NEUG,PSYC231: Experimental Data Analysis in Python

John Serences, jserences@ucsd.edu

October 5th, 2020

Class 00

Introductions…

- **John Serences (jserences@ucsd.edu)**
- Professor in Department of Psychology, Neuroscience Graduate Program
- UCSD undergrad, Johns Hopkins graduate school, Salk Institute postdoc
- Research: selective attention and memory systems in humans, focus on using computational models to link brain activity and behavior
- http://serenceslab.ucsd.edu/

Important resources for students

- **UCSD’s principles of community**
- **Counseling and Psychology Services (CAPS)**. “CAPS provides FREE, confidential, psychological counseling and crisis services for registered UCSD students. CAPS also provides a variety of groups, workshops, and drop-in forums.”
- **CARE** at the Sexual Assault Resource Center is the UC San Diego confidential advocacy and education office for sexual harassment, sexual violence and gender-based violence (dating violence, domestic violence, stalking).
- **Office for the Prevention of Harassment & Discrimination (OPHD)**. OPHD "works to resolve complaints of discrimination and harassment through formal investigation or alternative resolution.”

Central repository for class material

- https://github.com/JohnSerences/NEU-PSYC-231-Fall2020

- Good to familiarize yourself with GitHub – commonly used tool for collaborative programing

Goals of the course

- Develop solid understanding of the Python language and the Jupyter environment.
  - Open science, data and code sharing
  - Replicability, best practices
- Introductory course for people new to Python and new to coding
  - Experience in another language may help, but no programming experience is necessary
- Why learn to code?
  - Its actually really fun to solve complex problems…by the end of this course you will be impressed with how much you can do

Goals of the course…

- Bring everyone along – coding is something that many of you may fear, but all of you can do!
- If you’re good at it, get even better by helping other people…best way to REALLY learn

Most important slide in the class!

- **Don’t be afraid to make mistakes**.
  - You’ll see me make plenty of mistakes, and that **is a normal part of the process**.

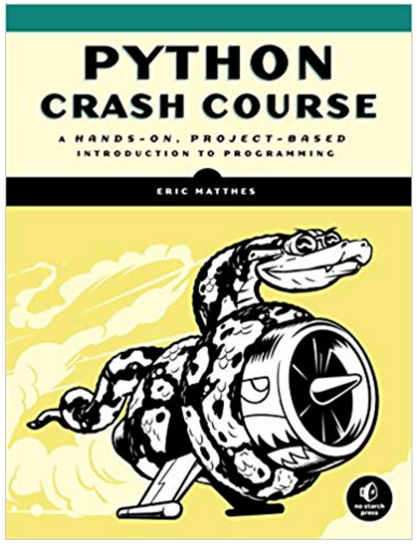
- **Embrace mistakes as learning opportunities.**
  - While learning how to fix a mistake, you will also learn 10 other cool things that you didn’t even know that you needed to know.

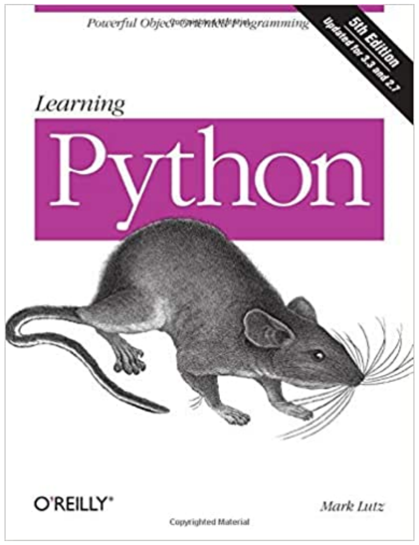
Most important slides in the class!

- Try to ignore the snarky jerks who lurk in stack overflow and other forums…
- Meta example of snarky comments coming into a discussion about how to prevent snarky comments:
  - https://meta.stackoverflow.com/questions/372285/flagging-snarky-comments-to-is-it-possible-questions

Books that might help?

- Python crash course : a hands-on, project-based introduction to programming by Eric Matthes, Nov 2015
  - New on Amazon for about $30, used for $2.95-$15
- Learning Python, 5th Edition by Mark Lutz
  - $40 new, used ~$15-$20
- **<u>Make sure its PYTHON 3 (not 2)!!!</u>**





Problem sets

- Each week, there will be a problem set to work on during class.
- Provides hands-on practice that ***is necessary*** to develop fluency.
- Grade: at end of quarter, turn in all of your notebooks.
  - Focus not on perfect code, but on getting through the exercises
  - You'll get out of this only what you put into it….

