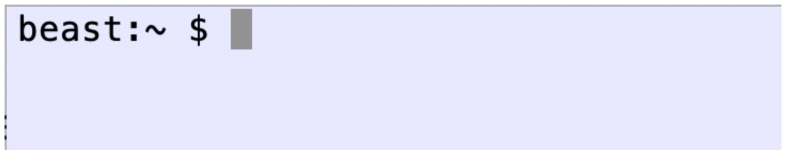


- But, the "terminal" is an old-school text-based interface that has a number of advantages

The terminal is running a *shell*

Terminal.app

- The UNIX shell is an interactive domain-specific language used to control and monitor the UNIX operating system (Mac OS)
- It is also a programming language, though we'll use it mostly to move files around, execute commands, ...
- You need to get comfortable on the UNIX command line because, at minimum, you will control cloud computing facilities using the command line
- We type commands at the **$** *prompt* and hit return to execute



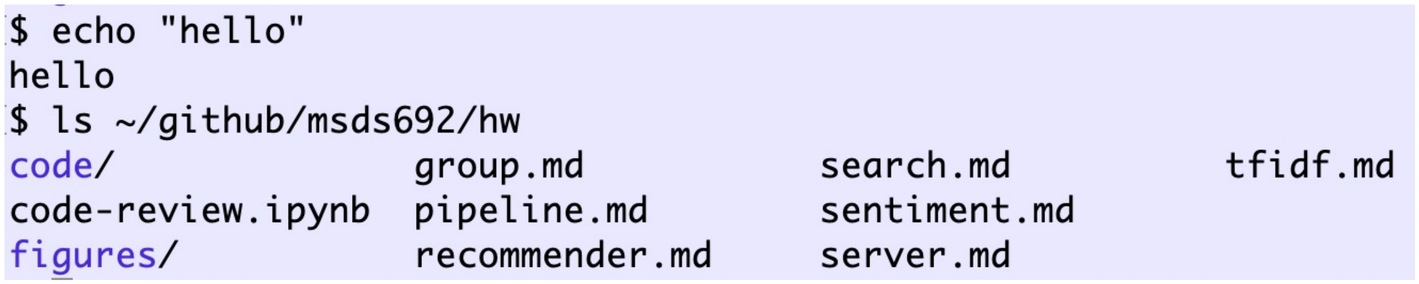
```
beast:~ $ 
```

(*Note: OS X changed the default shell to be zsh not bash.*)

UNIVERSITY OF SAN FRANCISCO

Commands are analogous to Python function calls, including arguments

- In Python we say **print("hello")**, but in the shell we don't use parentheses around arguments and use spaces not commas



```
$ echo "hello"
hello
$ ls ~/github/msds692/hw
code/             group.md          search.md         tfidf.md
code-review.ipynb pipeline.md       sentiment.md
figures/          recommender.md    server.md
```

- Notation: *command arg1 arg2 arg3*
  vs python: *command(arg1, arg2, arg3)*

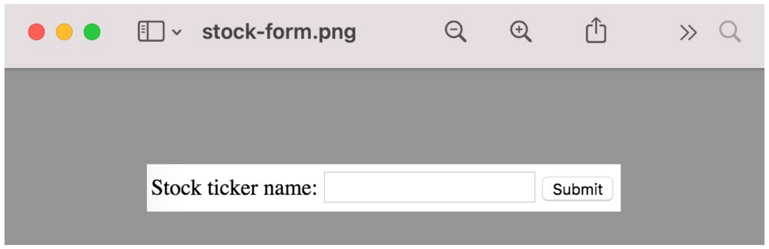
UNIVERSITY OF SAN FRANCISCO

Executing programs by opening files

- Instead of double-clicking on an image file, for example, we can tell the terminal to open it

```
$ ls ~/github/msds692/hw/figures
article1.png              hashtable4.png            search-page.png
article2.png              json-jq.png               stock-form.png
articles.png              parrt-follows.png         testdatahtml.png
aws-ami.png               parrt-tweets.png          testdataxml.png
bbc.png                   pipeline.graffle          trump-follows.png
datahtml.png              pipeline.png              trump-tweets.png
dataxml.png               pythonanywhere.png        tsla-json.png
hashtable-empty.png       redbang.png               tsla-xml.png
hashtable2.png            search-file-page.png      twitter-app-creation.png
hashtable3.png            search-interview.png
```

```
$ open ~/github/msds692/hw/figures/stock-form.png
$ 
```



