Python for Astronomy

ASTR 302, Winter quarter 2026

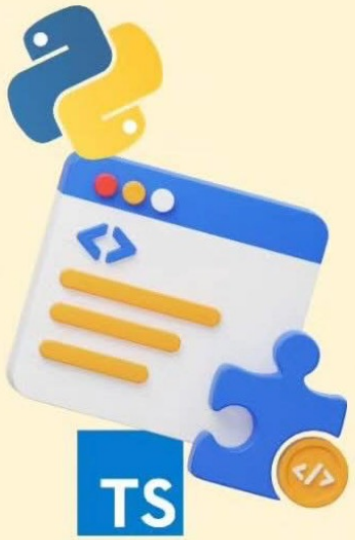
ivezic@uw.edu

Today

- Why Python for Astronomers?
- Administrivia
- Getting set up (JupyterHub & Slack)

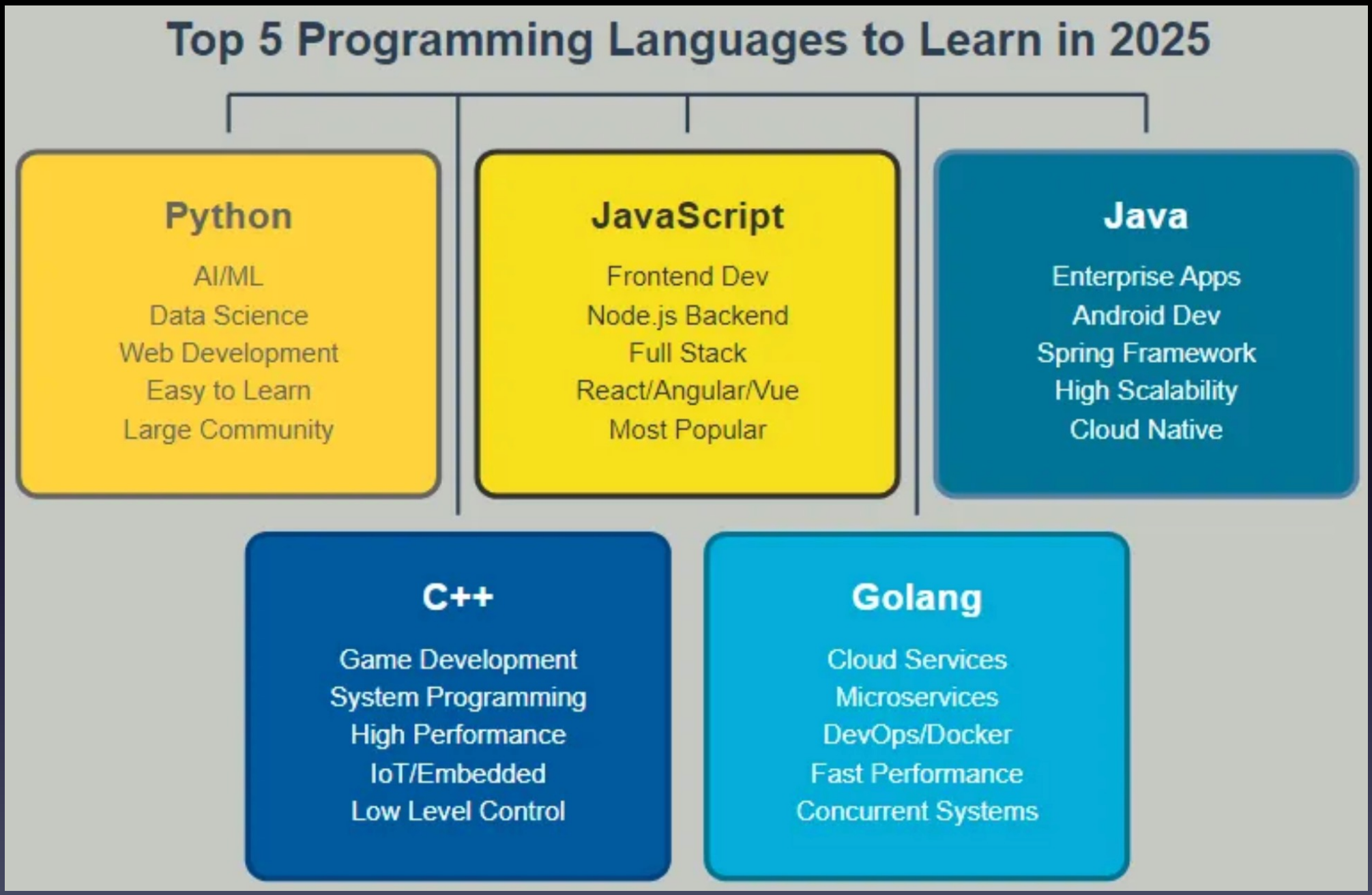
Most Popular Programming Languages of 2025 !

