Programming Principles in Python (CSCI 503/490)

Introduction

Dr. David Koop

Northern Illinois University

Python Experience?

Programming Principles?

Why Python?

Productivity

Libraries, Libraries, Libraries

What about speed?

Northern Illinois University

Administrivia

- Course Web Site

- TAs: Naga Jyothi Kota & Angel Prathyusha Koyi

- Syllabus

  - Plagiarism

  - Accommodations

- Assignments

- Tests: 2 (Feb. 21, Apr. 3) and Final (May 6)

- Course is offered to both undergraduates (CS 490) and graduates (CS 503)

  - Grad students have extra topics, exam questions, assignment tasks

Academic Honesty

- **Do not cheat!**
- You will receive a **zero** for any assignment/exam/etc. where cheating has occurred
- You will **fail** the course if you cheat more than once
- Misconduct is reported through the university's system
- You **may** discuss problems and approaches with other students
- You **may not** copy or transcribe code from another source
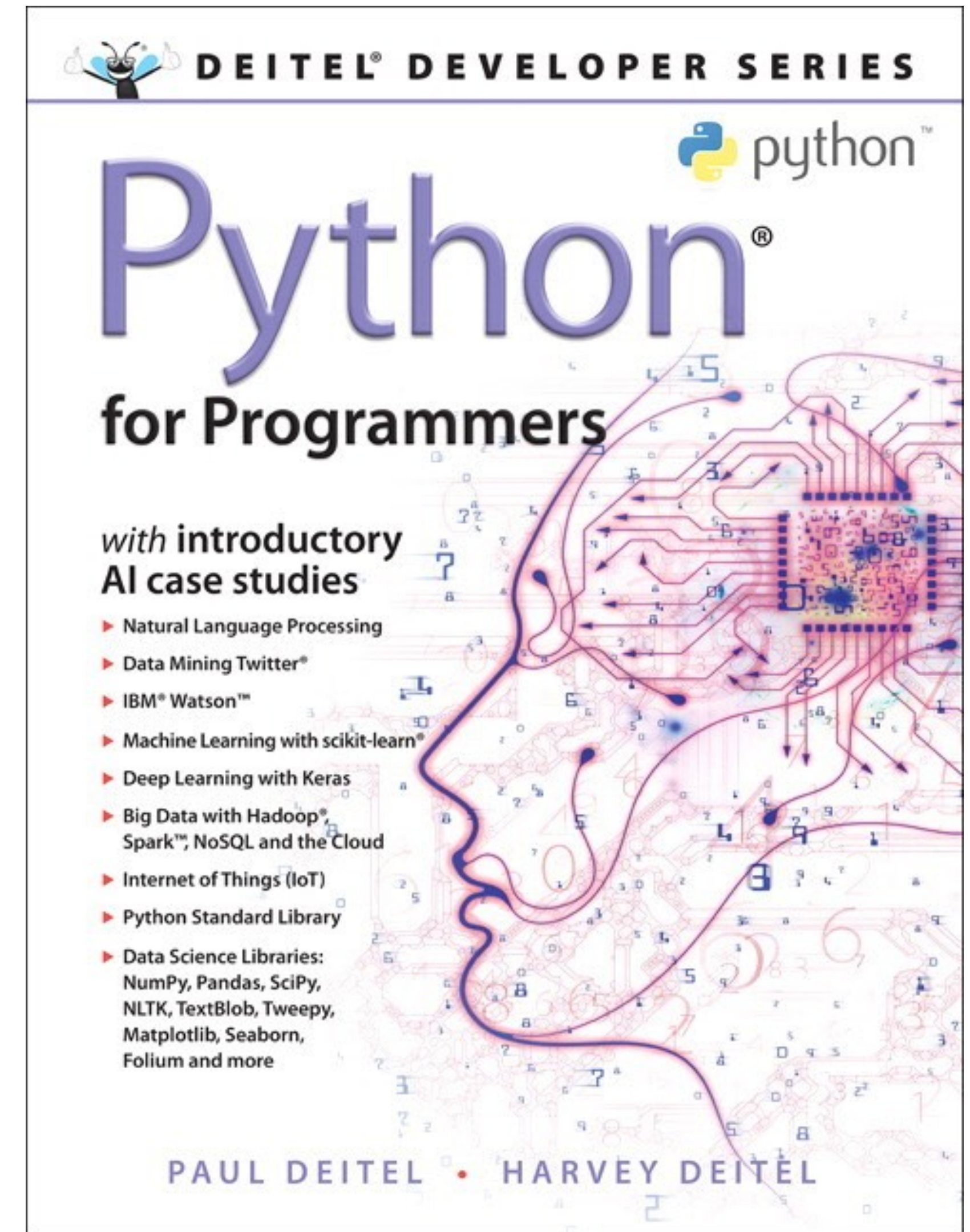
Schedule