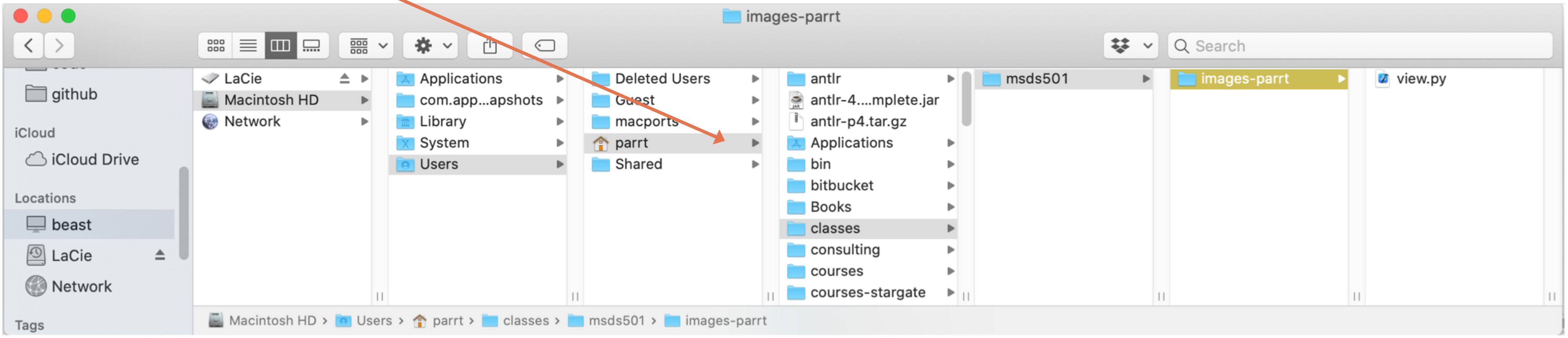
Looking at file contents

- **cat** *filename*: show entire file
- **head** *filename*: show first n lines of file
- **tail** *filename*: show last n lines of file

```
$ head -5 hours-worked.csv
"LOCATION","INDICATOR","SUBJECT","MEASURE","FREQUENCY","TIME","Value","Flag Codes"
"AUS","HRWKD","TOT","HR_WKD","A","1979",1834,
"AUS","HRWKD","TOT","HR_WKD","A","1980",1836,
"AUS","HRWKD","TOT","HR_WKD","A","1981",1820,
"AUS","HRWKD","TOT","HR_WKD","A","1982",1802,
beast:~ $ pwd          # print current working directory
/Users/parrt
beast:~ $ 
```

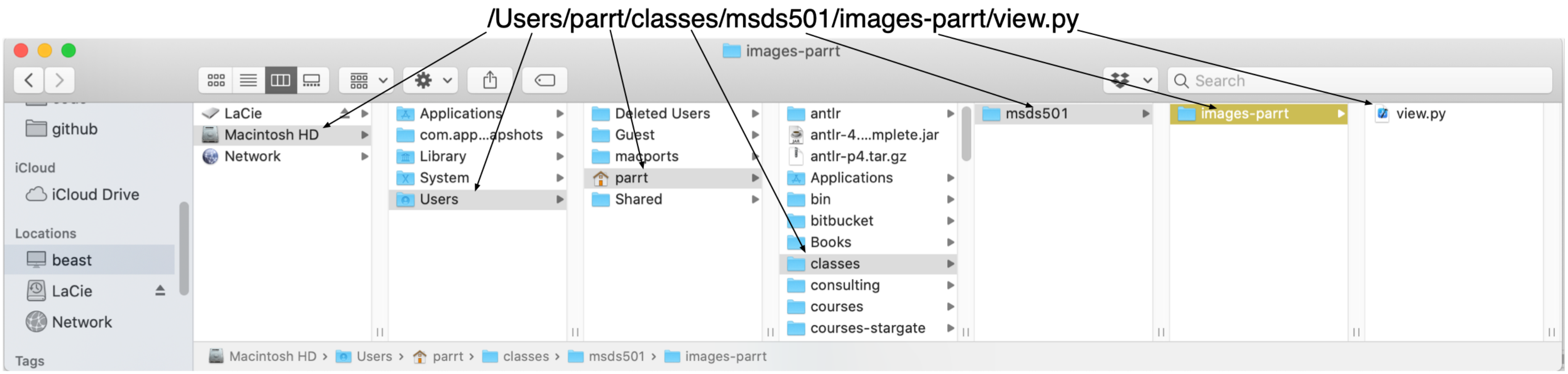
Current working directory

- The shell has a number of state variables, one of which is the *current working directory*, and it is by far the most important
- Most commands execute relative to this working directory
- When terminal opens, working directory is set to your user home directory abbreviated as "~"



Path specifications

- As we saw previously, the folders or directories on your disk represent a tree; files in a folder represent leaves of the tree
- A *fully-qualified* path to a file starts with "/" and consists of the directories used to reach the file from the root of the disk; root is "/" and we separate path elements with "/"
- *Relative pathnames* do not start with "/" and are relative to WD