Python	30.9%
Java	25.1%
JavaScript	15.2%
C#	7.2%
PHP	6.2%
C	5.9%
R	3.7%
objective c	2.4%
Swift	2.3%
Type Script	1.1%



Similar but optimized differently...

Top 5 Programming Languages to Learn in 2025



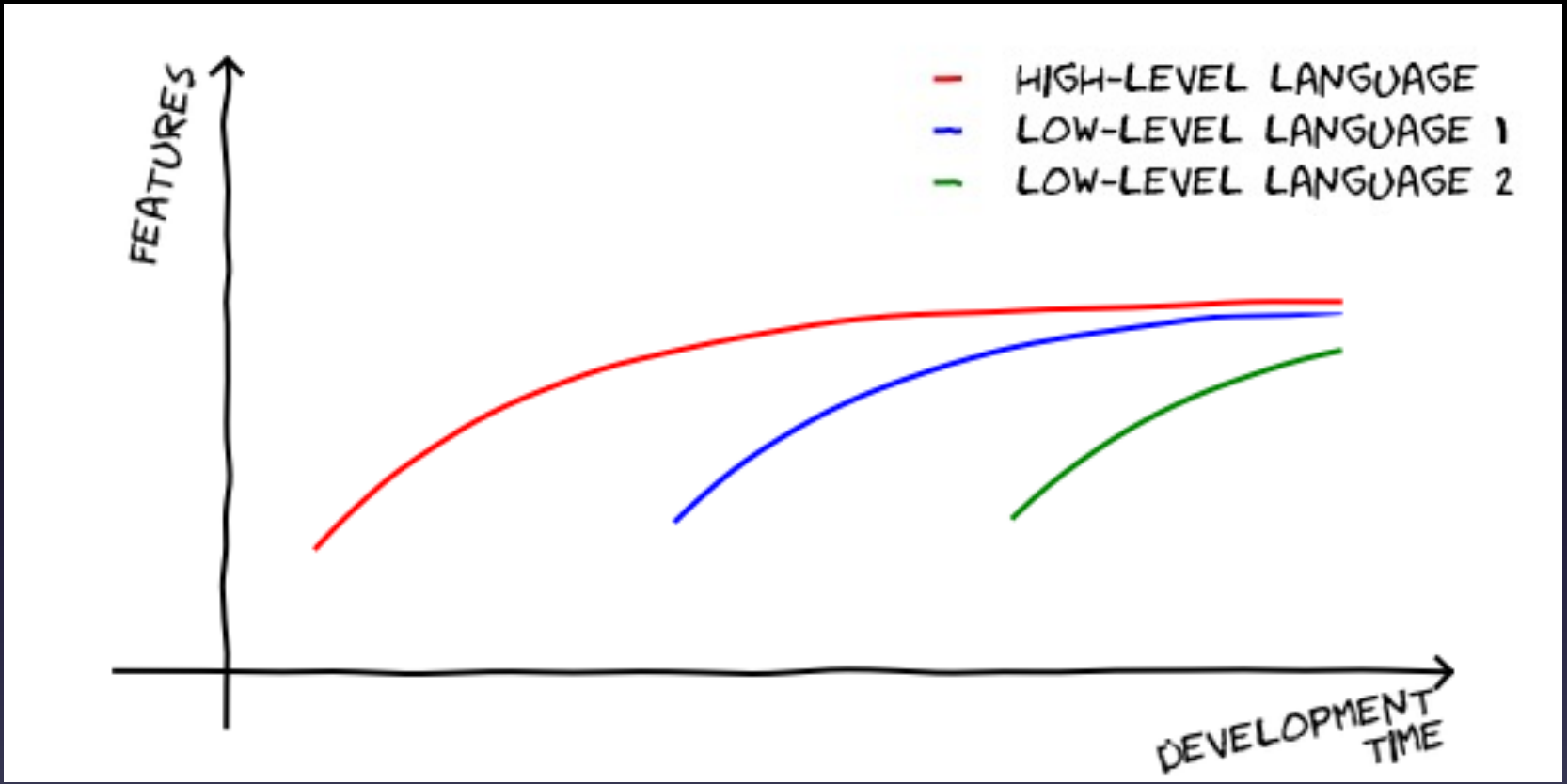
Why Python for Astronomers?