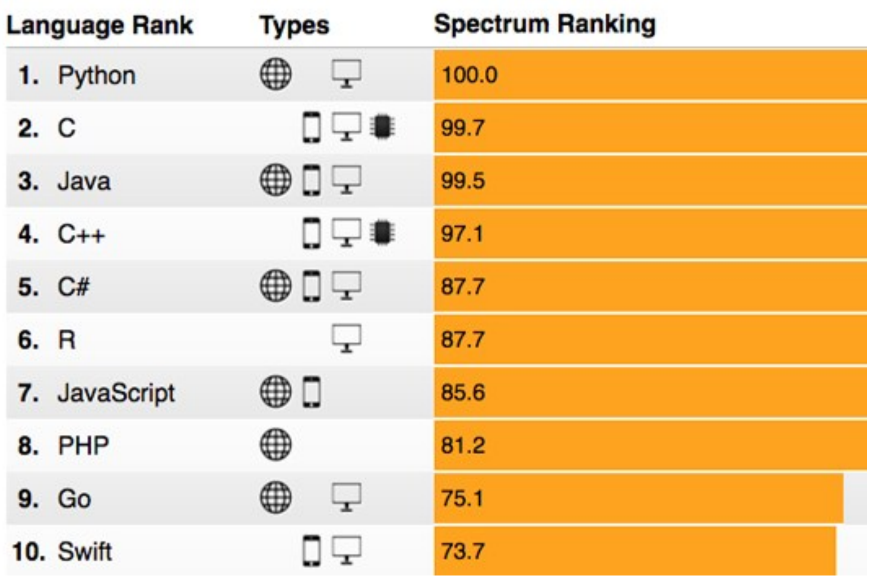
Why learn Python?

- Incredibly flexible for data analysis (modules/libraries)
- Quick development for prototyping/production, excellent GUI support
- Support for generating and compiling C code (faster execution)
- Good balance of flexibility and power against complexity of language/constructs (e.g. Visual Basic/Matlab vs C/C++ vs. Assembly)

Python vs. Matlab

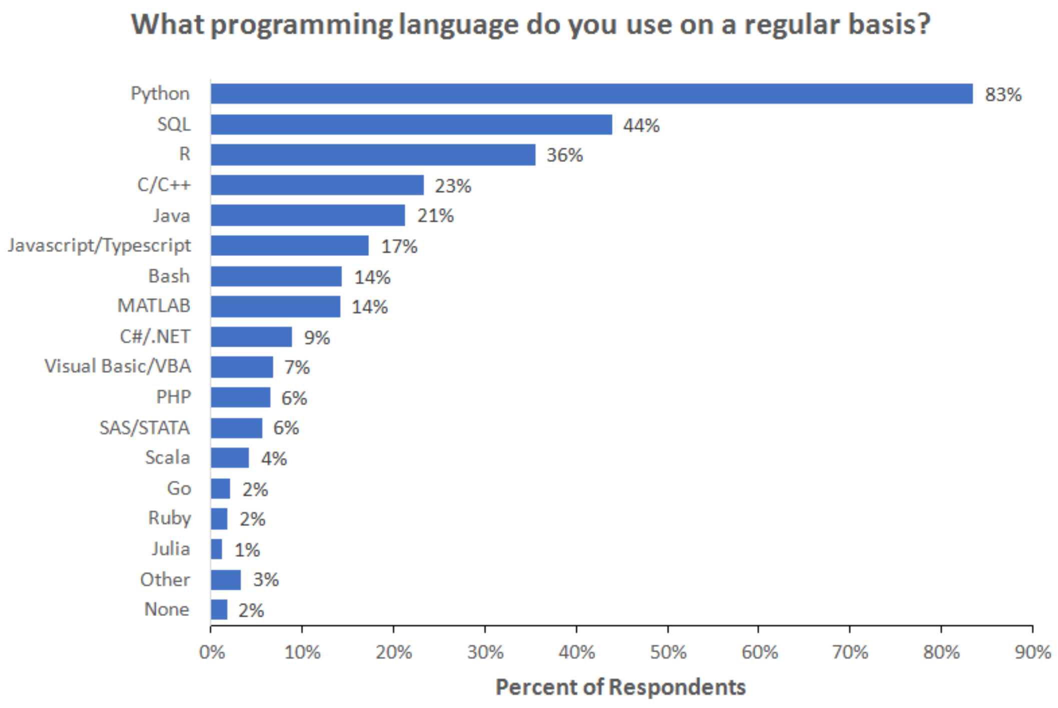
- Programming style
  - 0 vs 1 based indexing
  - block indent
  - () vs [] for function calls, array indexing
- Use in industry/academia
  - Python is far more common in industry – prototyping to full development
  - Many new branches of analysis/computing are led by the Python community with Matlab playing catch-up
- Bleeding edge – good and bad
- Open development community – good and bad
- Some references (note the affiliation of authors …)
    - https://www.mathworks.com/products/matlab/matlab-vs-python.html
    - https://pyzo.org/python_vs_matlab.html
    - http://phillipmfeldman.org/Python/Advantages_of_Python_Over_Matlab.html
    - Perhaps the most balanced (and relevant): https://blog.thedataincubator.com/2017/10/matlab-vs-python-numpy-for-academics-transitioning-into-data-science/