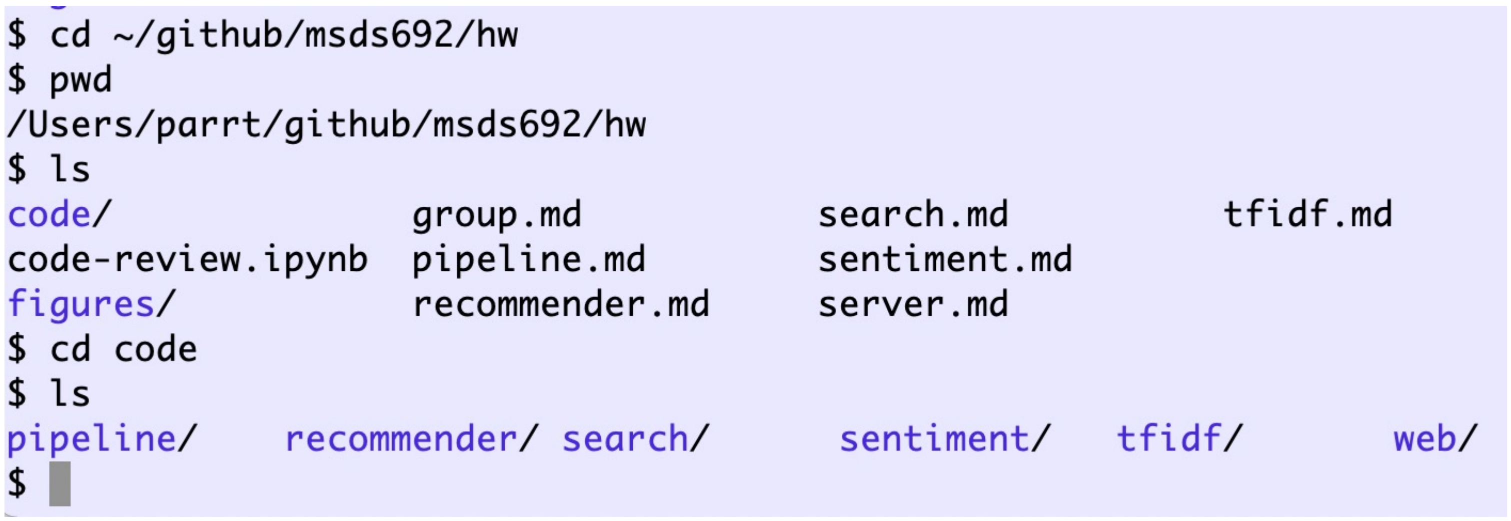


Useful directories to know about

- Other then your home directory, ***/Users/**youruser*, you should know about:
  - /Applications
  - /tmp
  - /usr/local (such as brew's install area /usr/local/Cellar)
  - /bin, /usr/local/bin
- Also to configure zsh (your shell), see file **~/.zshrc**

cd: change working directory

- To "move" around the disk hierarchy/tree, use **cd** command to change the current working directory (i.e., where am I?)

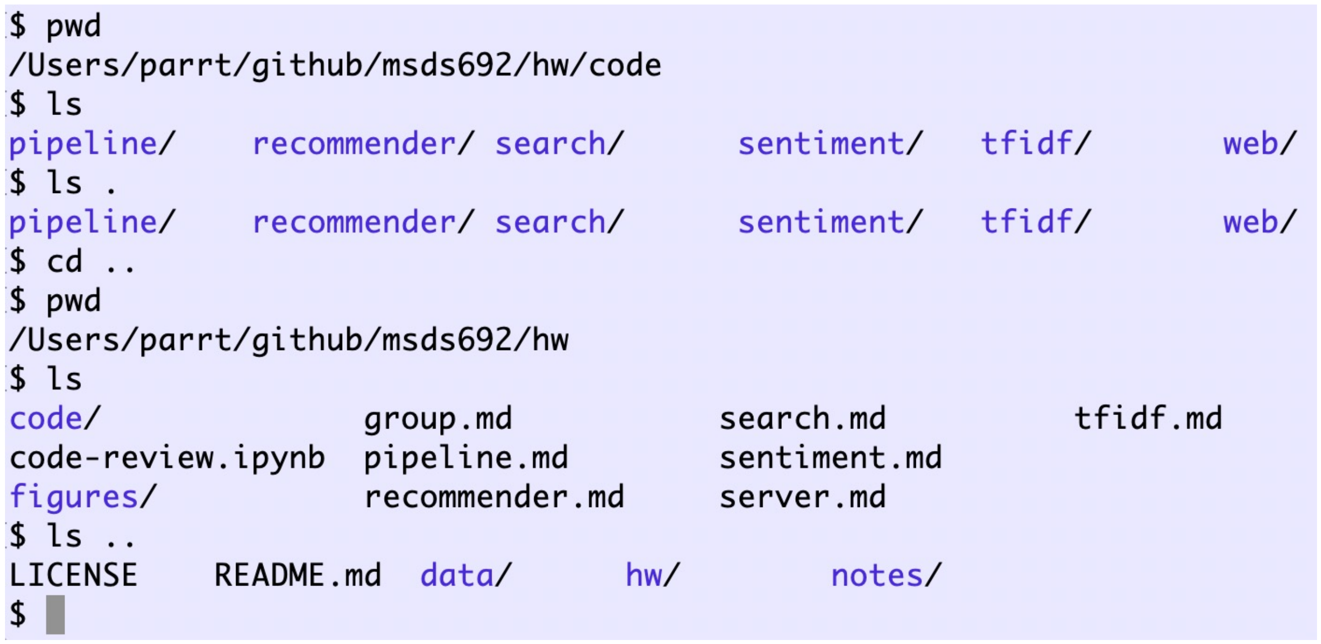


```
$ cd ~/github/msds692/hw
$ pwd
/Users/parrt/github/msds692/hw
$ ls
code/              group.md          search.md          tfidf.md
code-review.ipynb  pipeline.md       sentiment.md
figures/           recommender.md    server.md
$ cd code
$ ls
pipeline/    recommender/ search/      sentiment/    tfidf/       web/
$ 
```

