CMPU 100

Programming with Data

Fall 2025

Hello, computer

Hello, computer
I QUIT SMOKING

We use computers every day as electronic *black boxes* that do amazing things by

collecting,

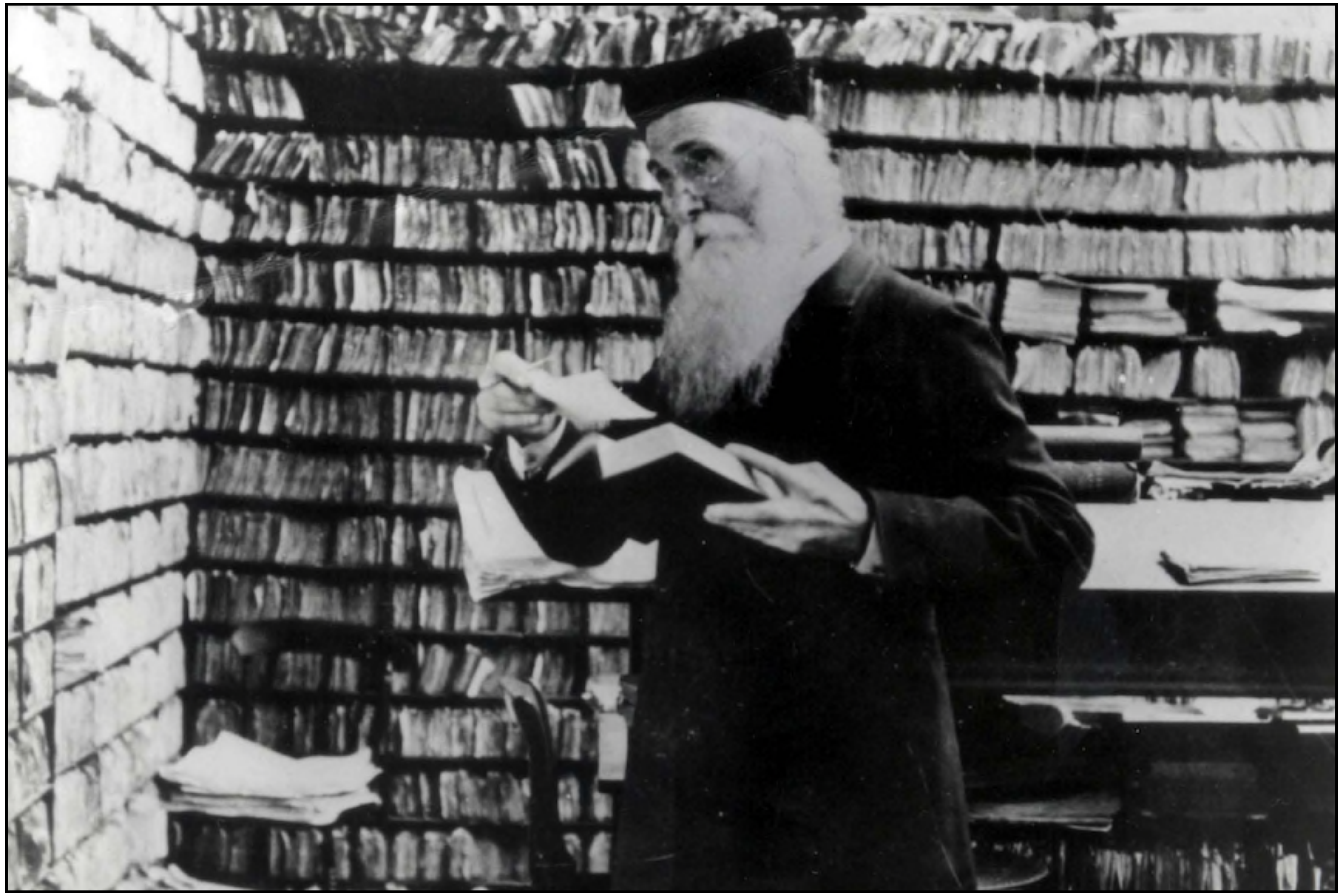
storing,

retrieving, and

transforming *data*.

“Many people think of data as numbers alone, but data can also consist of words or stories, colors or sounds, or any type of information that is systematically collected, organized, and analyzed…”

D’Ignazio & Klein, *Data Feminism*, 2020



James Murray compiling the *Oxford English Dictionary*, c. 1928.

Computers only do very basic things.

Numerical calculations:

Add

Subtract

…

Symbolic manipulations

Compare two numbers

Substitute one string of letters and numbers for another

…

But when trillions of these simple operations are arranged in the right order, amazing computations can be carried out:

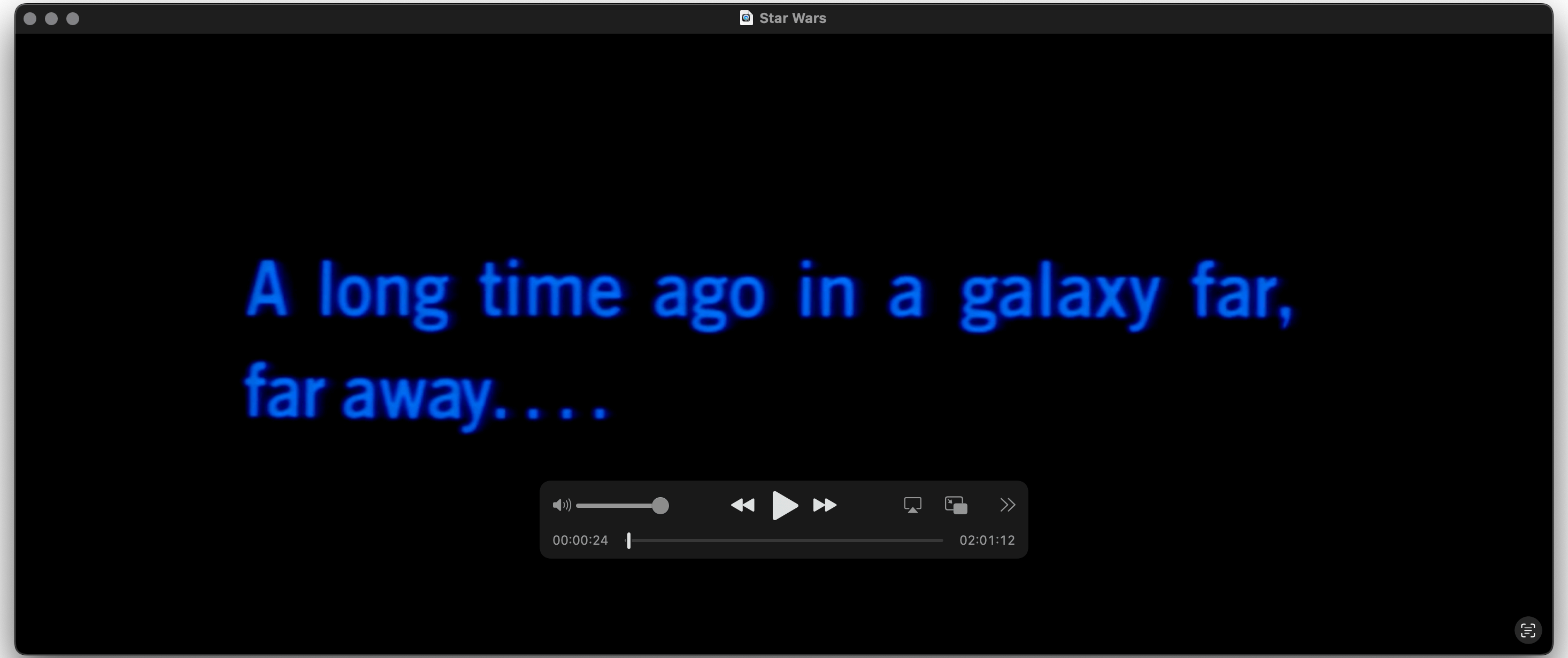
forecasting tomorrow's weather 🌨️

deciding where to drill for oil 🛢️

finding which places a person's most likely to visit 🚗

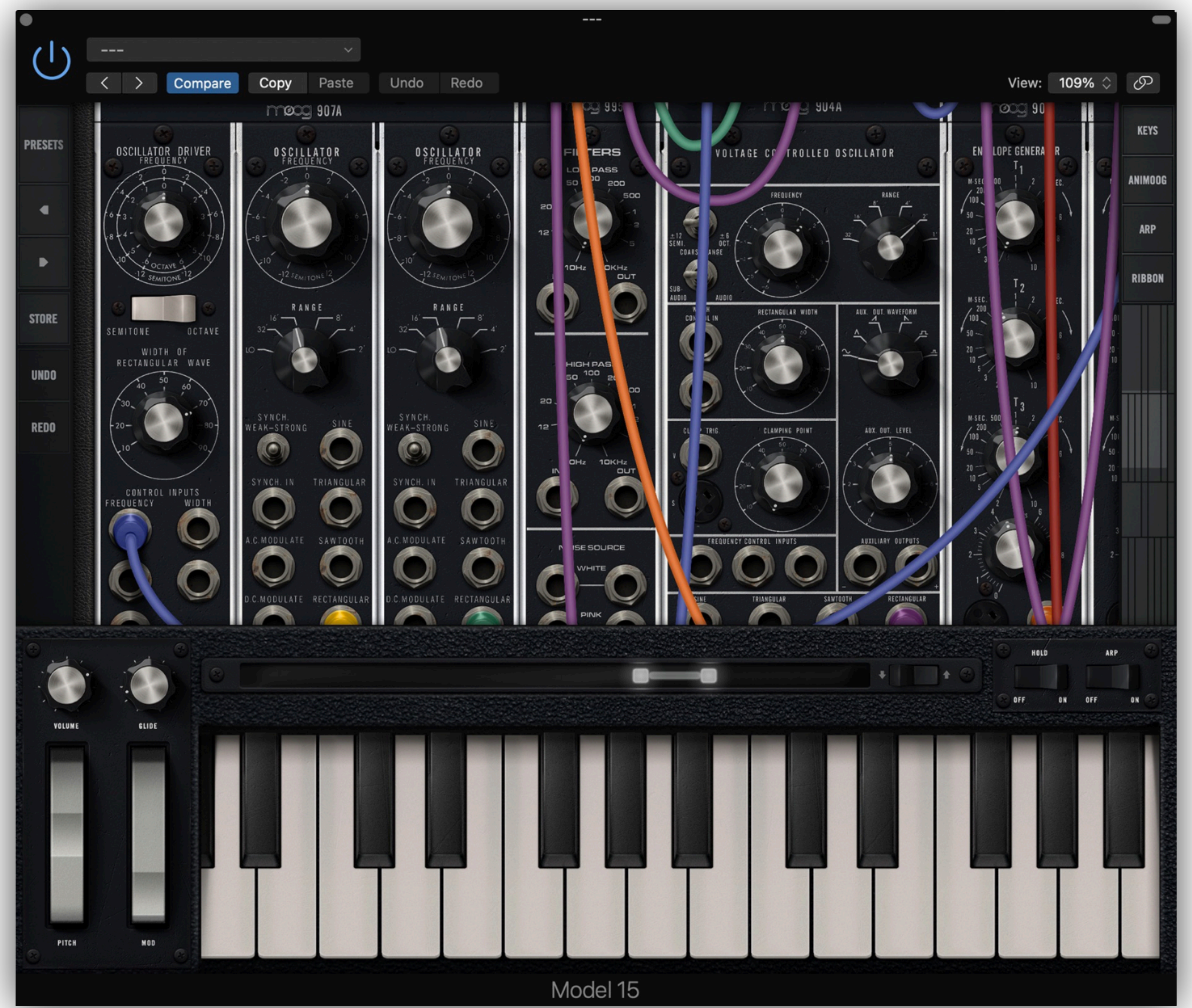
figuring out who would make a great couple 😘😍