Bottom line on Python vs Matlab (and other languages)



Language Rank	Types	Spectrum Ranking
1. Python	🌐 🖥	100.0
2. C	📱 🖥 ▪	99.7
3. Java	🌐 📱 🖥	99.5
4. C++	📱 🖥 ▪	97.1
5. C#	🌐 📱 🖥	87.7
6. R	🖥	87.7
7. JavaScript	🌐 📱	85.6
8. PHP	🌐	81.2
9. Go	🌐 🖥	75.1
10. Swift	📱 🖥	73.7

IEEE Spectrum 2017



Kaggle, 2018

Vy Vo
PhD, computational neuroscience UCSD

- **Research Scientist @ Intel Labs**
- Studying ways to improve machine learning & artificial intelligence, inspired by findings in computational & cognitive neuroscience.
- Skilled at distilling large data sets using Python, Matlab, and R.
  - Specialties include: supervised learning with linear and nonlinear classifiers; dimensionality reduction with PCA/ICA; multivariate, linear, logistic, and other types of regression; model fitting using gradients or grid search; and more.
  - Optimizing analysis on large and noisy datasets using parallel programming.

Python programming environments

- Many approaches/environments to develop code
    - Command line interface…pretty basic, no frills
    - Traditional IDE (Integrated Development Environment)…from simple to fancy (.py files)
        - **PyCharm**, IDLE, ATOM, Sublime, Spyder
    - Notebooks…Integrated web-based environment
        - iPython notebook, aka: Jupyter

Jupyter Notebook Environment (https://jupyter.org/)

- Contains live code, equations, visualizations and narrative text all in one place
- Easy to share – cross platform and (should) run on any computer and any OS and will produce the same output
- Google Colab is a Jupyter notebook environment that requires no additional setup
  - Runs on virtual machine that is set up when your session starts (and is recycled after session idle)
  - Supports Python 2.7 (deprecated) and Python 3.7 (current active version)
  - All major extensions (modules/libraries)
  - Easy to share directly on drive or after downloading in open source .ipynb format

Key concepts for today

- Variable: symbolic name that refers to an **object** (or to a chunk of data)
    - Objects can be a letter string, number, list of letter strings or numbers, etc.
    - Many specialized types of object for storing each type of information: **str , int, float, list**, dictionary, etc.
    - The data is contained within the object
    - A **variable** is a useful (i.e. readable/memorable) label for an object

Key concepts for today

- Different objects can be used for different purposes
  - If you want to store a name or a human-readable label for data, use a string
  - If you are dealing with numbers, use an int or a float
  - If you are dealing with a bunch of strings or numbers, use a list (array)

Key concepts for today

- Method: a function that is available for a given type of object (or available to the variable that refers to the object)
  - You can use methods to manipulate the data that are assigned to a variable
  - Example: if you have a list of words, the sort() method will re-arrange the list in alphabetical order
  - Object oriented programming!

Some shortcut keys for Google Colab

- On a PC cntrl = control key, on Mac cntrl = “apple” command key
  - New cell above: cntrl+M A
  - New cell below: : cntrl+M B
  - Convert to code cell: cntrl+M Y
  - Convert to text cell: cntrl+M M

- Run a cell (execute code or display markdown): cntrl+ENTER