- ***Python has recently become the common language of astronomy (specifically, astronomical data analysis).***

- The goal of this course is to teach you how to effectively use the tools available in the Python ecosystem (the language, the libraries, and *the concepts underpinning both*) for your research. Think of it as a part of a series: ASTR 300 -> 302 -> 324 -> 497 (Astro Surveys & Big Data).

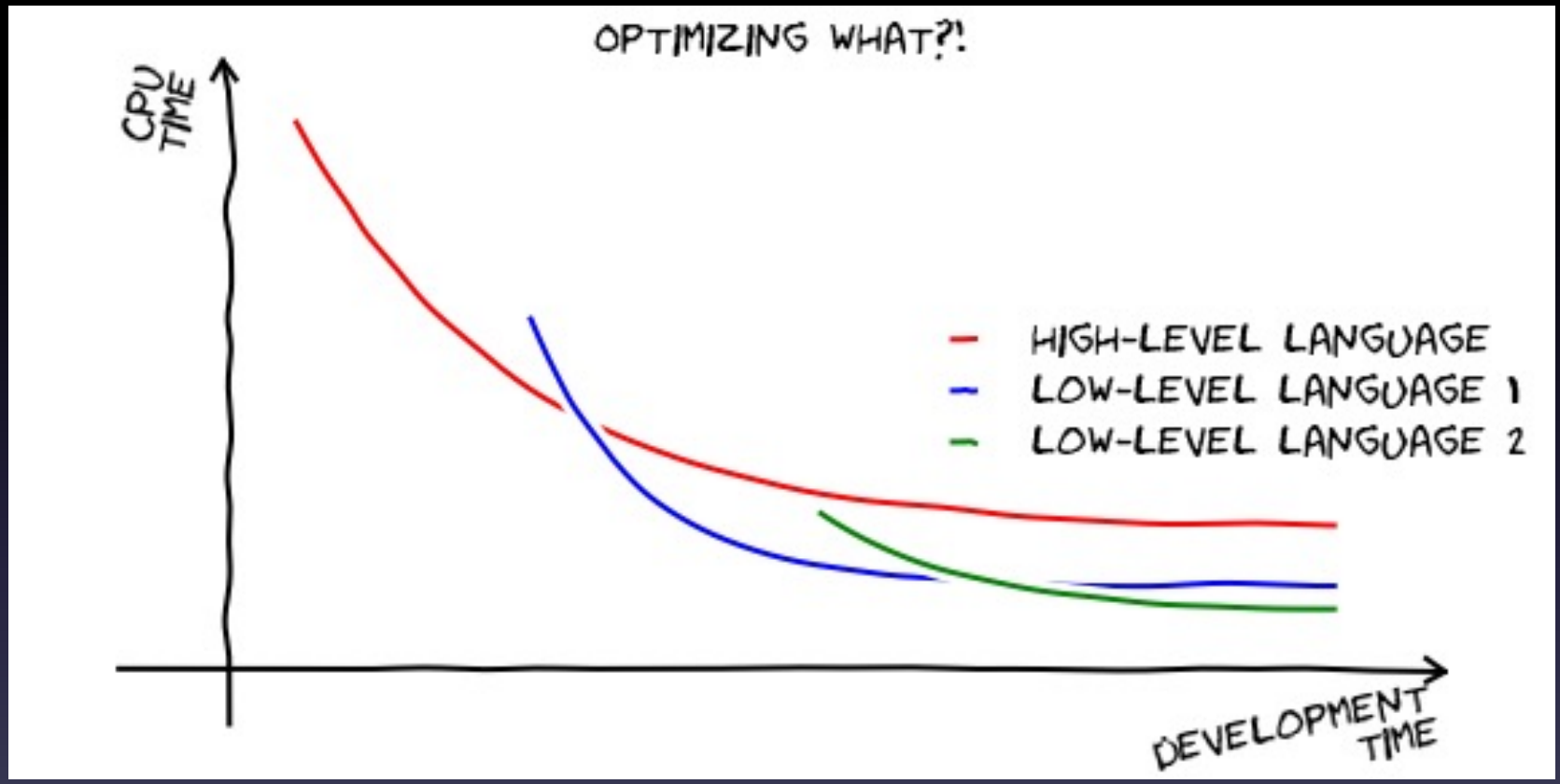
- While motivated by astronomy, the skills learned here should be broadly useful. It should easily transfer to more general data-science work.