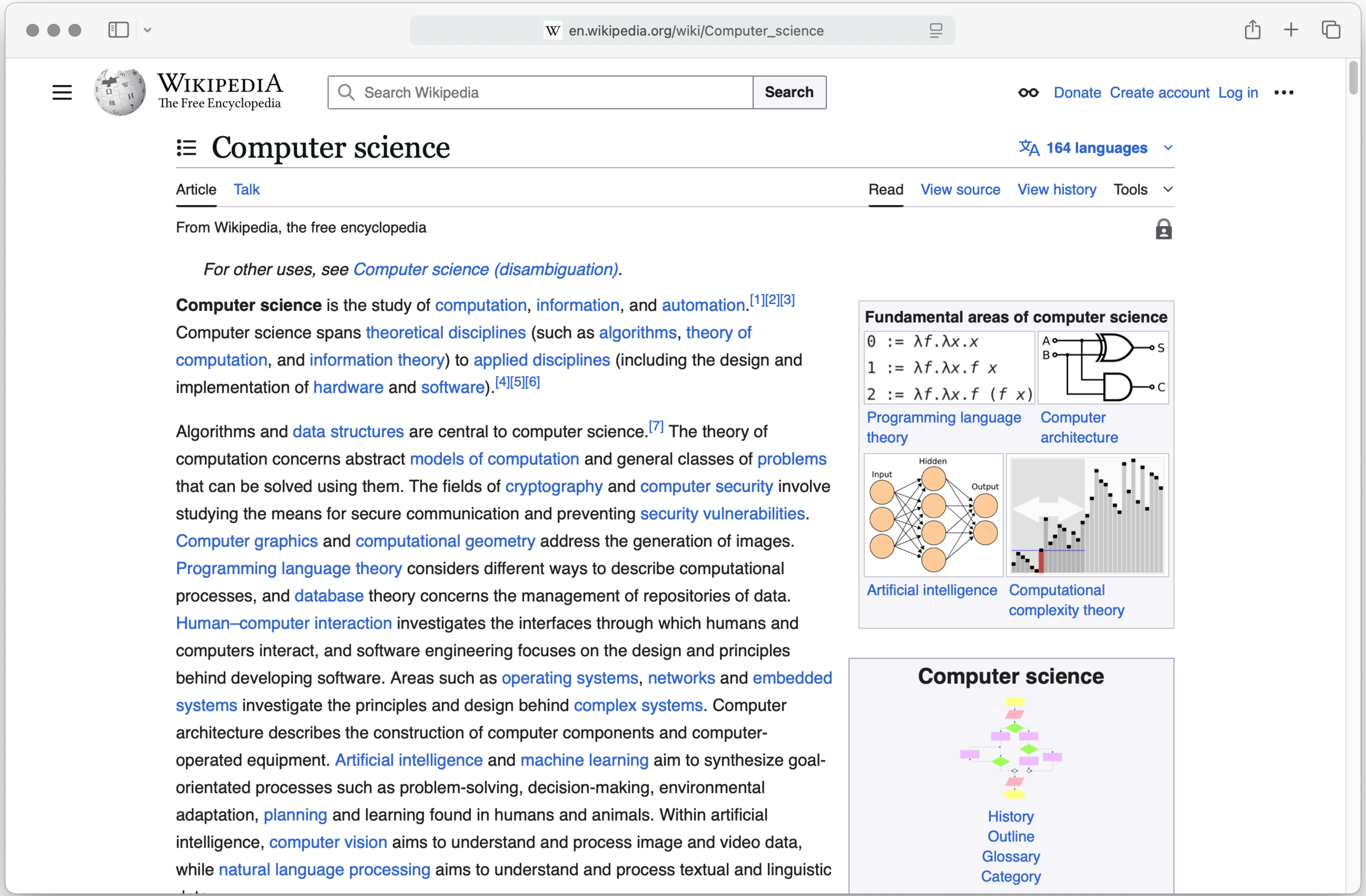
…



Star Wars

A long time ago in a galaxy far,
far away. . . .


Compare
Copy
Paste
Undo
Redo
View: 109%
PRESETS
STORE
UNDO
REDO
KEYS
ANIMOOG
ARP
RIBBON
OSCILLATOR DRIVER
OSCILLATOR
VOLTAGE CONTROLLED OSCILLATOR
FILTERS
VOLUME
GLIDE
PITCH
MOD
HOLD
ARP
Model 15



Computer science

From Wikipedia, the free encyclopedia

For other uses, see Computer science (disambiguation).

Computer science is the study of computation, information, and automation.[1][2][3] Computer science spans theoretical disciplines (such as algorithms, theory of computation, and information theory) to applied disciplines (including the design and implementation of hardware and software).[4][5][6]

Algorithms and data structures are central to computer science.[7] The theory of computation concerns abstract models of computation and general classes of problems that can be solved using them. The fields of cryptography and computer security involve studying the means for secure communication and preventing security vulnerabilities. Computer graphics and computational geometry address the generation of images. Programming language theory considers different ways to describe computational processes, and database theory concerns the management of repositories of data. Human–computer interaction investigates the interfaces through which humans and computers interact, and software engineering focuses on the design and principles behind developing software. Areas such as operating systems, networks and embedded systems investigate the principles and design behind complex systems. Computer architecture describes the construction of computer components and computer-operated equipment. Artificial intelligence and machine learning aim to synthesize goal-orientated processes such as problem-solving, decision-making, environmental adaptation, planning and learning found in humans and animals. Within artificial intelligence, computer vision aims to understand and process image and video data, while natural language processing aims to understand and process textual and linguistic

Fundamental areas of computer science

$0 := \lambda f.\lambda x.x$

$1 := \lambda f.\lambda x.f\ x$

$2 := \lambda f.\lambda x.f\ (f\ x)$

Programming language theory

Computer architecture

Artificial intelligence

Computational complexity theory

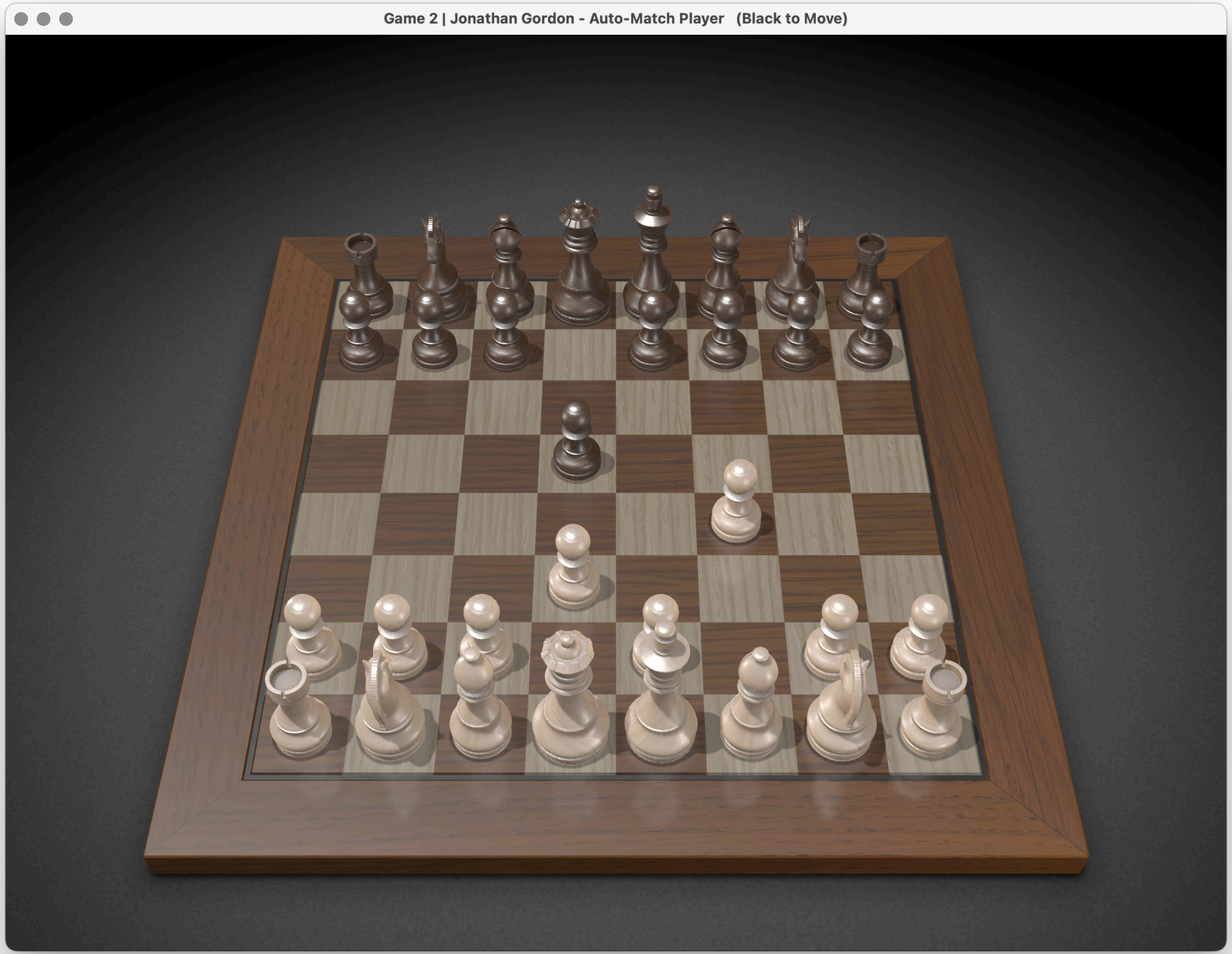
Computer science

History

Outline

Glossary

Category


Game 2 | Jonathan Gordon - Auto-Match Player (Black to Move)

The magic of a computer is its ability to become almost anything you can imagine…

The magic of a computer is its ability to become almost anything you can imagine…

…as long as you can explain *exactly* what that is.

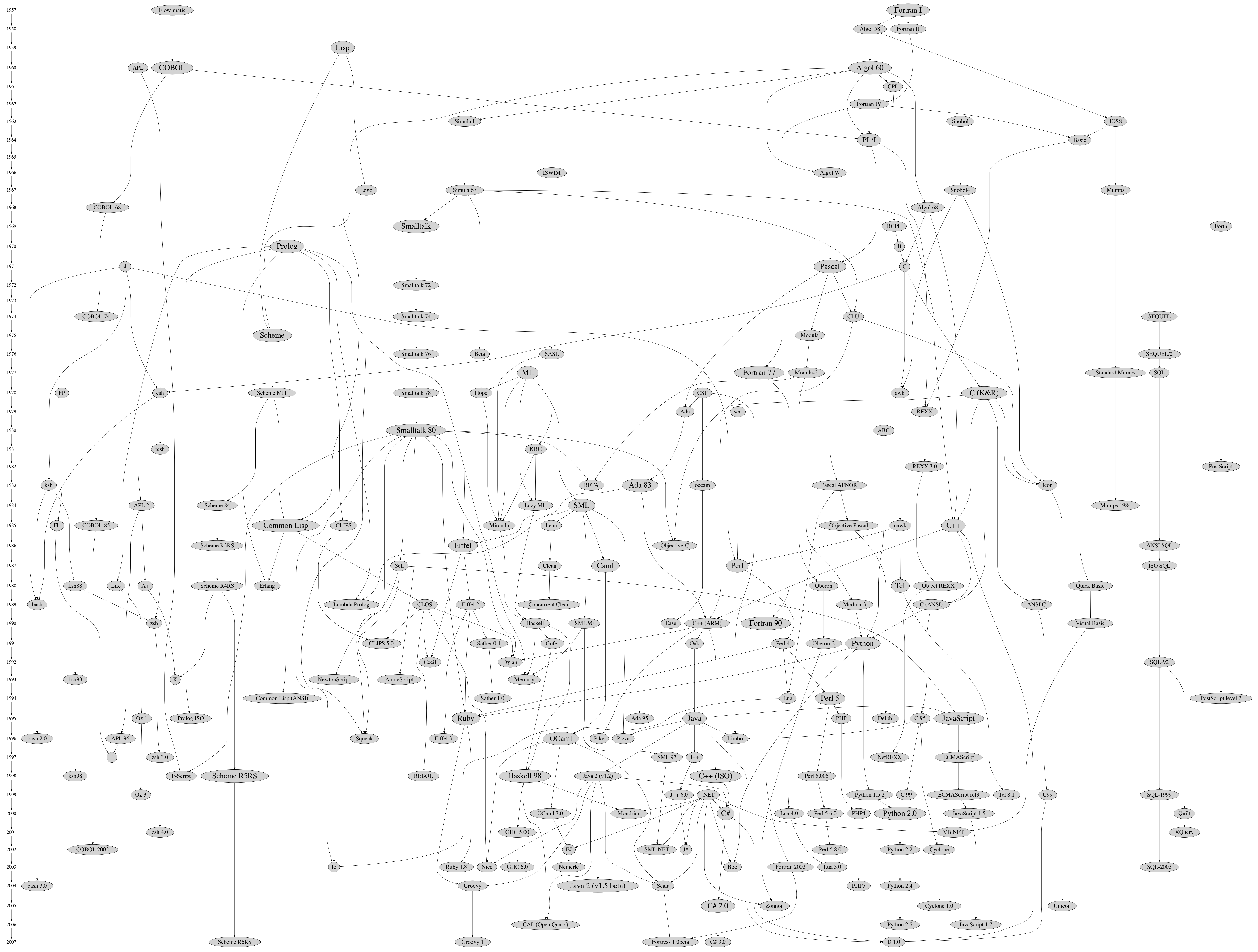
When we program a computer to do something,
everything needs to be described precisely.