- Lectures are 12:30-1:45pm MW in PM 110
  - Better for learning if you are engaged
  - **Ask questions**
  - Please advise me of any issues
- Any changes will be announced as soon as possible
- Slides will be posted to the course website

Office Hours & Email

- TA office hours will be held in person in TA Offices

  - **Tentative**: M 2-5pm, Tu 9:30am-12:30pm, W 1:00-4:00pm, Th 9:30am-12:30pm

- Prof. Koop's office hours will be held in person in PM 461

  - M: 1:45-3:00pm, W: 10:45am-12:00pm, or by appointment

  - You do not need an appointment to stop by during scheduled office hours,

  - If you wish to meet virtually, please schedule an appointment

  - If you need an appointment, please email me with **details** about what you wish to discuss and times that would work for you

- Many questions can be answered via email. **Please consider writing an email before scheduling a meeting.**

Course Material

- Textbook:
  - Recommended: Python for Programmers
  - Good overview + data science examples
- Many other resources are available:
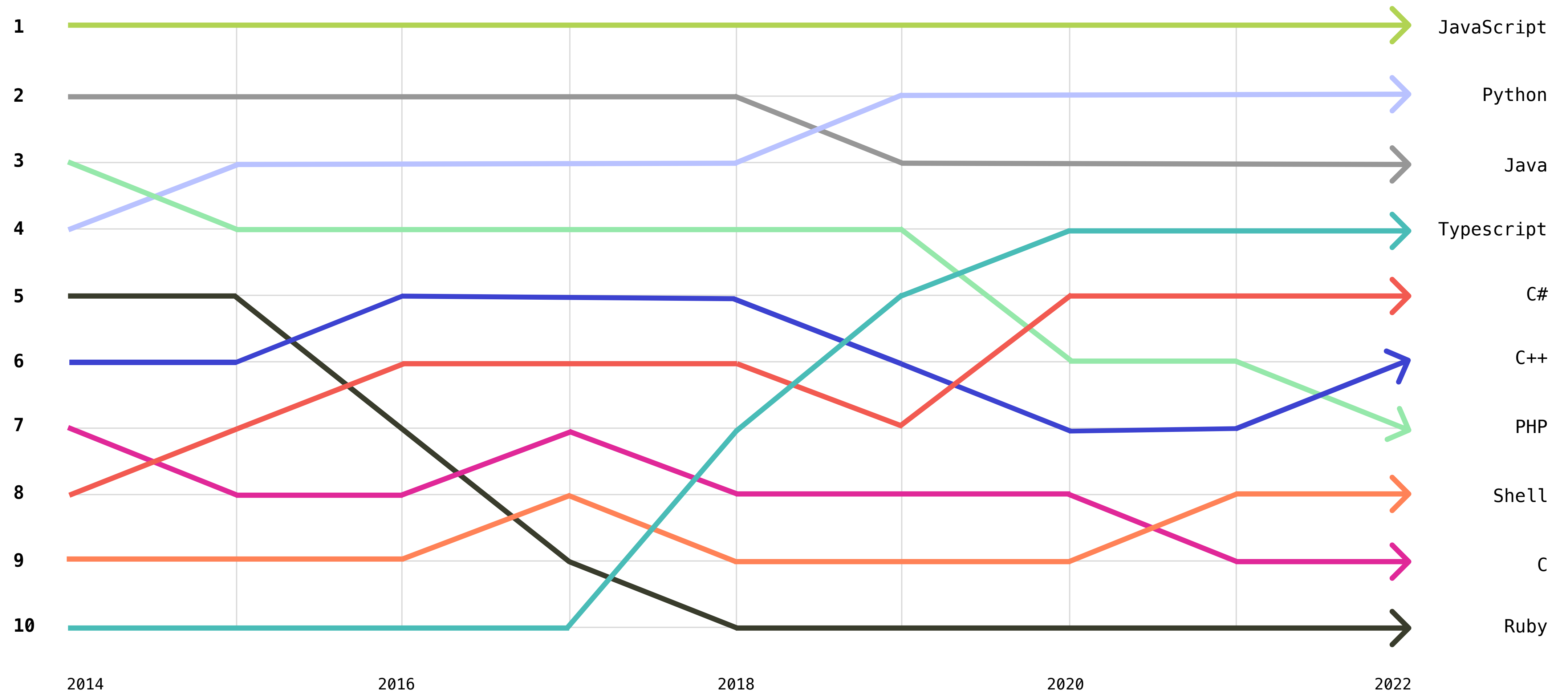 - https://wiki.python.org/moin/BeginnersGuide
  - https://wiki.python.org/moin/IntroductoryBooks
  - http://www.pythontutor.com
  - https://www.python-course.eu
  - https://software-carpentry.org/lessons/