Why Python?



Higher-level languages, in general, shorten your time to research results. Python is one such language that is easy to learn, freely available, similar to legacy languages (IDL, FORTRAN, C/C++), and was mature enough at just the right time to spur adoption.

As usual, lunches don't come free



Computational efficiency is the price you pay for speed of development. But that is typically a good trade to make.

From

https://www.squareboat.com/blog/advantages-and-disadvantages-of-python

Advantages

- Comprehensive Libraries and Frameworks
- Ease of learning and use
- Highly Embeddable
- Automatic memory mgmt
- Open source and large community

Disdvantages

- Slow Execution Speed
- Dynamically typed
- High Memory Consumption
- Threading Limitations
- Database Interaction Challenges

How Many of you are comfortable with…

Writing a command-line Python script (.py)

Using Python lists()

Using Python dicts()

Using tuples

Writing Python functions

Writing Python classes

Writing a Python module

Working with numpy arrays

Making a mpl scatter plot

Plotting an image w. mpl

Exception handling

Writing a list comprehension

Writing a lambda function

Using decorators

Writing a decorator

Utilizing ABC classes

"Hey, look, a survey!"

"The Only Thing That Is Constant Is Change"

- typically attributed to Heraclitus

- In astronomical data analysis (and related computing disciplines), things change too quickly to be learned only once. Learning is a process that never ends.
- I will try to teach you what currently appears to be the best set of tools and techniques to have in your toolbox.
- But know this will change. So the major emphasis of this course will be on *understanding the concepts*, and learning how to continuously keep your knowledge up-to-date.

Languages may change,
but concepts transfer!