FEDERAL RESERVE NOT
THE UNITED STATES OF AMERICA
THIS NOTE IS NOT WORTH THE PAPER IT'S PRINTED ON, NOHOW
D 80285227 C
WEISSHAUPT, D.C.
NONE
NO DOLLARS
ERIS 1992 K
Treasurer of the United States
Secretary of the Treasury
0

eblong.com/zarf/zero-bill.html

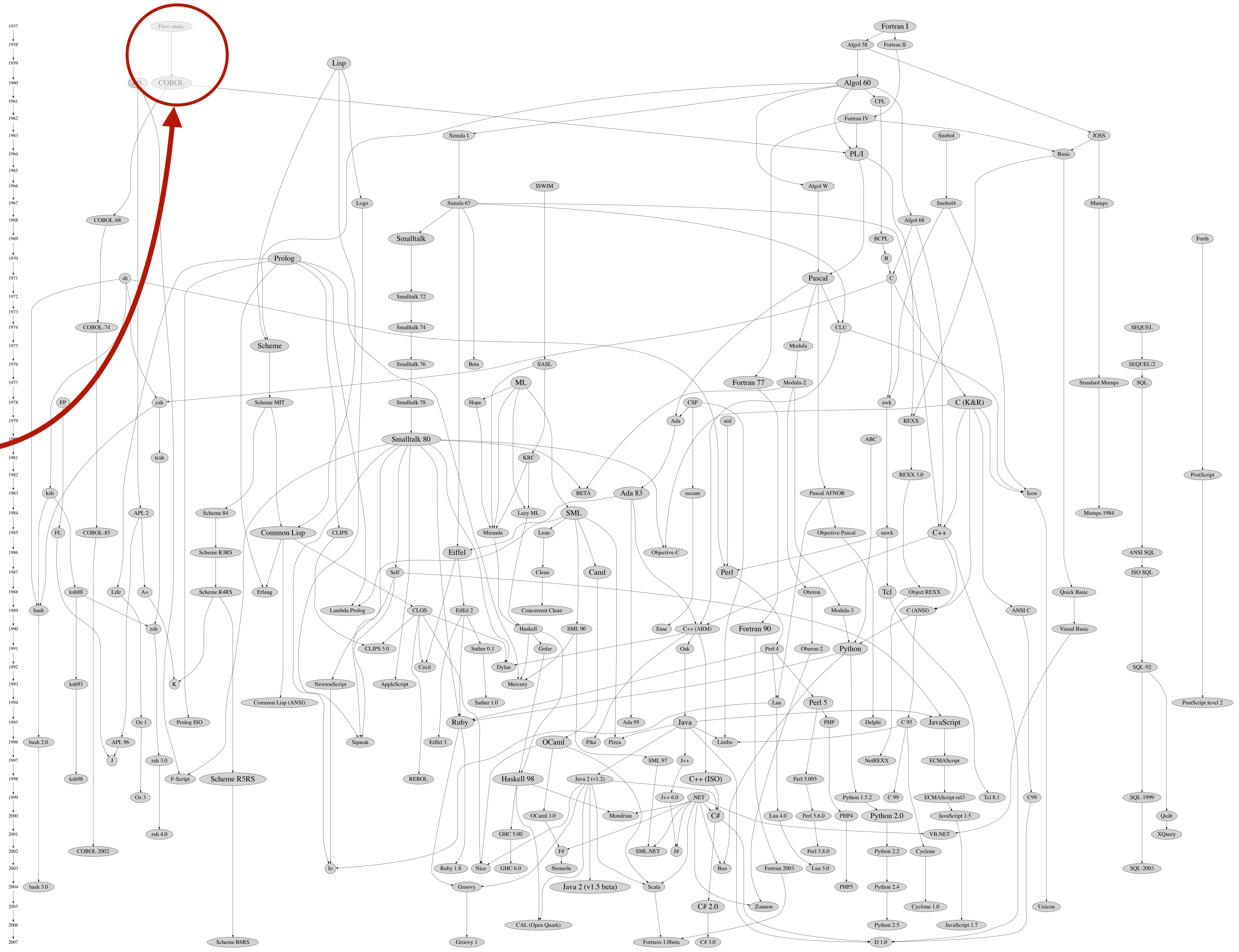
When computers behave intelligently, it’s because a person used *their* intelligence to design an intelligent program.

To tell the computer exactly how to behave, we give it instructions using a *programming language*.



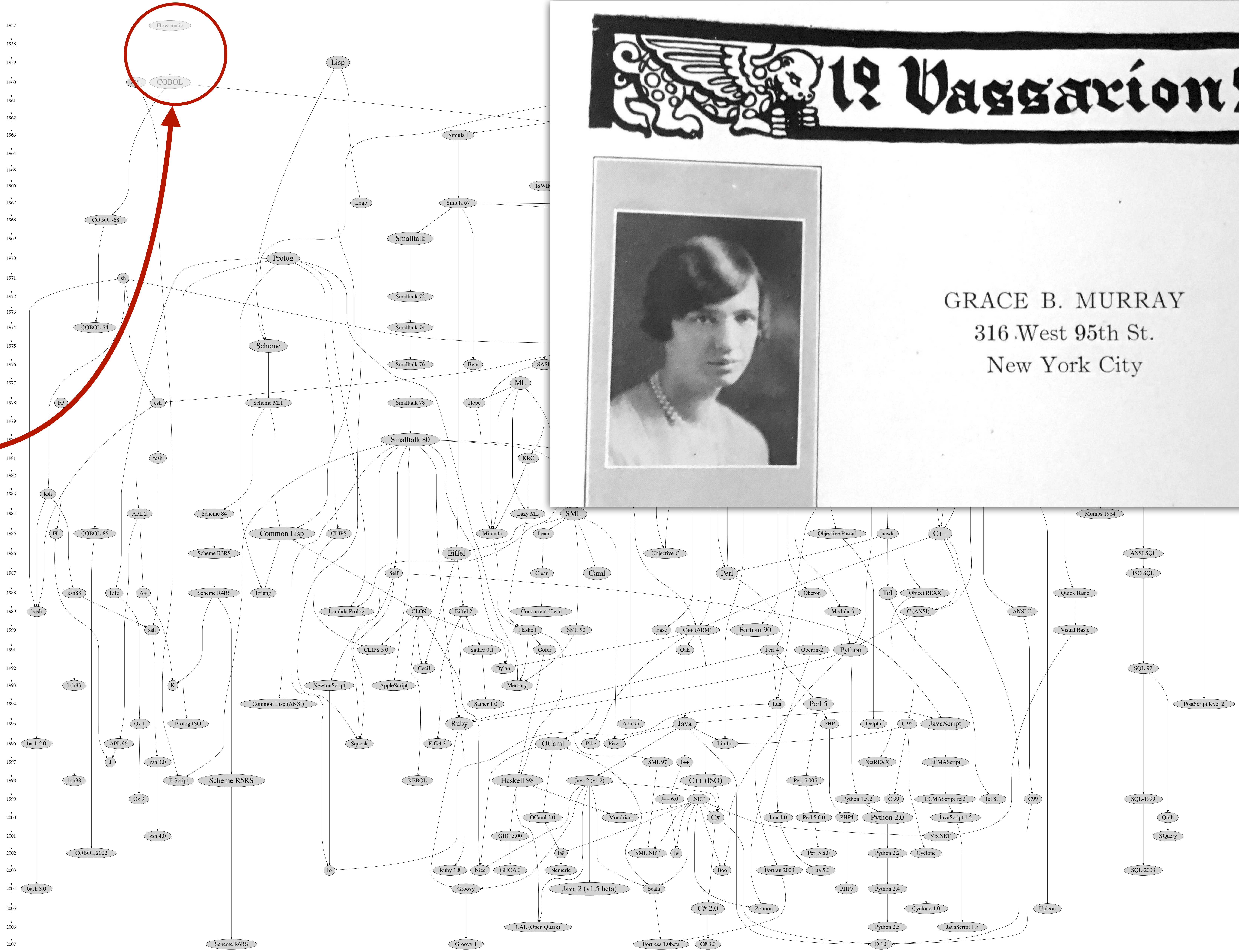
github.com/stereobooster/programming-languages-genealogical-tree