UNIVERSITY OF SAN FRANCISCO

Dot, Dot-Dot

- Dot "." means current working directory
- Dot-Dot ".." means directory above current working directory



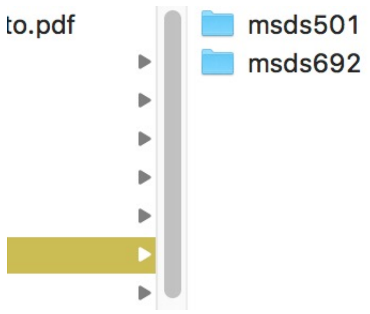
```
$ pwd
/Users/parrt/github/msds692/hw/code
$ ls
pipeline/     recommender/ search/       sentiment/    tfidf/        web/
$ ls .
pipeline/     recommender/ search/       sentiment/    tfidf/        web/
$ cd ..
$ pwd
/Users/parrt/github/msds692/hw
$ ls
code/              group.md           search.md          tfidf.md
code-review.ipynb  pipeline.md        sentiment.md
figures/           recommender.md     server.md
$ ls ..
LICENSE    README.md  data/      hw/        notes/
$ 
```

UNIVERSITY OF SAN FRANCISCO

Manipulating files and directories

- **mkdir** *newdirname*: make directory
- **cp** *source target*: copy file or directory
- **mv** *oldname newname*: rename or move files/dirs

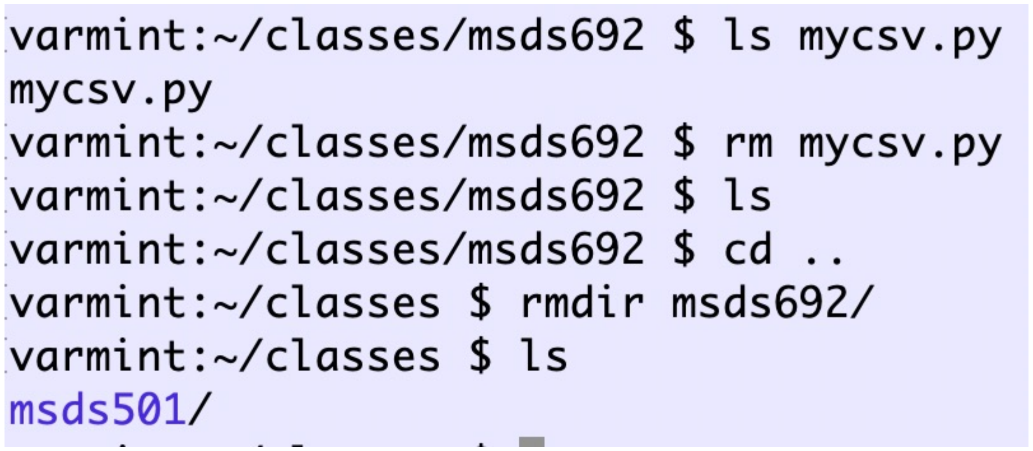
watch dir appear



```
varmint:~/classes $ ls
varmint:~/classes $ cd ~/classes
varmint:~/classes $ pwd
/Users/parrt/classes
varmint:~/classes $ ls
msds501/
varmint:~/classes $ mkdir msds692
varmint:~/classes $ cd msds692
varmint:~/classes/msds692 $ ls
varmint:~/classes/msds692 $ cp /Users/parrt/github/msds692/hw/code/pipeline/mycsv.py .
varmint:~/classes/msds692 $ ls
mycsv.py
varmint:~/classes/msds692 $
```

Removing files and directories

- **rm** *filename*: remove file
- **rmdir** *dirname*: remove empty directory
- **rm –rf** *dirname*: remove directory and everything underneath it

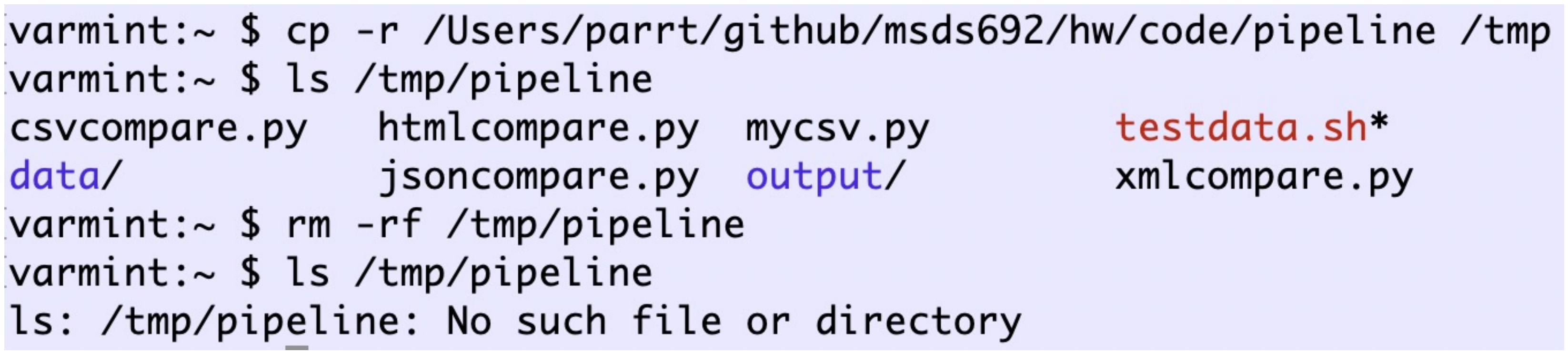


```
varmint:~/classes/msds692 $ ls mycsv.py
mycsv.py
varmint:~/classes/msds692 $ rm mycsv.py
varmint:~/classes/msds692 $ ls
varmint:~/classes/msds692 $ cd ..
varmint:~/classes $ rmdir msds692/
varmint:~/classes $ ls
msds501/
```