DEITEL® DEVELOPER SERIES

python

Python®

for Programmers

with **introductory AI case studies**

- ▶ Natural Language Processing
- ▶ Data Mining Twitter®
- ▶ IBM® Watson™
- ▶ Machine Learning with scikit-learn®
- ▶ Deep Learning with Keras
- ▶ Big Data with Hadoop®, Spark™, NoSQL and the Cloud
- ▶ Internet of Things (IoT)
- ▶ Python Standard Library
- ▶ Data Science Libraries: NumPy, Pandas, SciPy, NLTK, TextBlob, Tweepy, Matplotlib, Seaborn, Folium and more

PAUL DEITEL • HARVEY DEITEL

Northern Illinois University

Course Material

- Software:
  - Anaconda Python Distribution (https://www.anaconda.com/download): makes installing python packages easier
  - Jupyter Notebook: Web-based interface for interactively writing & executing Python code
  - JupyterLab: An updated web-based interface that includes the notebook and other cool features
  - JupyterHub: Access everything through a server

Python

- Started in December 1989 by Guido van Rossum
- "Python has surpassed Java as the top language used to introduce U.S. students to programming…" (ComputerWorld, 2014)
- Python is also a top language for data science
- High-level, interpreted language
- Supports multiple paradigms (OOP, procedural, functional)
- Help programmers write **readable** code, use less code to do more
- Lots of libraries for python
- Designed to be extensible, easy to wrap code from other languages like C/C++
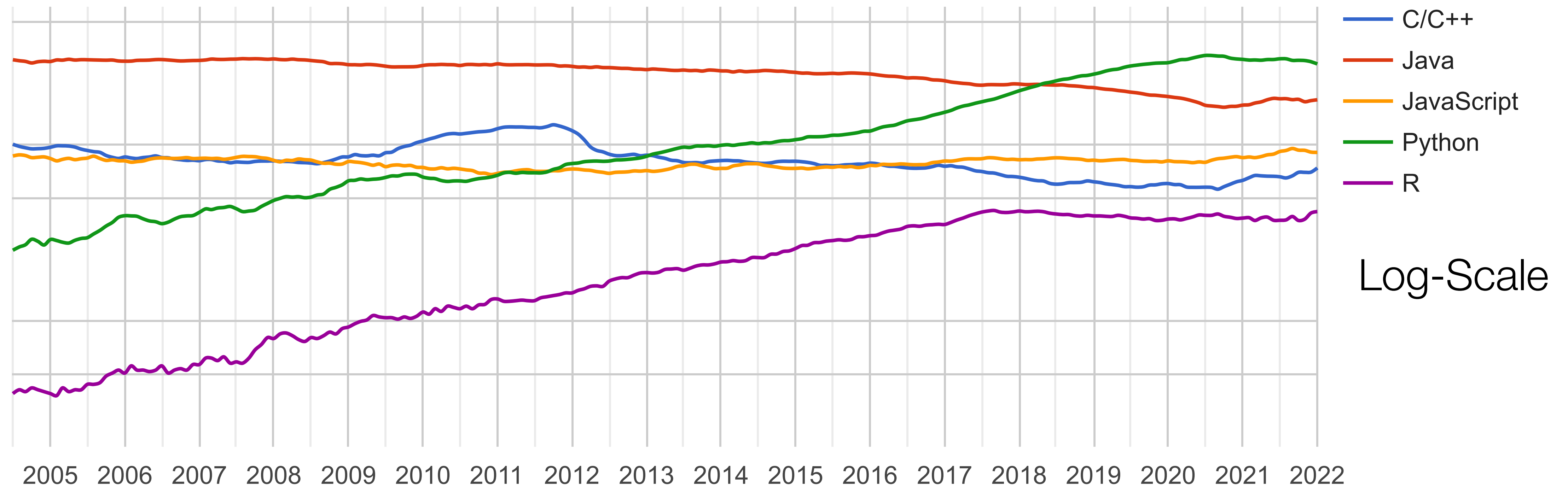- Open-source with a large, passionate community

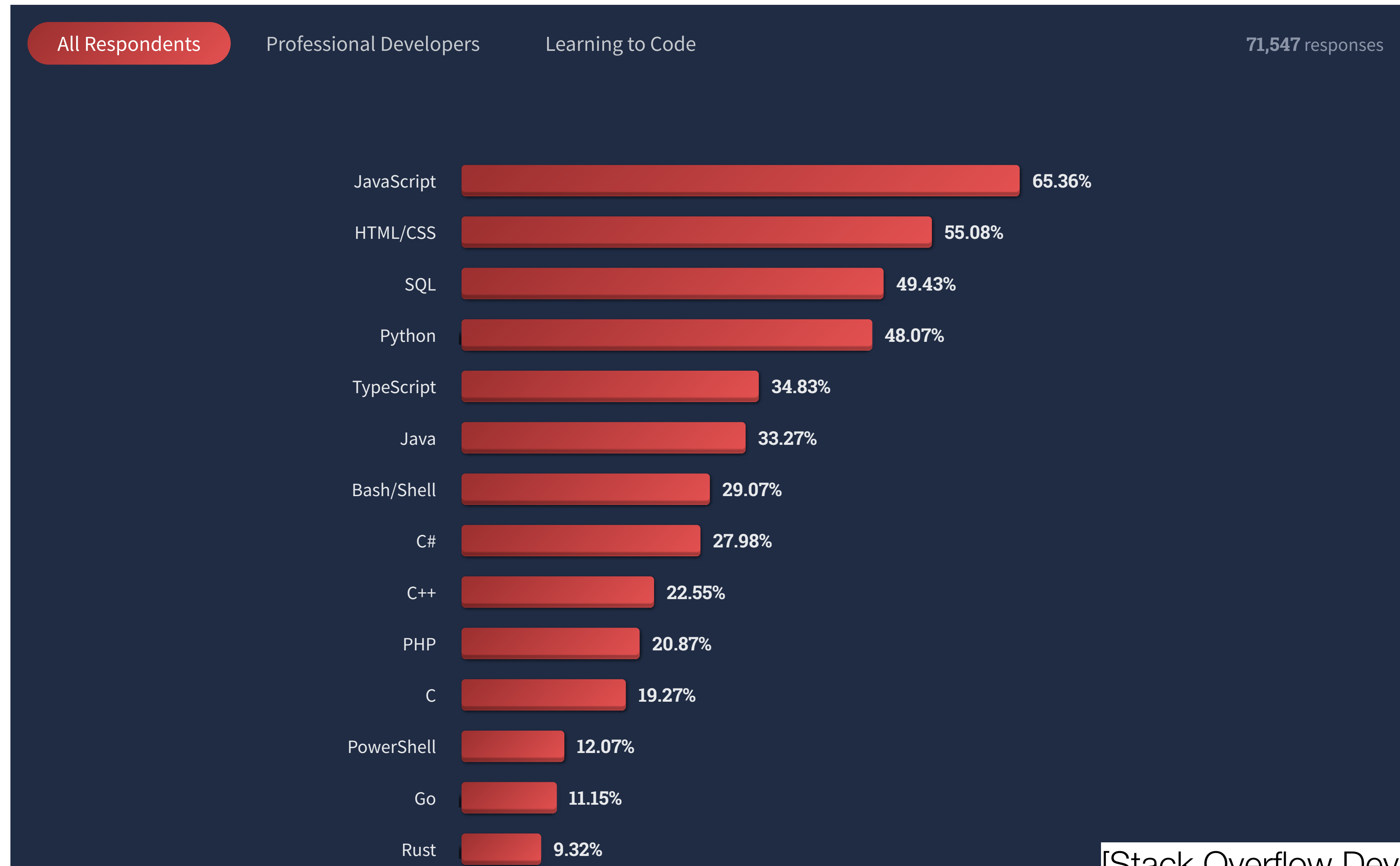# Python the #2 Programming Language in 2022