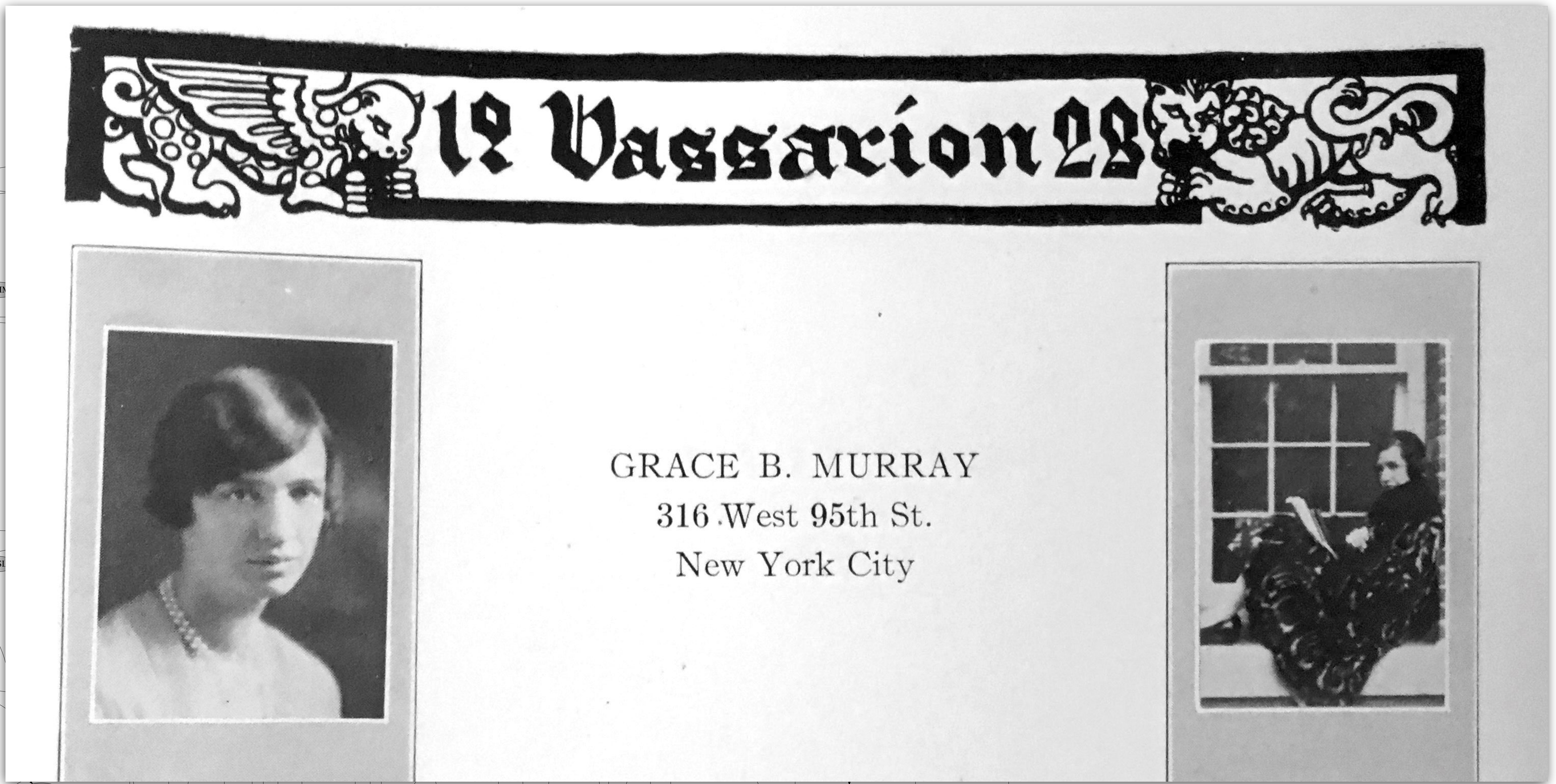
Grace Hopper



github.com/stereobooster/programming-languages-genealogical-tree

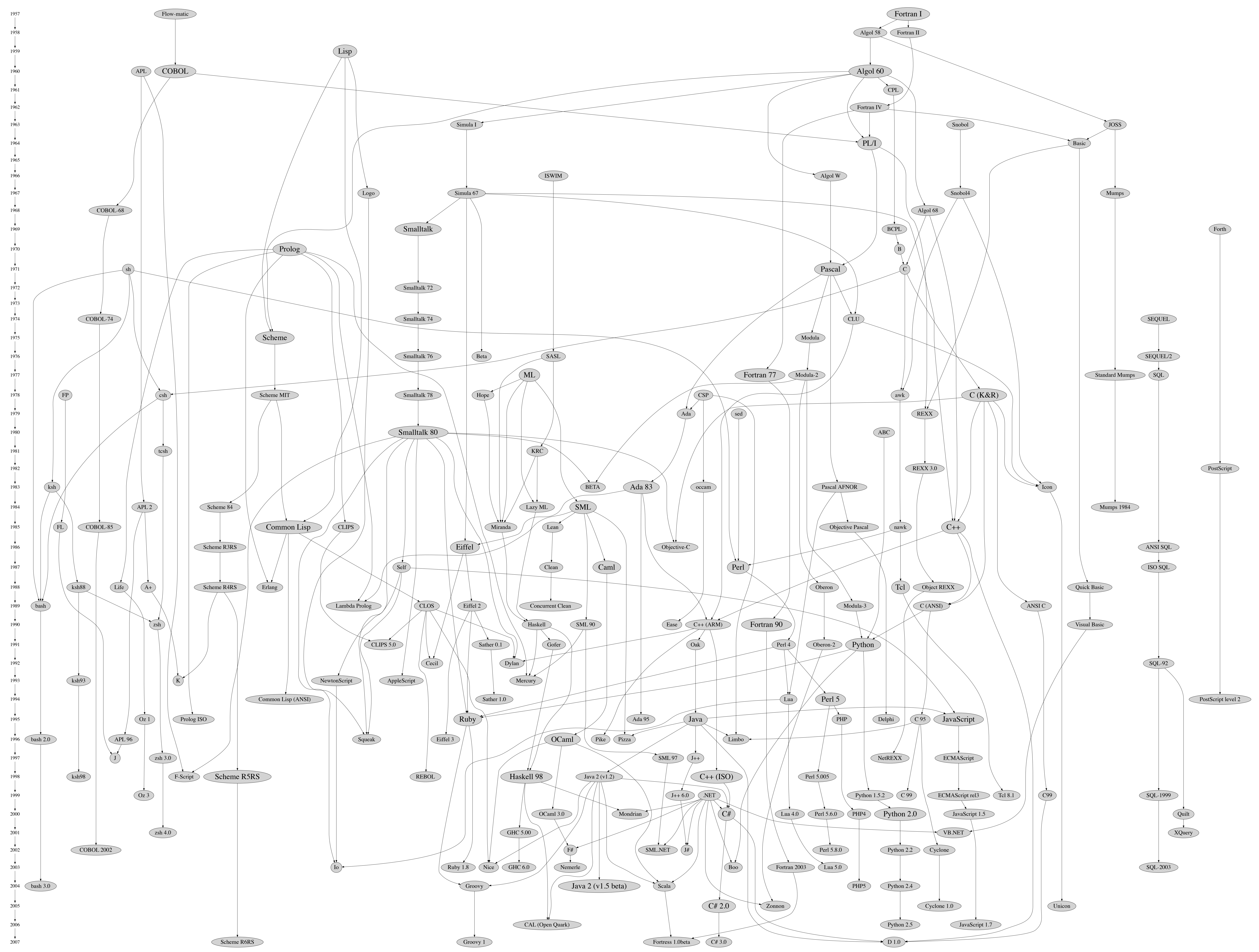
Grace Hopper




19 Vassarion 28

GRACE B. MURRAY
316 West 95th St.
New York City

github.com/stereobooster/programming-languages-genealogical-tree



github.com/stereobooster/programming-languages-genealogical-tree

There are many programming languages due to

intended use

history

habit

taste

Ancient history (my childhood)

LOGO Writer®

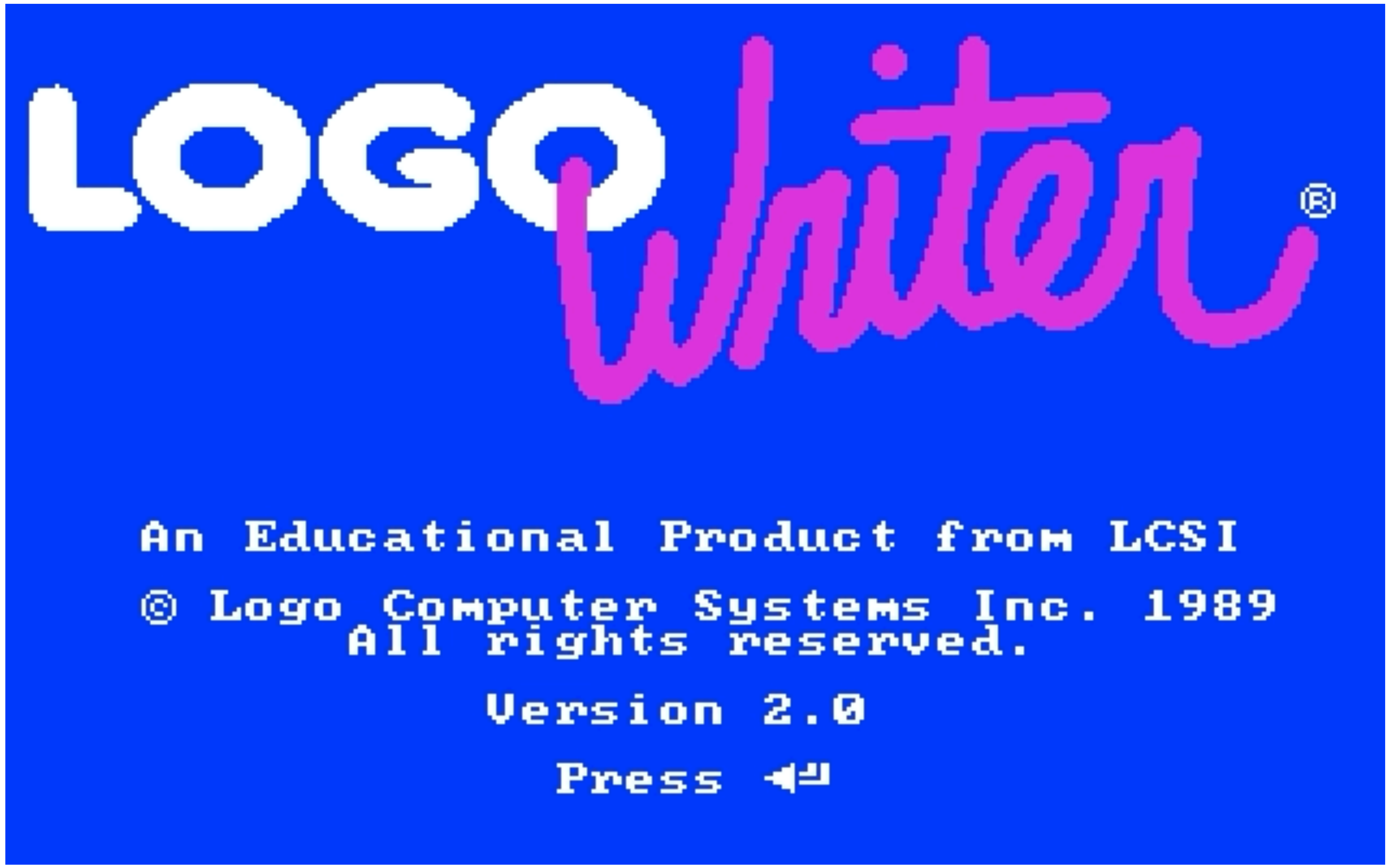
An Educational Product from LCSI

© Logo Computer Systems Inc. 1989
All rights reserved.

Version 2.0

Press ↵

Ancient history (my childhood)



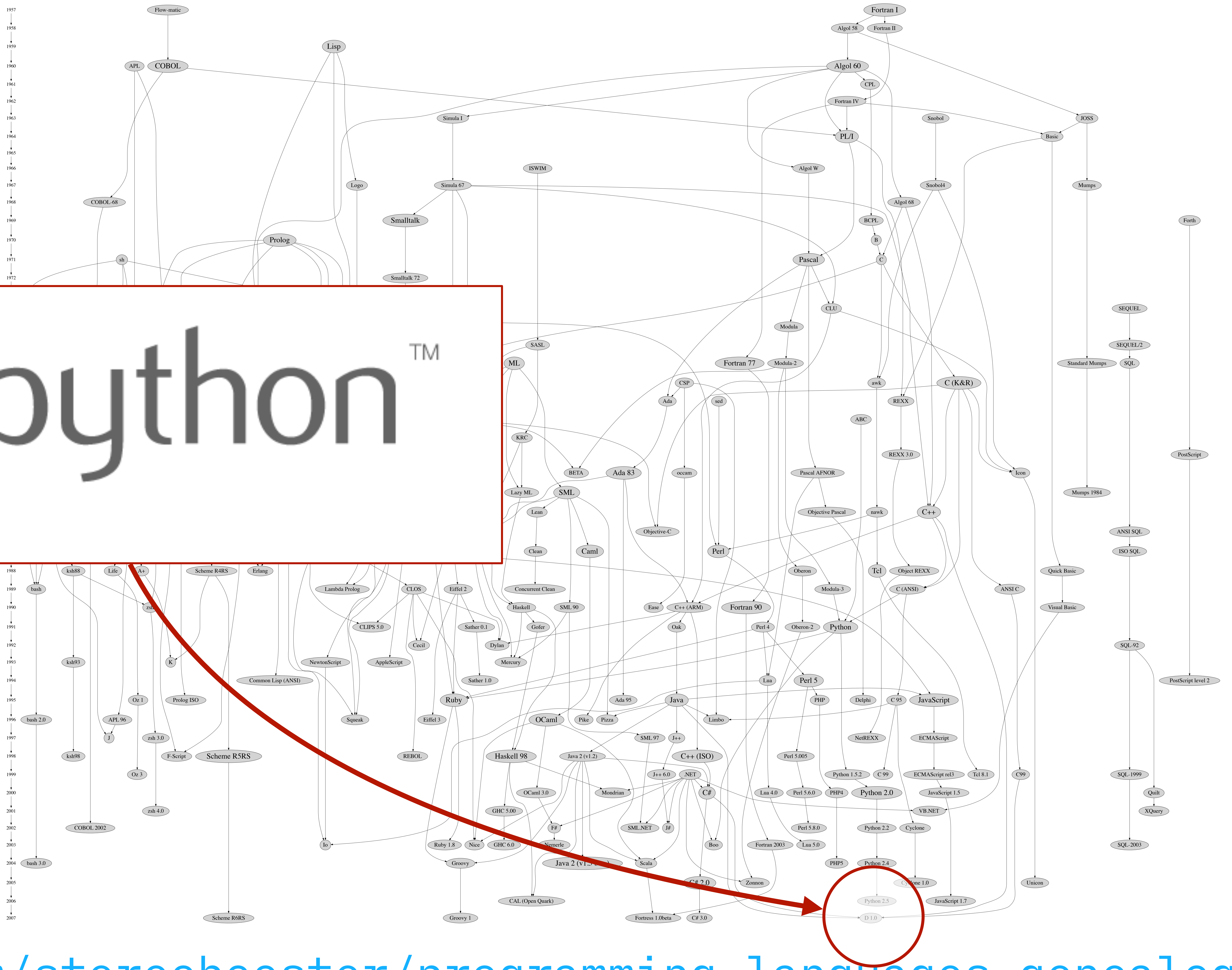
An Educational Product from LCSI

© Logo Computer Systems Inc. 1989
All rights reserved.