UNIVERSITY OF SAN FRANCISCO

Removing files and directories Cont’d

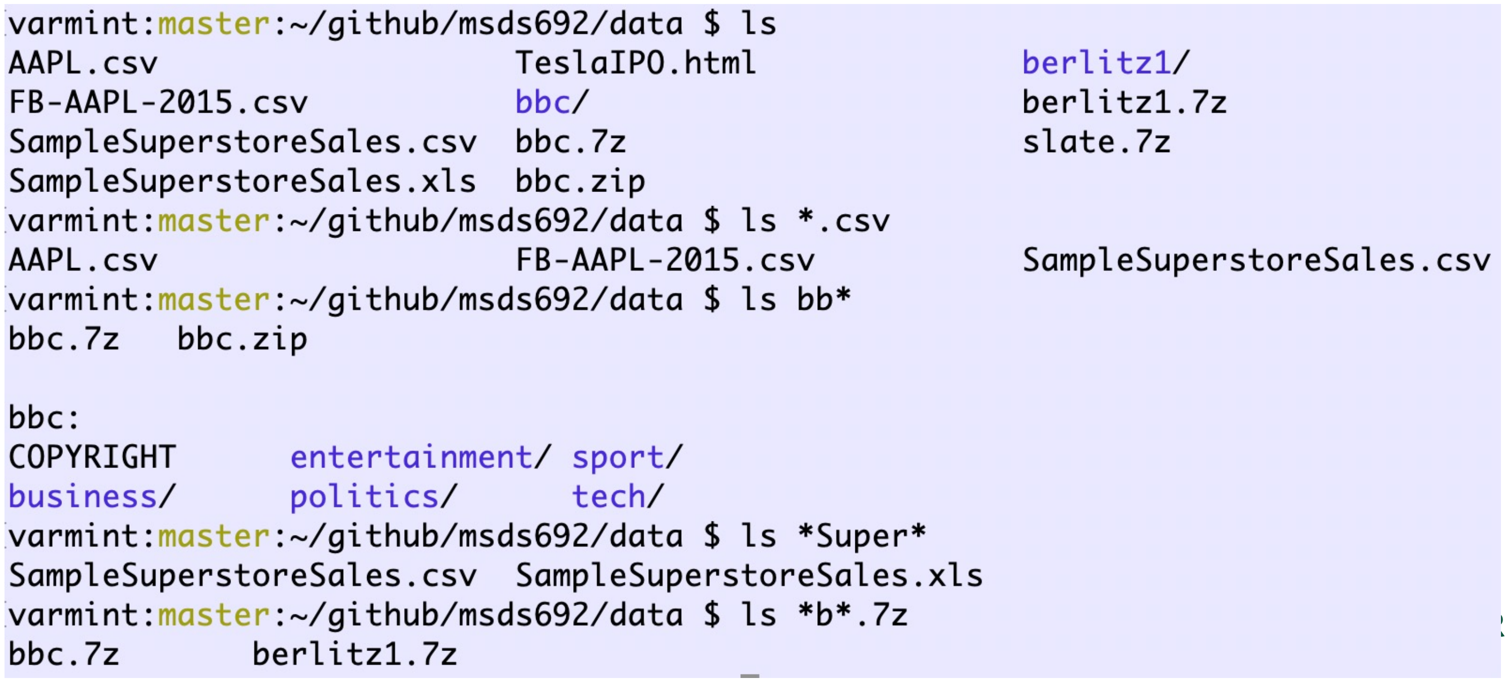
- **rm** *filename*: remove file
- **rmdir** *dirname*: remove empty directory
- **rm** –rf *dirname*: remove directory and everything underneath it



```
varmint:~ $ cp -r /Users/parrt/github/msds692/hw/code/pipeline /tmp
varmint:~ $ ls /tmp/pipeline
csvcompare.py   htmlcompare.py  mycsv.py        testdata.sh*
data/           jsoncompare.py  output/         xmlcompare.py
varmint:~ $ rm -rf /tmp/pipeline
varmint:~ $ ls /tmp/pipeline
ls: /tmp/pipeline: No such file or directory
```