[GitHub]

Even Wider Gap in Google Tutorial Searches
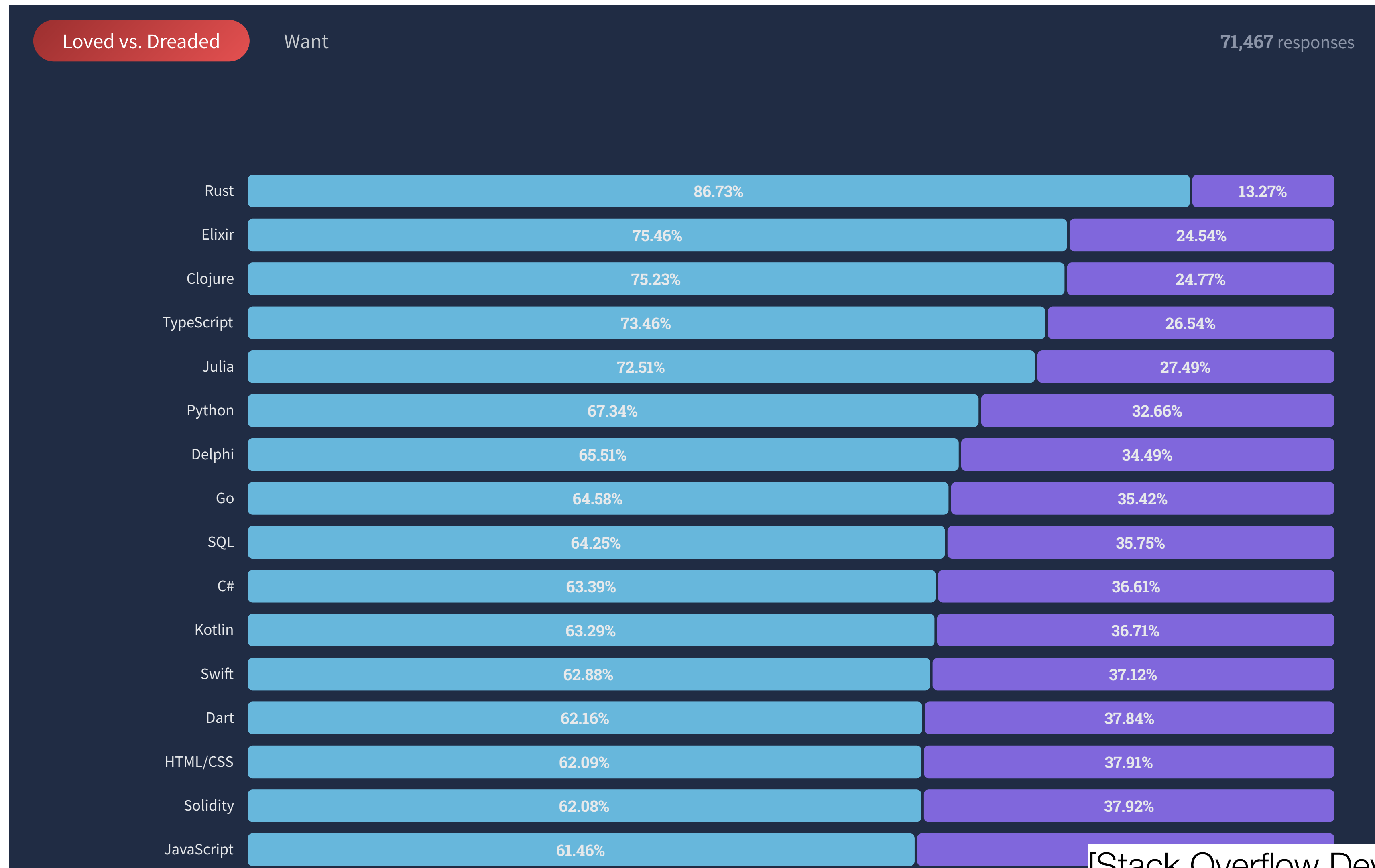
PYPL PopularitY of Programming Language