Version 2.0

Press ⏎

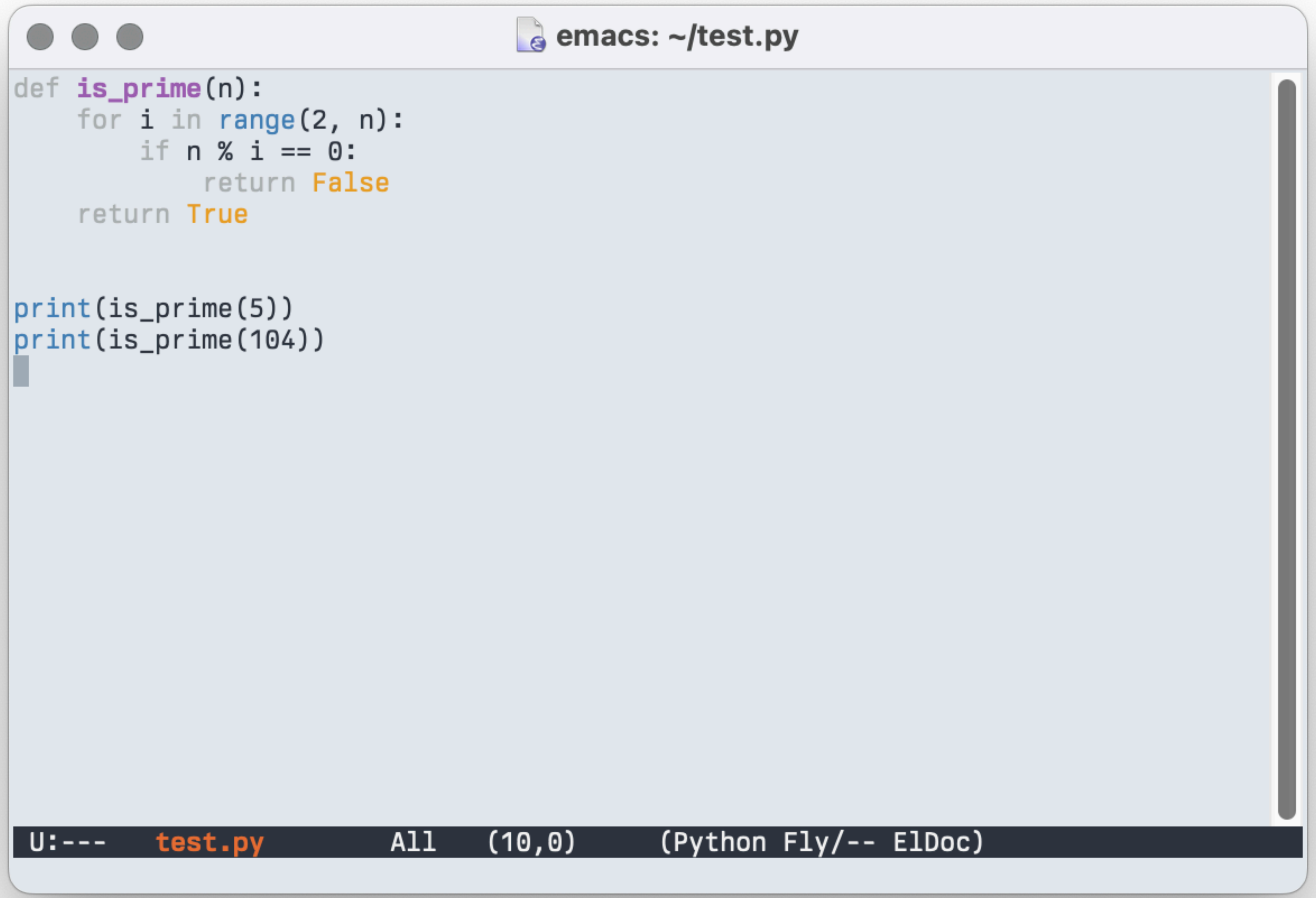
In this course, we’ll be working in a slightly more modern programming language.



python™

github.com/stereobooster/programming-languages-genealogical-tree

The traditional way of writing code is to use a text editor and then run the code in a command-line interface.

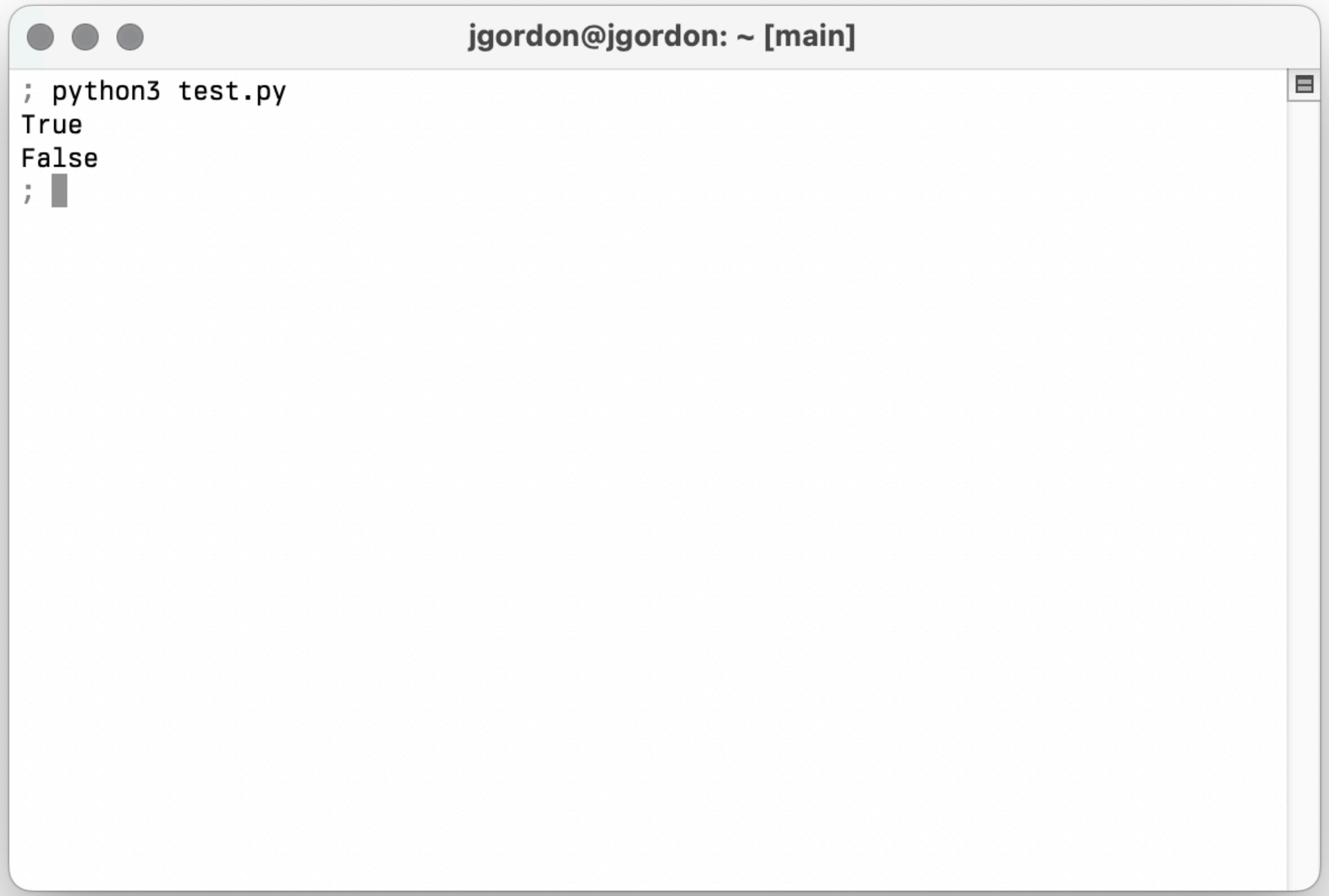


```
def is_prime(n):
    for i in range(2, n):
        if n % i == 0:
            return False
    return True


print(is_prime(5))
print(is_prime(104))
```

```
U:---   test.py      All   (10,0)     (Python Fly/-- ElDoc)
```

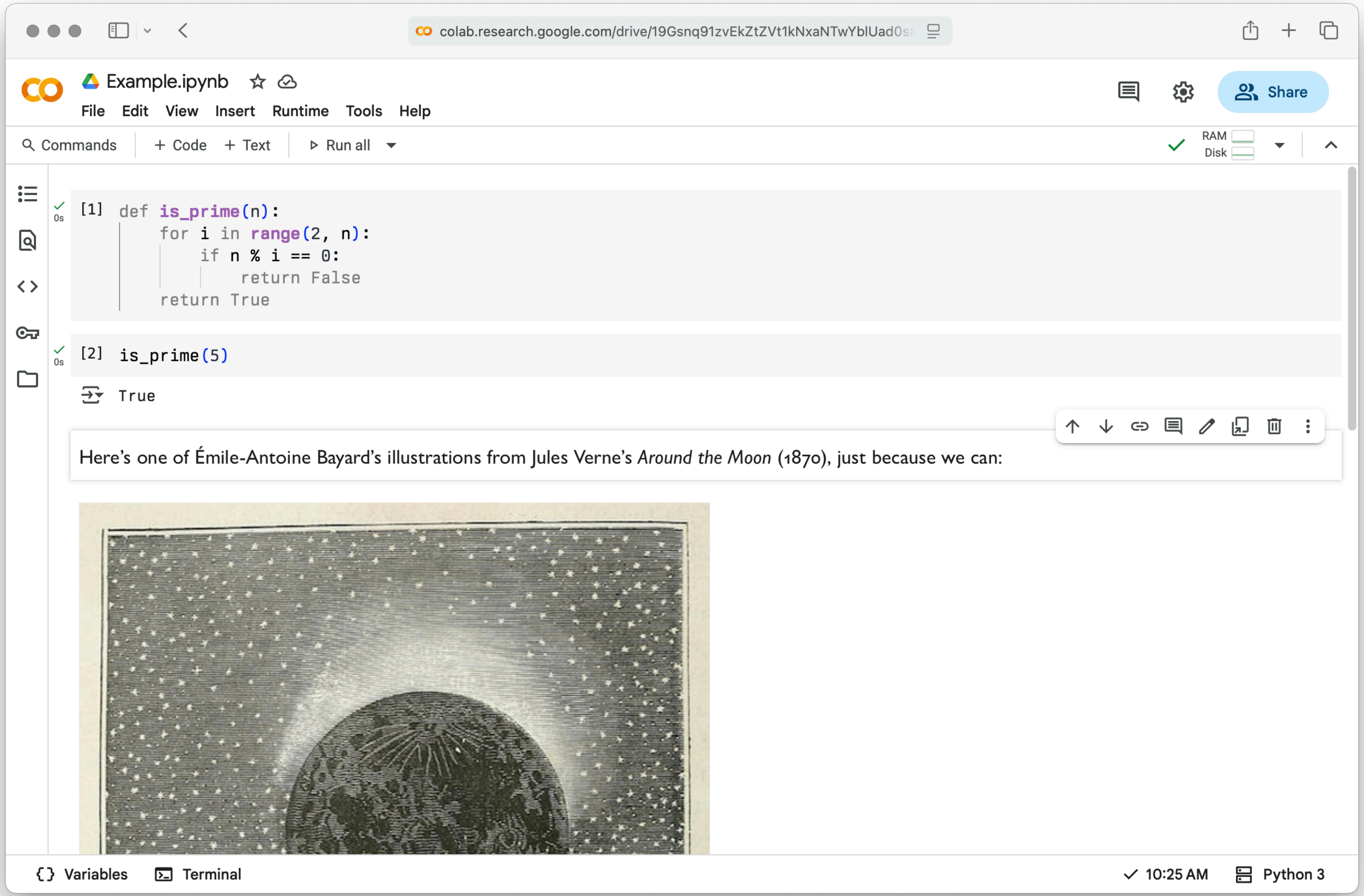
Emacs, a popular text editor



```
; python3 test.py
True
False
;
```