Wildcards

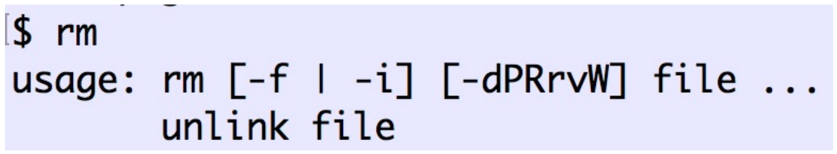
- Star "*" means roughly "any word that matches", such as all files
- Good example of something that's impossible with a GUI; imagine that you have 1000 datafiles and you need to delete all files whose names have the word “old”



```
varmint:master:~/github/msds692/data $ ls
AAPL.csv                    TeslaIPO.html              berlitz1/
FB-AAPL-2015.csv            bbc/                       berlitz1.7z
SampleSuperstoreSales.csv   bbc.7z                     slate.7z
SampleSuperstoreSales.xls   bbc.zip
varmint:master:~/github/msds692/data $ ls *.csv
AAPL.csv                    FB-AAPL-2015.csv           SampleSuperstoreSales.csv
varmint:master:~/github/msds692/data $ ls bb*
bbc.7z   bbc.zip

bbc:
COPYRIGHT        entertainment/ sport/
business/        politics/      tech/
varmint:master:~/github/msds692/data $ ls *Super*
SampleSuperstoreSales.csv   SampleSuperstoreSales.xls
varmint:master:~/github/msds692/data $ ls *b*.7z
bbc.7z       berlitz1.7z
```

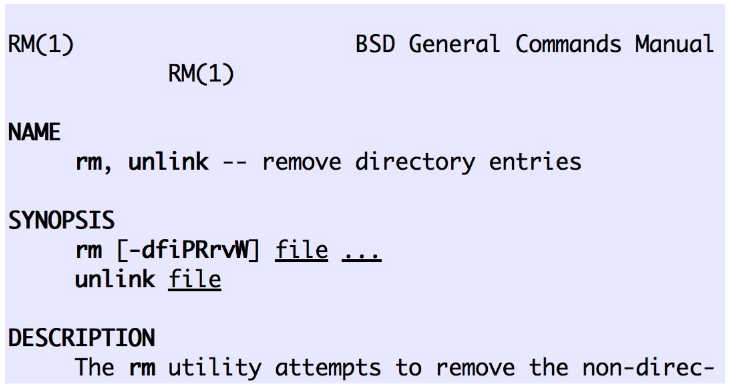
Manual pages; Getting help

- Google search (your shell is called **zsh**)
- Stackoverflow
- Often you can type the command without arguments and it will give a help line:



```
$ rm
usage: rm [-f | -i] [-dPRrvW] file ...
       unlink file
```

- Or, type “**man rm**” to get the manual page:



```
RM(1)                    BSD General Commands Manual
           RM(1)

NAME
     rm, unlink -- remove directory entries

SYNOPSIS
     rm [-dfiPRrvW] file ...
     unlink file

DESCRIPTION
     The rm utility attempts to remove the non-direc-
```

Installing more terminal commands

- There are lots of useful UNIX programs available that are not currently installed on your machine
- The **homebrew** program installs new code for you
- For example, we'll likely install a Python library that requires an open source graphing tool called graphviz:

```
$ brew install graphviz
```

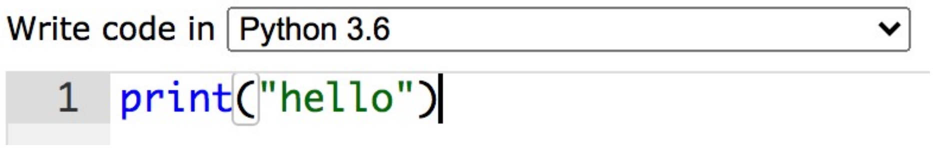
Python-related commands

UNIVERSITY OF SAN FRANCISCO

(Hit Ctrl-D to exit python, go back to shell)