Log-Scale

Legend: C/C++, Java, JavaScript, Python, R

X-axis: 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022

[PopularitY of Programming Language]

StackOverflow Language Usage

All Respondents Professional Developers Learning to Code   71,547 responses

Language	Percentage
JavaScript	65.36%
HTML/CSS	55.08%
SQL	49.43%
Python	48.07%
TypeScript	34.83%
Java	33.27%
Bash/Shell	29.07%
C#	27.98%
C++	22.55%
PHP	20.87%
C	19.27%
PowerShell	12.07%
Go	11.15%
Rust	9.32%

[Stack Overflow Developer Survey, 2022]

StackOverflow Language Preferences

Rust is on its seventh year as the most loved language with 87% of developers saying they want to continue using it.

Rust also ties with Python as the most wanted technology with TypeScript running a close second.

	Loved vs. Dreaded	Want	71,467 responses

Language	Loved	Dreaded
Rust	86.73%	13.27%
Elixir	75.46%	24.54%
Clojure	75.23%	24.77%
TypeScript	73.46%	26.54%
Julia	72.51%	27.49%
Python	67.34%	32.66%
Delphi	65.51%	34.49%
Go	64.58%	35.42%
SQL	64.25%	35.75%
C#	63.39%	36.61%
Kotlin	63.29%	36.71%
Swift	62.88%	37.12%
Dart	62.16%	37.84%
HTML/CSS	62.09%	37.91%
Solidity	62.08%	37.92%
JavaScript	61.46%	

[Stack Overflow Developer Survey, 2022]

D. Koop, CSCI 503/490, Spring 2024

Northern Illinois University

18

StackOverflow Language Interest

Rust is on its seventh year as the most loved language with 87% of developers saying they want to continue using it.

Rust also ties with Python as the most wanted technology with TypeScript running a close second.