Command-line interface

While we could do everything in command-line interfaces, they aren't the best suited for data science work, which often requires *visualizations* and *written reports*.


colab.research.google.com/drive/19Gsnq91zvEkZtZVt1kNxaNTwYblUad0s
Example.ipynb
File Edit View Insert Runtime Tools Help
Share
Commands + Code + Text Run all
RAM Disk
Variables Terminal 10:25 AM Python 3

```
[1] def is_prime(n):
        for i in range(2, n):
            if n % i == 0:
                return False
        return True
```

```
[2] is_prime(5)
```

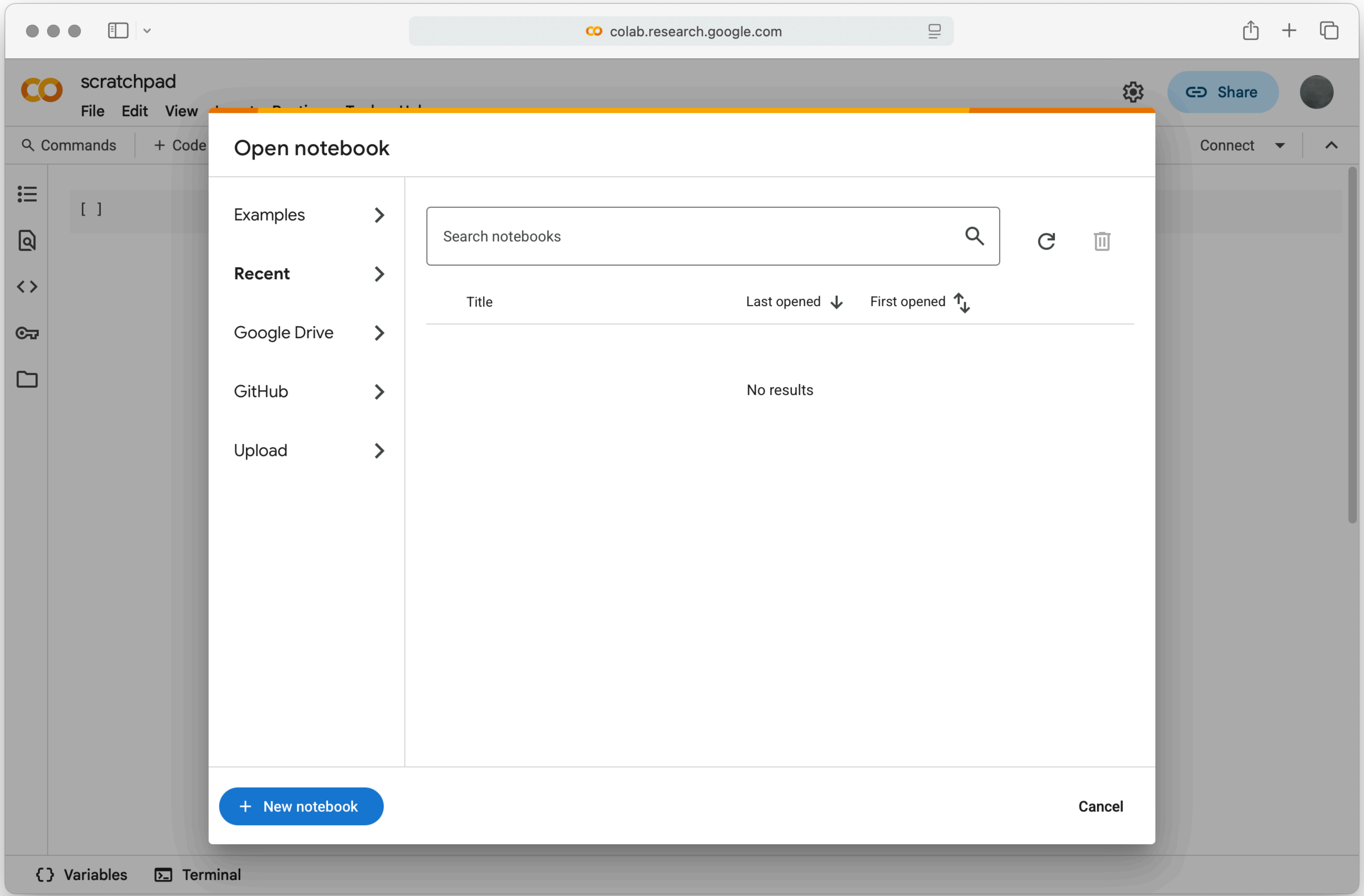
```
True
```

Here's one of Émile-Antoine Bayard's illustrations from Jules Verne's *Around the Moon* (1870), just because we can:

Jupyter notebooks allow us to write and run code in a single document, together with accompanying text, tables, and images.

For this class, we'll use Colab, Google's version of Jupyter notebooks, which let you log in with your Vassar account and store the notebooks you write in your Google Drive.

Let’s go!



colab.research.google.com

See notebook for example.

What will we do in this course?

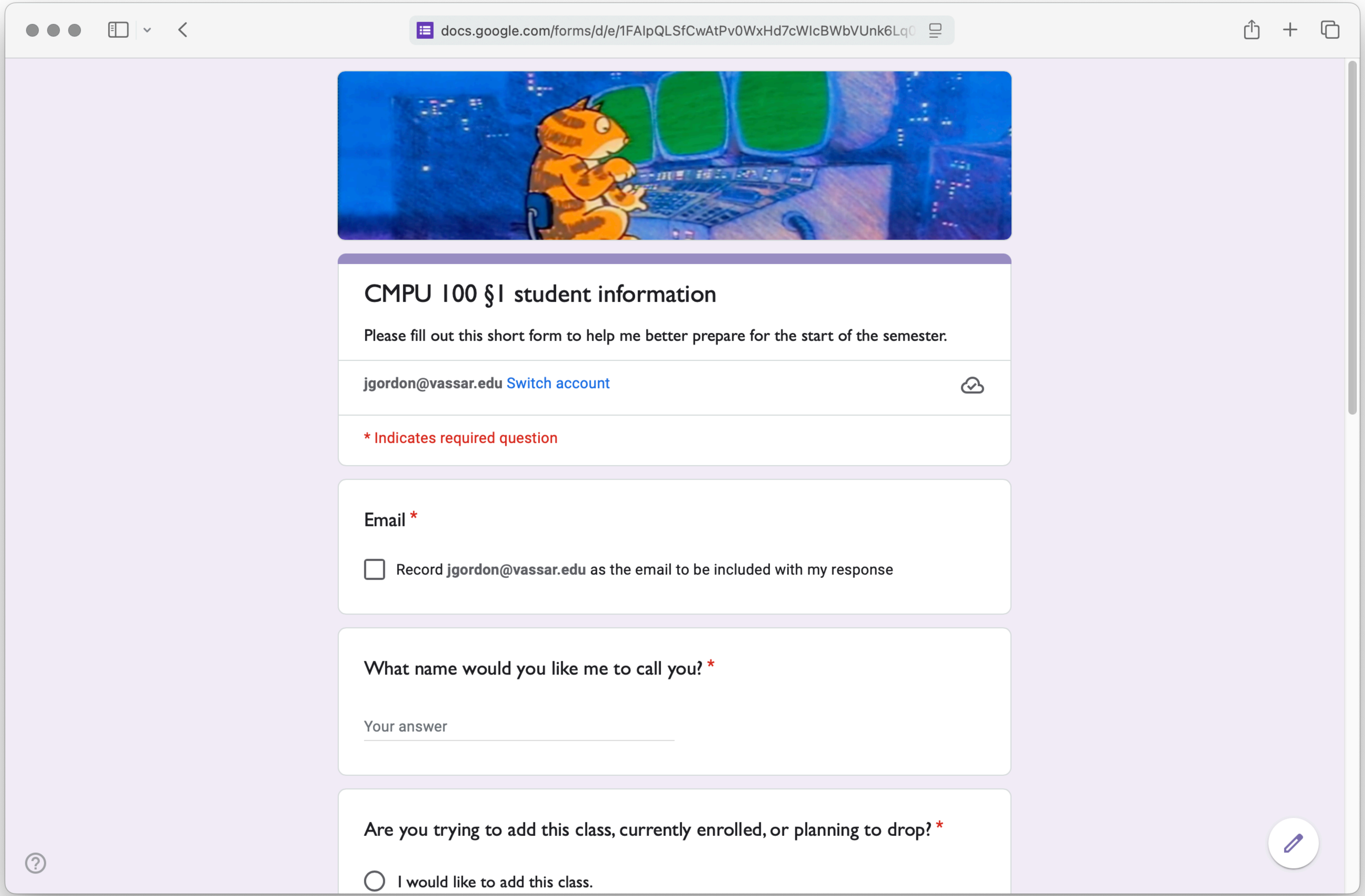
1	*Data design*	Identify and organize the data needed to solve a problem
2	*Computational problem solving*	Break a problem down into subproblems that can be solved with computations
3	*Programming*	Express computations over the data
4	*Testing*	Test those computations to make sure they're doing what they're supposed to

0 *Society*

Think about whether it’s a good idea to solve the problem – and how your solution might affect the world around you.

You will leave this course with applicable skills that you can use *even if you don’t take any future computer science or data science courses*.

Course information


docs.google.com/forms/d/e/1FAIpQLSfCwAtPv0WxHd7cWIcBWbVUnk6Lq

CMPU 100 §1 student information

Please fill out this short form to help me better prepare for the start of the semester.