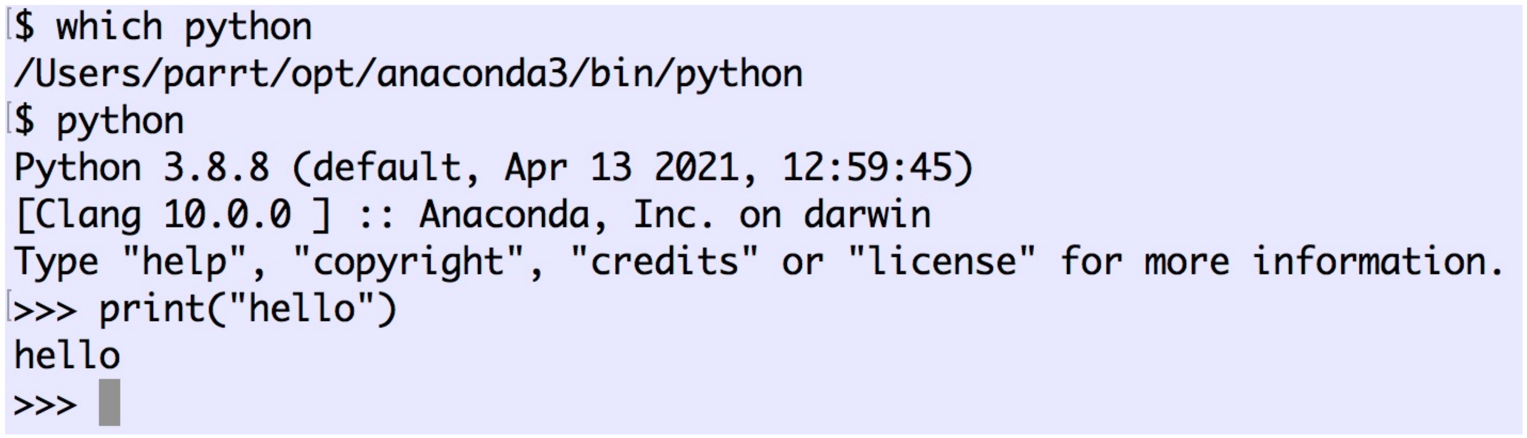
Interactive Python console from terminal

- Pythontutor gives us a text box to edit code and a run button to visualize the execution



```
print("hello")
```

- Can also execute Python interactively line by line from the shell



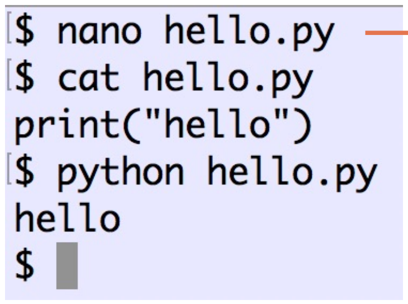
```
$ which python
/Users/parrt/opt/anaconda3/bin/python
$ python
Python 3.8.8 (default, Apr 13 2021, 12:59:45)
[Clang 10.0.0 ] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> print("hello")
hello
>>> 
```

Warning: there are multiple copies of Python on your Mac possibly

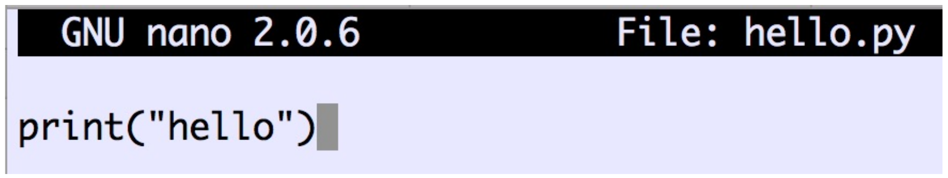
UNIVERSITY OF SAN FRANCISCO

Executing python scripts

- All of the code we type into the Python console disappears when we exit and return to the command line
- Save python into a .py file, using your favorite editor, such as **nano**



```
[$ nano hello.py
[$ cat hello.py
print("hello")
[$ python hello.py
hello
$ 
```



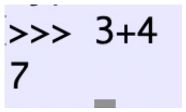
```
 GNU nano 2.0.6                 File: hello.py

print("hello")
```

(To Save, hit Ctrl-X then "Y" to save changes then hit return at "File Name to Write: …")

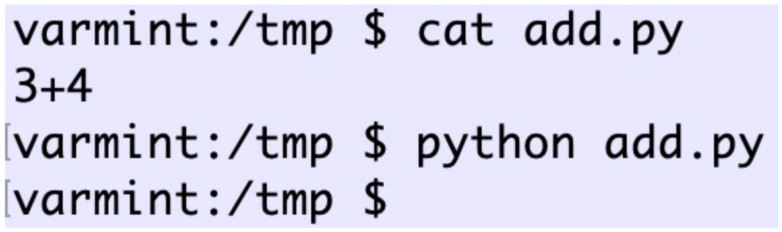
- We call this a Python script, program, or simply a Python file
- Use "**python** *file.py*" from terminal to execute the script in "batch mode"
- NOTE: *file.py* must be a TEXT file, w/o formatting like in M$ Word files

UNIVERSITY OF SAN FRANCISCO

Warning: interactive console vs scripts

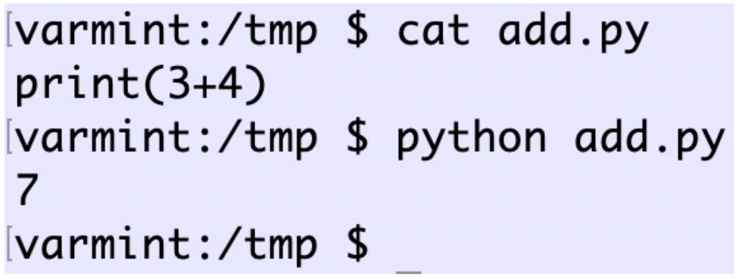
- In the console or Jupyter lab, typing an expression evaluates it and displays the result
- In a script file, no output is generated unless you use **print()**
- Compare console: 

```
>>> 3+4
7
```

to script: 

```
varmint:/tmp $ cat add.py
3+4
[varmint:/tmp $ python add.py
[varmint:/tmp $
```