Loved vs. Dreaded   **Want**

71,467 responses
% of developers who are not developing with the language or technology but have expressed interest in developing with it

Language	Percent
Rust	**17.6%**
Python	**17.59%**
TypeScript	**17.03%**
Go	**16.41%**
JavaScript	**12.98%**
Kotlin	**8.08%**
C++	**7.67%**
SQL	**6.51%**
C#	**6.11%**
Java	**5.6%**
Dart	**4.83%**
C	**4.34%**
Swift	**4.3%**

[Stack Overflow Developer Survey, 2022]

Modes of Computation

- Python is **interpreted**: you can run one line at a line without compiling
- Interpreter in the Shell
  - Execute line by line
  - Hard to structure loops
  - Usually execute whole files (called scripts) and edit those files
- Notebook
  - Richer results (e.g. images, tables)
  - Can more easily edit past code
  - Re-execute any cell, whenever

Python Differences

- Dynamic Typing

  - A variable does not have a fixed type

  - Example: `a = 1; a = "abc"`

- Indentation

  - Braces define blocks in Java, good style is to indent but not required

  - Indentation is critical in Python

```
z = 20
if x > 0:
    if y > 0:
        z = 100
else:
    z = 10
```

JupyterLab and Jupyter Notebooks

Jupyter Notebooks

- Display rich representations and text

- Uses Web technology

- Cell-based

- Built-in editor

- GitHub displays notebooks
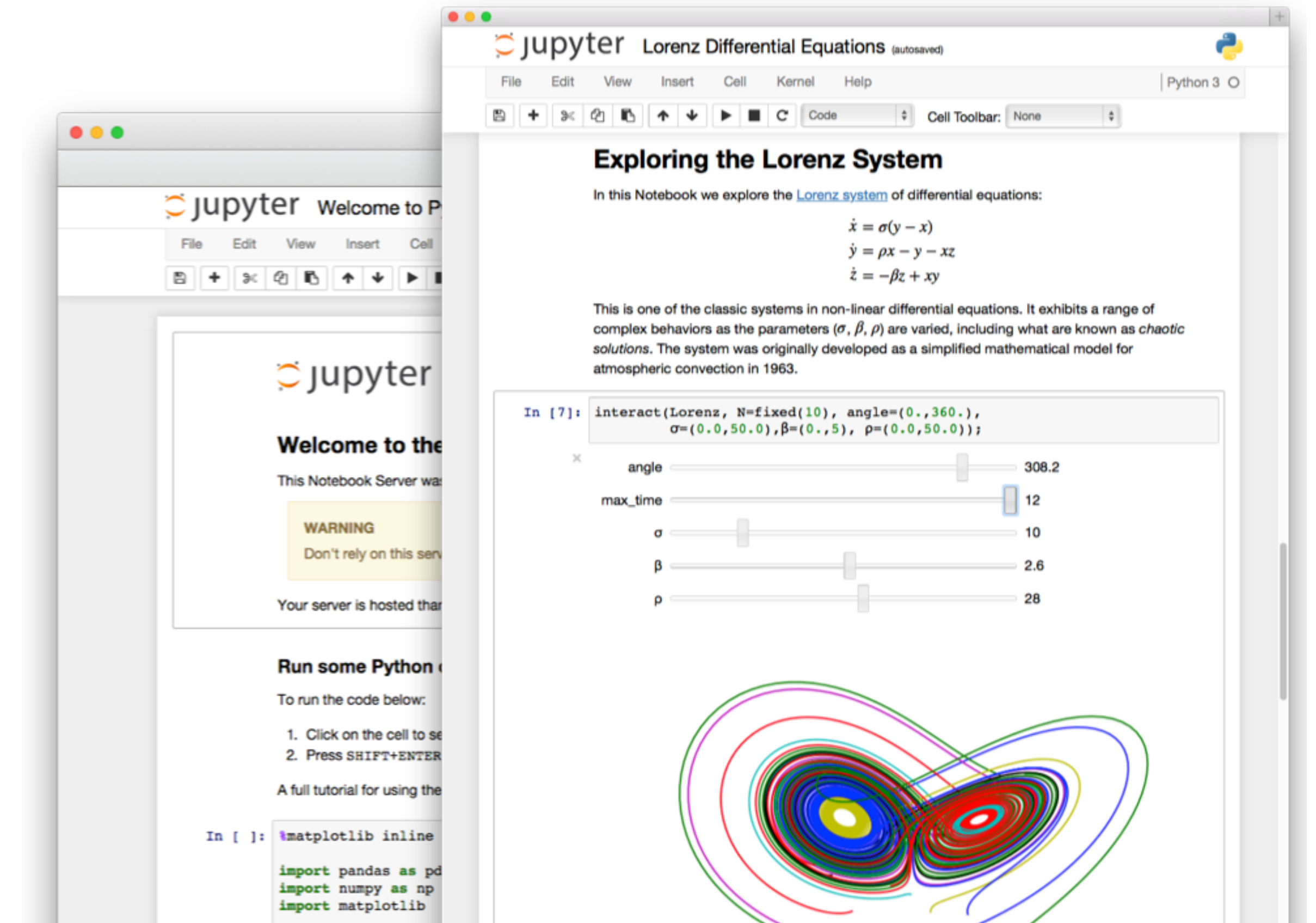
Jupyter Notebooks