jgordon@vassar.edu Switch account

* Indicates required question

Email *

☐ Record jgordon@vassar.edu as the email to be included with my response

What name would you like me to call you? *

Your answer

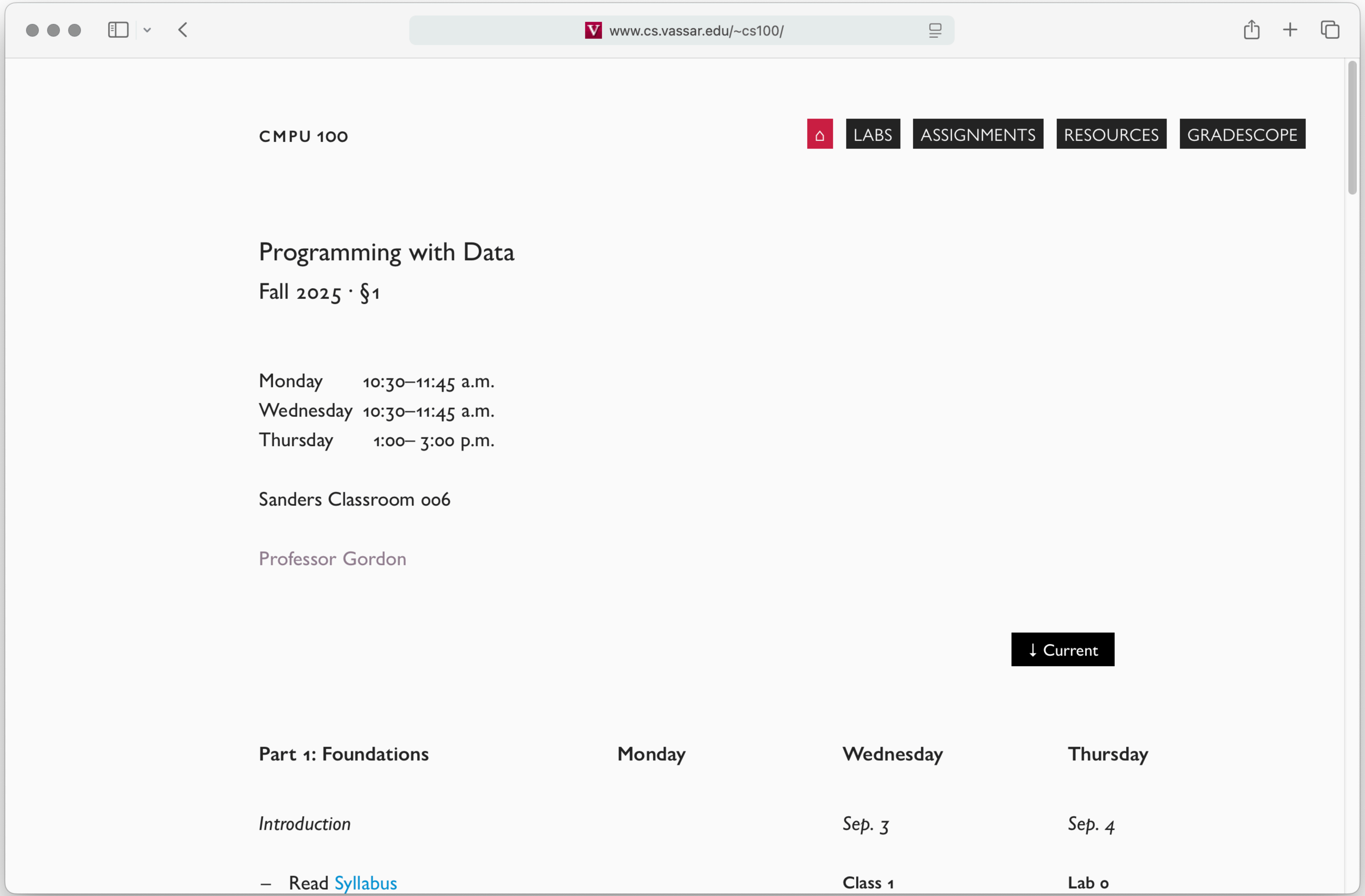
Are you trying to add this class, currently enrolled, or planning to drop? *

○ I would like to add this class.

forms.gle/NYvWEnh7Ub86hU8J6

Class	Monday	10:30–11:45 a.m.
	Wednesday	10:30–11:45 a.m.
Lab	Thursday	1:00– 3:00 p.m.

Sanders Classroom 006



CMPU 100

Programming with Data

Fall 2025 · §1

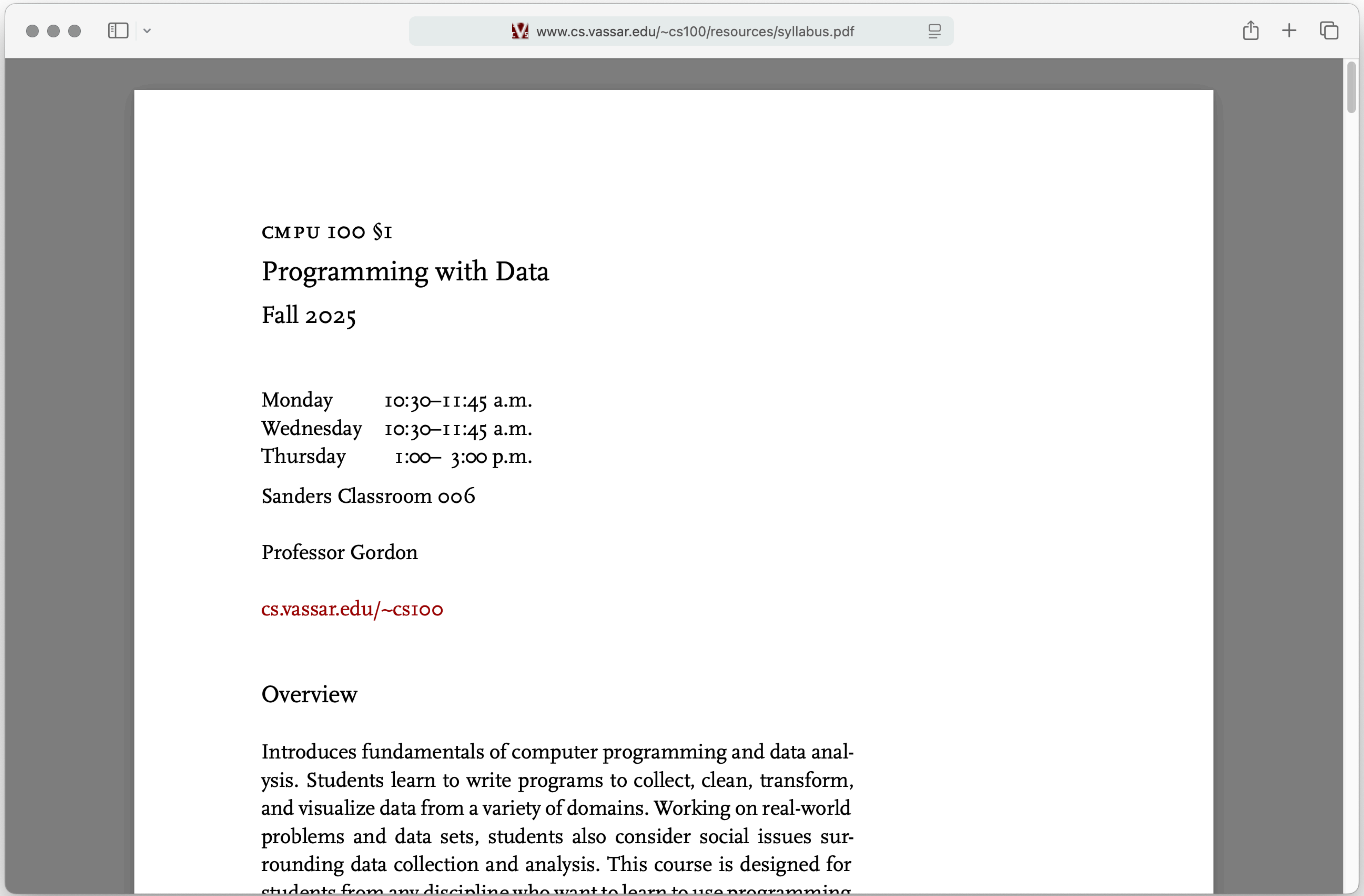
Monday	10:30–11:45 a.m.
Wednesday	10:30–11:45 a.m.
Thursday	1:00– 3:00 p.m.

Sanders Classroom 006

Professor Gordon

Part 1: Foundations	Monday	Wednesday	Thursday
Introduction		*Sep. 3*	*Sep. 4*
– Read Syllabus		**Class 1**	**Lab 0**

cs.vassar.edu/~cs100



CMPU 100 §1

Programming with Data

Fall 2025

Monday	10:30–11:45 a.m.
Wednesday	10:30–11:45 a.m.
Thursday	1:00– 3:00 p.m.