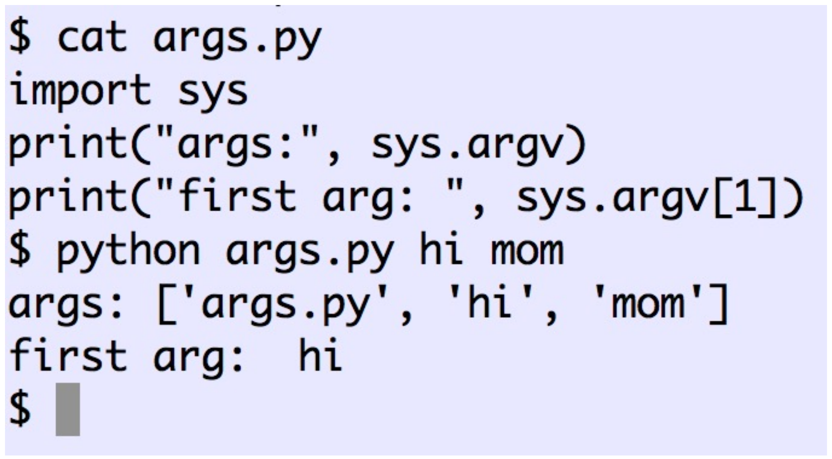
- Must use **print()** to get output:



```
[varmint:/tmp $ cat add.py
print(3+4)
[varmint:/tmp $ python add.py
7
[varmint:/tmp $
```

Passing arguments to python scripts

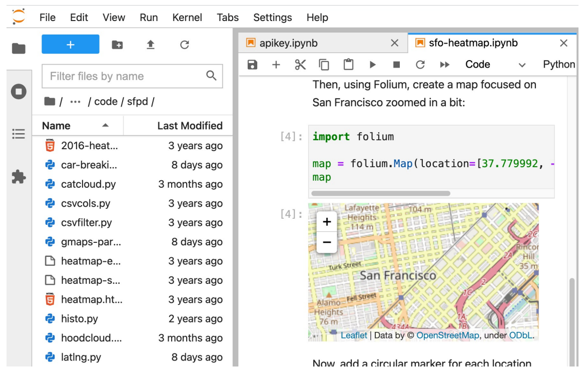
- Sometimes python scripts need information about their environment, such as where to find data files
- The executing script can access arguments from the command line used to launch it using the **sys** package:



```
$ cat args.py
import sys
print("args:", sys.argv)
print("first arg: ", sys.argv[1])
$ python args.py hi mom
args: ['args.py', 'hi', 'mom']
first arg:  hi
$ 
```

This is a very good reason why you should never use spaces in your directory or file names

Launch Jupyter Lab (notebooks)



Name	Last Modified
2016-heat...	3 years ago
car-breaki...	8 days ago
catcloud.py	3 months ago
csvcols.py	3 years ago
csvfilter.py	3 years ago
gmaps-par...	8 days ago
heatmap-e...	3 years ago
heatmap-s...	3 years ago
heatmap.ht...	3 years ago
histo.py	2 years ago
hoodcloud....	3 months ago
latlng.py	8 days ago

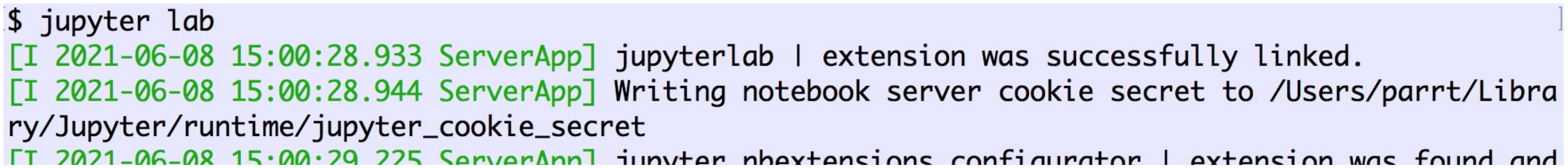
Then, using Folium, create a map focused on San Francisco zoomed in a bit:

```
[4]: import folium

     map = folium.Map(location=[37.779992, -
     map
```

[4]:

Now, add a circular marker for each location

- A "notebook" is a sequence of "cells" that can contain code, output, notes, etc.
- A notebook is stored like a script but into a *file*.**ipynb** file not **.py**
- A server that we launch from the command line starts up a Python interpreter and connects to a browser window where we can make notes and execute code snippets interactively



```
$ jupyter lab
[I 2021-06-08 15:00:28.933 ServerApp] jupyterlab | extension was successfully linked.
[I 2021-06-08 15:00:28.944 ServerApp] Writing notebook server cookie secret to /Users/parrt/Libra
ry/Jupyter/runtime/jupyter_cookie_secret
[I 2021-06-08 15:00:29.225 ServerApp] jupyter_nbextensions_configurator | extension was found and
```

Tutorials:

https://www.youtube.com/watch?v=RFabWieskak
https://www.dataquest.io/blog/jupyter-notebook-tutorial/

UNIVERSITY OF SAN FRANCISCO