- An interactive programming environment
- Runs in your web browser
- Displays results (even interactive maps) inline
- Originally designed for Python
- Supports other languages, too
- You decide how to divide code into executable cells
- Shift+Enter (or the "play" button) to execute a cell

Notebooks in JupyterLab

- Directory view on left

- Create new notebooks using "+" button, "New" from the File menu, or Launcher window

  - Notebook originally has name "Untitled"

  - Click on "Untitled" to change the name (do this!)

- Save a notebook using the command under the File menu

- Shutting down the notebook — use Close and Shutdown Kernel

  - Web browser is **interface** to display code and results

  - **Kernel** actually runs the code: usually see messages in a console/terminal window

Notebooks in JupyterLab

- Open a notebook by going back to the file browser and clicking on it like you would in a desktop view

- Past results are displayed—does not mean they are loaded in memory

- Use "Run All" or "Run All Above" to re-execute past work

  - If you shut down the kernel, all of the data and variables you defined need to be redefined (so you need to re-run all)

  - **Watch Out—Order Matters**: If you went back and re-executed cells in a different order than they are shown, doing "Run All" may not produce the same results!

- Edit mode (green) versus Command mode (blue == **Be Careful**)

- Learn keyboard shortcuts

Notebooks in JupyterLab

- Can write code or plain text (can be styled Markdown)

  - Choose the type of cell using the dropdown menu

- Cells break up your code, but all data is **global**

  - Defining a variable `a` in one cell means that variable is accessible in **any** other cell

  - This includes cells **above** the cell `a` was defined in!

- Remember **Shift+Enter** to execute

- Enter just adds a new line

- Use ?<function_name> for help

- Use Tab for **auto-complete** or suggestions

JupyterLab

- More than just notebooks:
  - Text editor
  - Console
  - Custom components (Many extensions)
- Arrange multiple documents and views
- <u>JupyterLab Documentation</u>

Using Python & JupyterLab Locally

- www.anaconda.com/download/

- Anaconda has JupyterLab

- Use Python 3.12

- Anaconda Navigator

  - GUI application for managing Python environment

  - Can install packages

  - Can start JupyterLab

- Can also use the shell to do this:

  - `$ jupyter lab`

  - `$ conda install <pkg_name>`

Using Python & JupyterLab on Course Server

- Stay tuned…

Chicago Food Inspections

- Data: Information about food facility inspections in Chicago

- Data Source: https://data.cityofchicago.org/Health-Human-Services/Food-Inspections/4ijn-s7e5/data

- Fields: Name, Facility Type, Risk, Violations, Location, etc.

Chicago Food Inspections Exploration

- Based on David Beazley's PyData Chicago talk

- YouTube video: https://www.youtube.com/watch?v=j6VSAsKAj98

- Our in-class exploration:

  - Don't focus on the syntax

  - Focus on how interactive Python makes this exploration work well