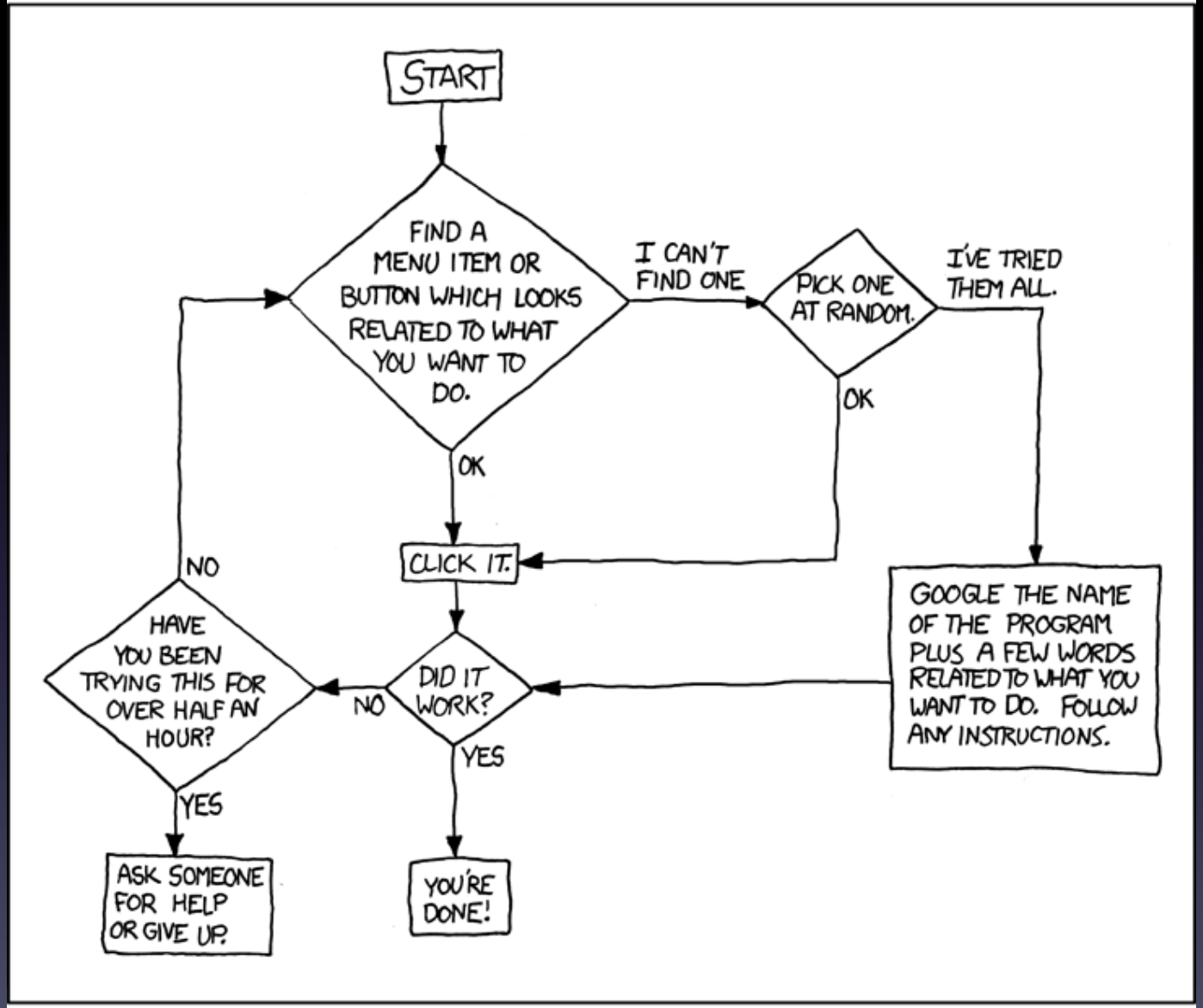
Learning how to Learn

- We'll also learn ***how*** to learn: how to discover information, keep your knowledge up-to-date, find leads that help you solve problems, evaluate their worthiness.

- This may be the most valuable piece to take away from this class.

DEAR VARIOUS PARENTS, GRANDPARENTS, CO-WORKERS, AND OTHER "NOT COMPUTER PEOPLE."

WE DON'T MAGICALLY KNOW HOW TO DO EVERYTHING IN EVERY PROGRAM. WHEN WE HELP YOU, WE'RE USUALLY JUST DOING THIS:



PLEASE PRINT THIS FLOWCHART OUT AND TAPE IT NEAR YOUR SCREEN. CONGRATULATIONS; YOU'RE NOW THE LOCAL COMPUTER EXPERT!

Class Projects

- January: a short (re)introduction to the basics (Python, git, how to find information, teamwork & software dev. methodology).

- Form a few "startups", each with an (ambitious) product in mind.
  - 4-5 people each. Work in teams, using github and agile sprints.
  - Example: Build a visualization widget for running some code
  - Example: Build a website to tell whether Rubin could find a newly discovered NEO
  - project pitch day: February 3
- Feb-March: work towards that goal in weekly sprints

Lectures

- When: Tue/Thu,10:00am-11:20am
- Were & how: PAA 216
  - Please bring your laptops

- Lecture structure (may change from week to week):
  - About an ~hour for introducing new material. **!!! WILL BE HANDS-ON !!!**
  - Leave ~20 minutes to work on homeworks, questions, open-ended discussion
  - Interrupt and ask questions at any time!

Course Materials

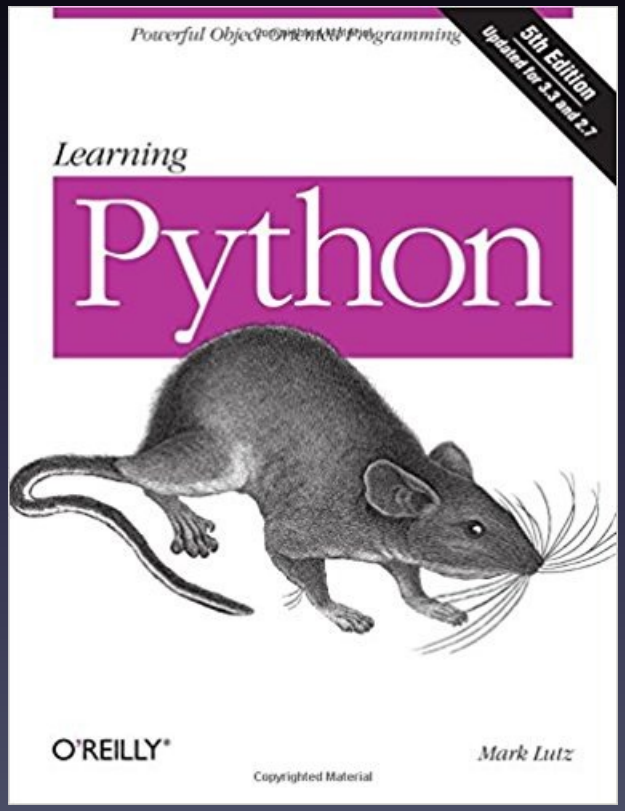
- I'll be adding most of what we need to the following repository on GitHub:

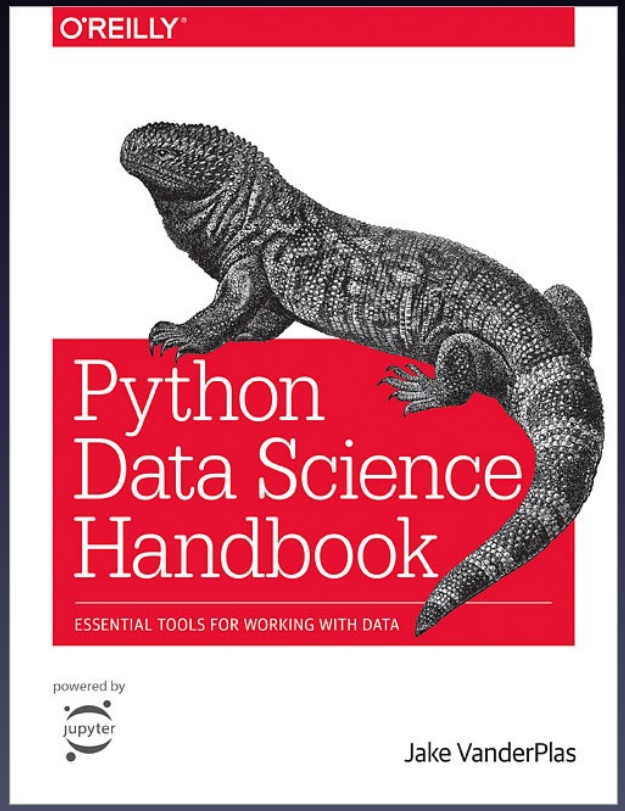
https://github.com/uw-astr-302/astr-302-w26

Action: Let's make sure everyone can access this repository.

Textbooks & Reference Material

- There is a wealth of material available online and as eBooks (for UW students, via UW Libraries & Seattle Public Library). The two I'd recommend to begin with:





+ many (many!) online writeups / blog posts that I will point you to over the next few weeks.

Syllabus

- Available at: https://github.com/uw-astr-302/astr-302-w26/blob/main/syllabus/astr-302-syllabus.pdf

  (this will take you to the class github repository).

- Things to note:
  - The syllabus *will* change as we go through the quarter.

    Your feedback will be invaluable!

Grading, Homeworks, Etc.

Homeworks (30% of the grade):

- Designed to exercise what we've learned recently.

Projects (70% of the grade):

- The projects you will build.

Advice: Turn in your homeworks on time!

Communication: Slack

- In this class, we won't be using a mailing list but an instant messaging (-like) tool called Slack (http://slack.com). Slack is heavily used today by many research & technology companies and projects.
- Request to join our department Slack at:
  - `https://uw-astronomy.slack.com`
  - `Then join the #astr-302 channel`
- What to use it for:
  - Asking questions, discussing the class, exchanging snippets of code, collaborating on projects (when appropriate).
  - Please prefer Slack to sending me e-mails. Two reasons:
    - Everyone can benefit from the question and answer.
    - Your colleagues may be able to help!

Action: Let's make sure everyone's on Slack.

Our working environment: class JupyterHub

https://dirac.us/hub302

- A UW-managed Jupyter
- Allows you to run Jupyter on a remote computer w/o having to have anything installed locally (like Google Docs, but for Jupyter)
- Everything "Just works" ™

Action: Let's try it out

Getting Python: Miniconda

- Locally, you can install the Miniconda Python Distribution.

https://conda.io/miniconda.html

Note: we don't need it for this class, but it'll be useful for you to learn how to set up a Python environment from scratch.

a) it's easy, and b) you won't always have sysadmins available to do it for you.

Next time

- A Walk Through Python!

Note: Please refresh your ASTR 300 memory by working through the notebooks at

https://github.com/UWashington-Astro300

(the ASTR 300 github repository)