Sanders Classroom 006

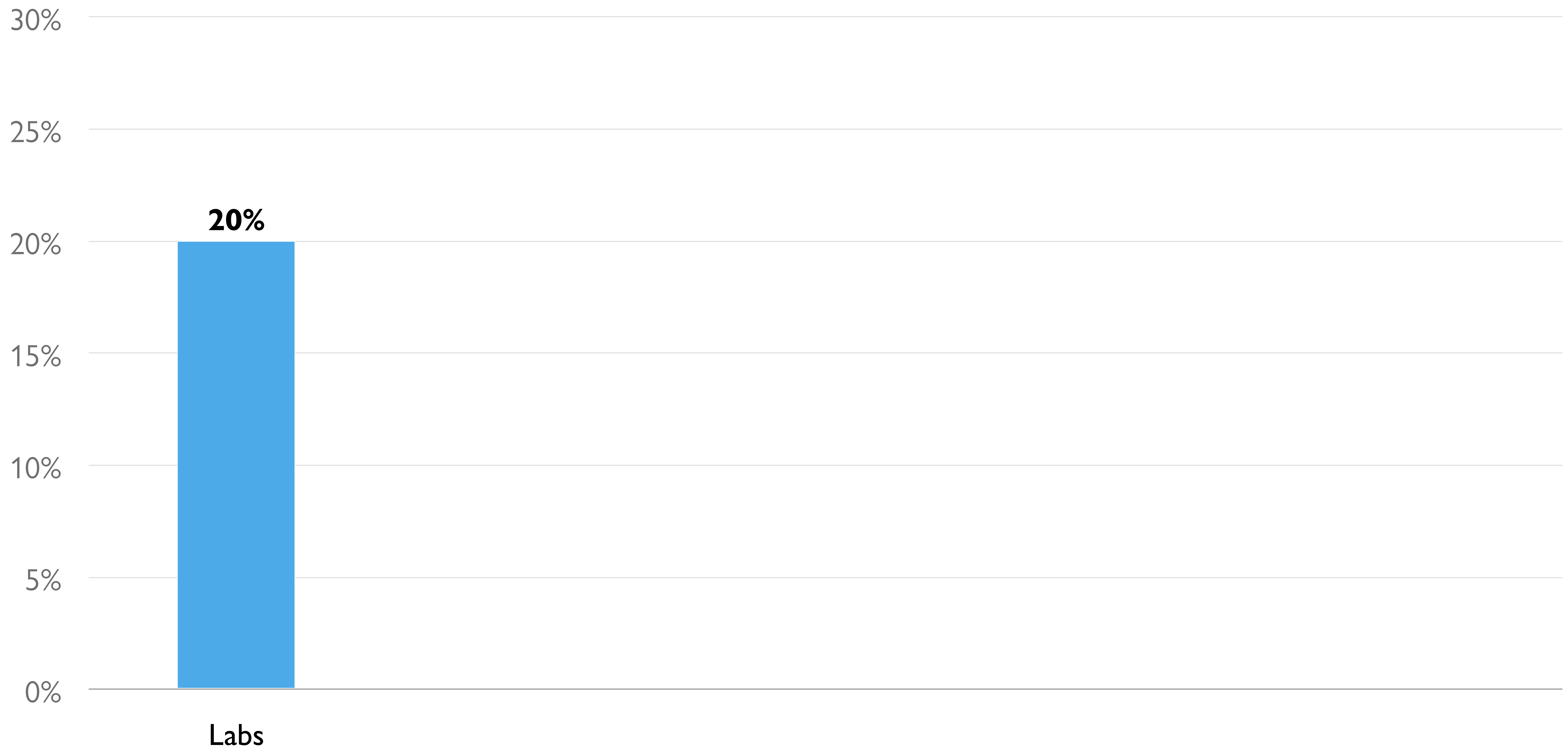
Professor Gordon

cs.vassar.edu/~cs100

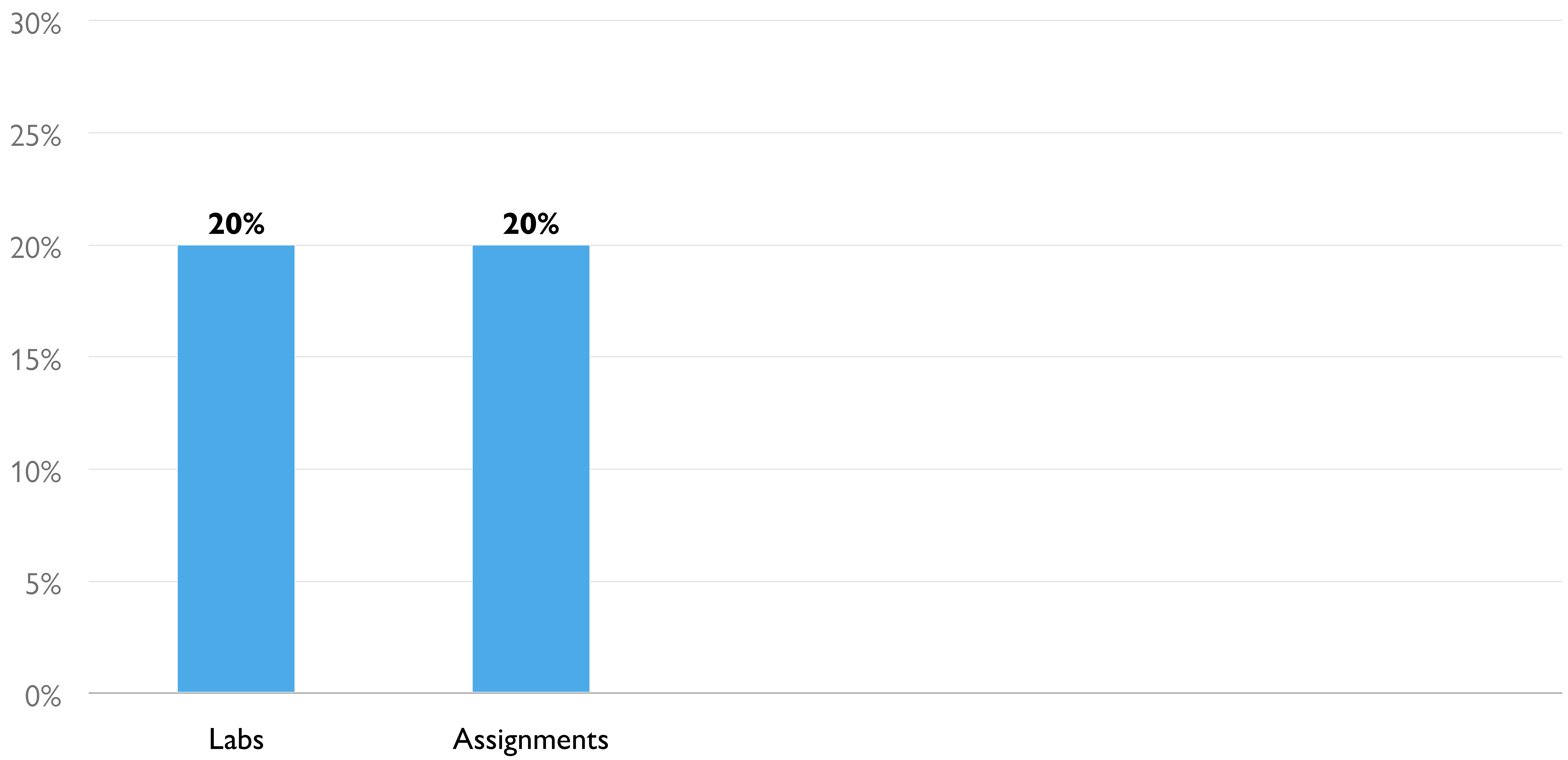
Overview

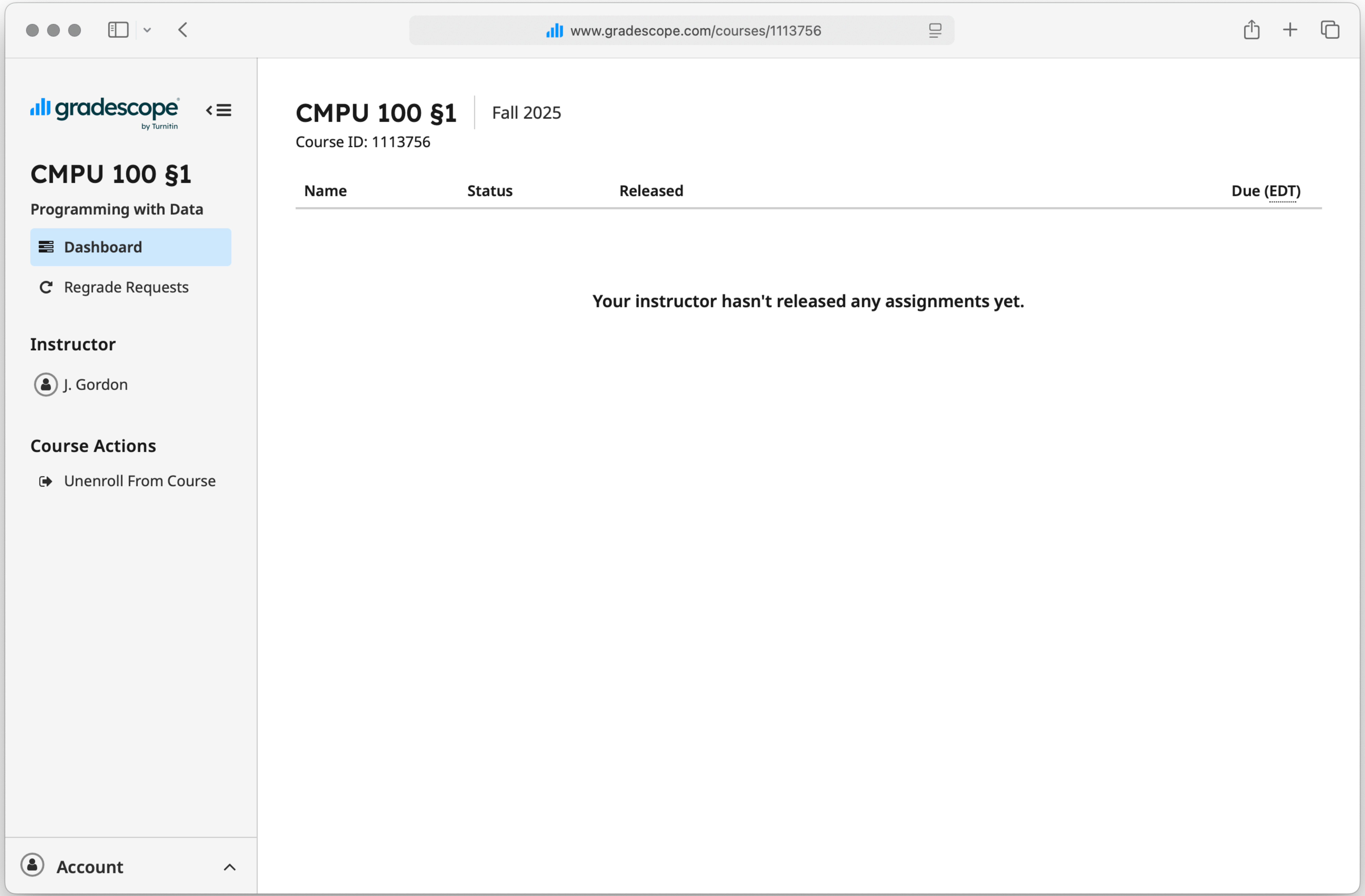
Introduces fundamentals of computer programming and data analysis. Students learn to write programs to collect, clean, transform, and visualize data from a variety of domains. Working on real-world problems and data sets, students also consider social issues surrounding data collection and analysis. This course is designed for students from any discipline who want to learn to use programming

Grading



Grading





www.gradescope.com/courses/1113756

gradescope by Turnitin

CMPU 100 §1

Programming with Data

Dashboard

Regrade Requests

Instructor

J. Gordon

Course Actions

Unenroll From Course

Account

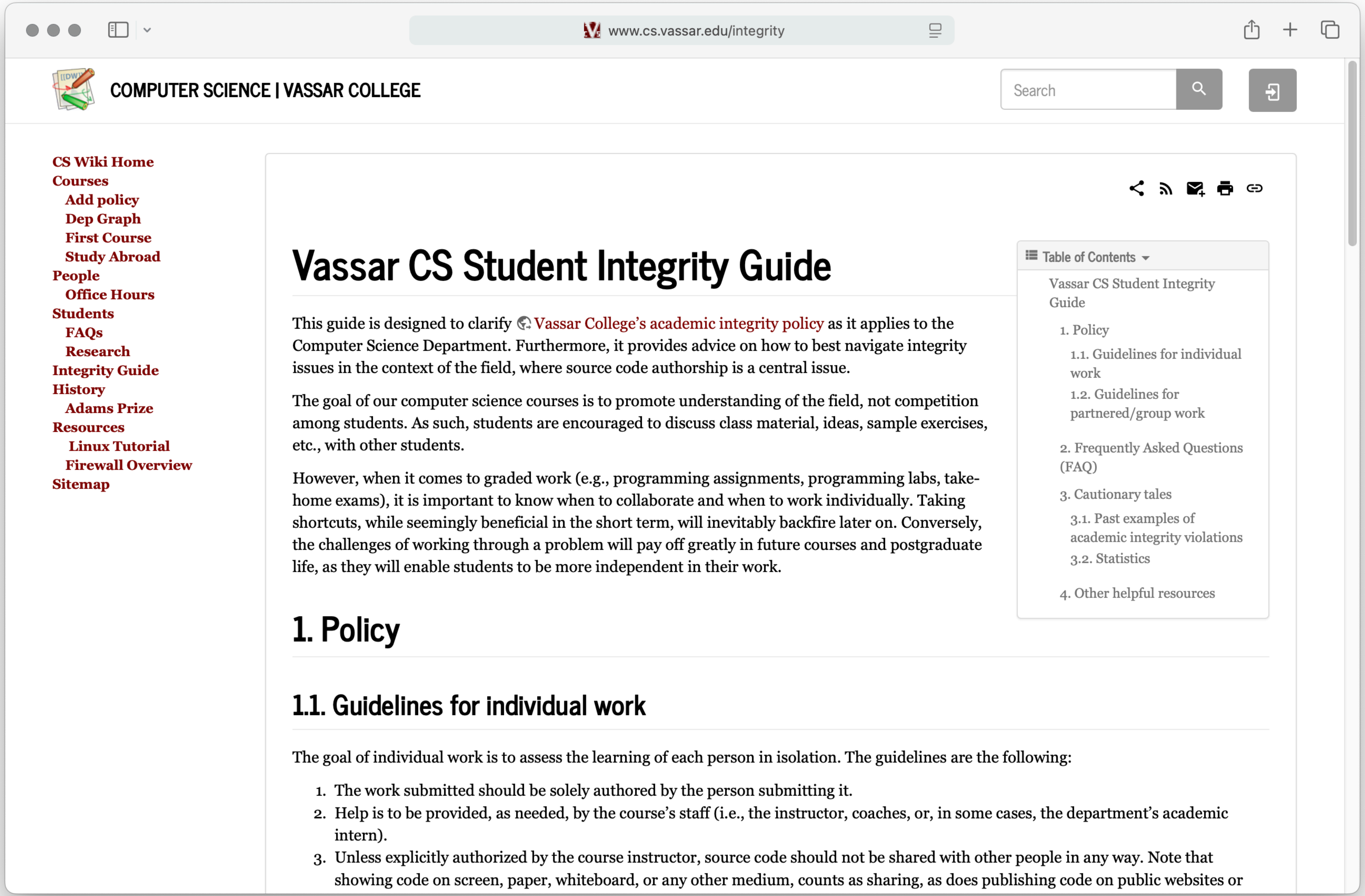
CMPU 100 §1 | Fall 2025

Course ID: 1113756

Name	Status	Released	Due (EDT)

Your instructor hasn't released any assignments yet.

gradescope.com


www.cs.vassar.edu/integrity
COMPUTER SCIENCE | VASSAR COLLEGE
Search

CS Wiki Home
Courses
Add policy
Dep Graph
First Course
Study Abroad
People
Office Hours
Students
FAQs
Research
Integrity Guide
History
Adams Prize
Resources
Linux Tutorial
Firewall Overview
Sitemap

Vassar CS Student Integrity Guide

Table of Contents

This guide is designed to clarify Vassar College's academic integrity policy as it applies to the Computer Science Department. Furthermore, it provides advice on how to best navigate integrity issues in the context of the field, where source code authorship is a central issue.

The goal of our computer science courses is to promote understanding of the field, not competition among students. As such, students are encouraged to discuss class material, ideas, sample exercises, etc., with other students.

However, when it comes to graded work (e.g., programming assignments, programming labs, take-home exams), it is important to know when to collaborate and when to work individually. Taking shortcuts, while seemingly beneficial in the short term, will inevitably backfire later on. Conversely, the challenges of working through a problem will pay off greatly in future courses and postgraduate life, as they will enable students to be more independent in their work.

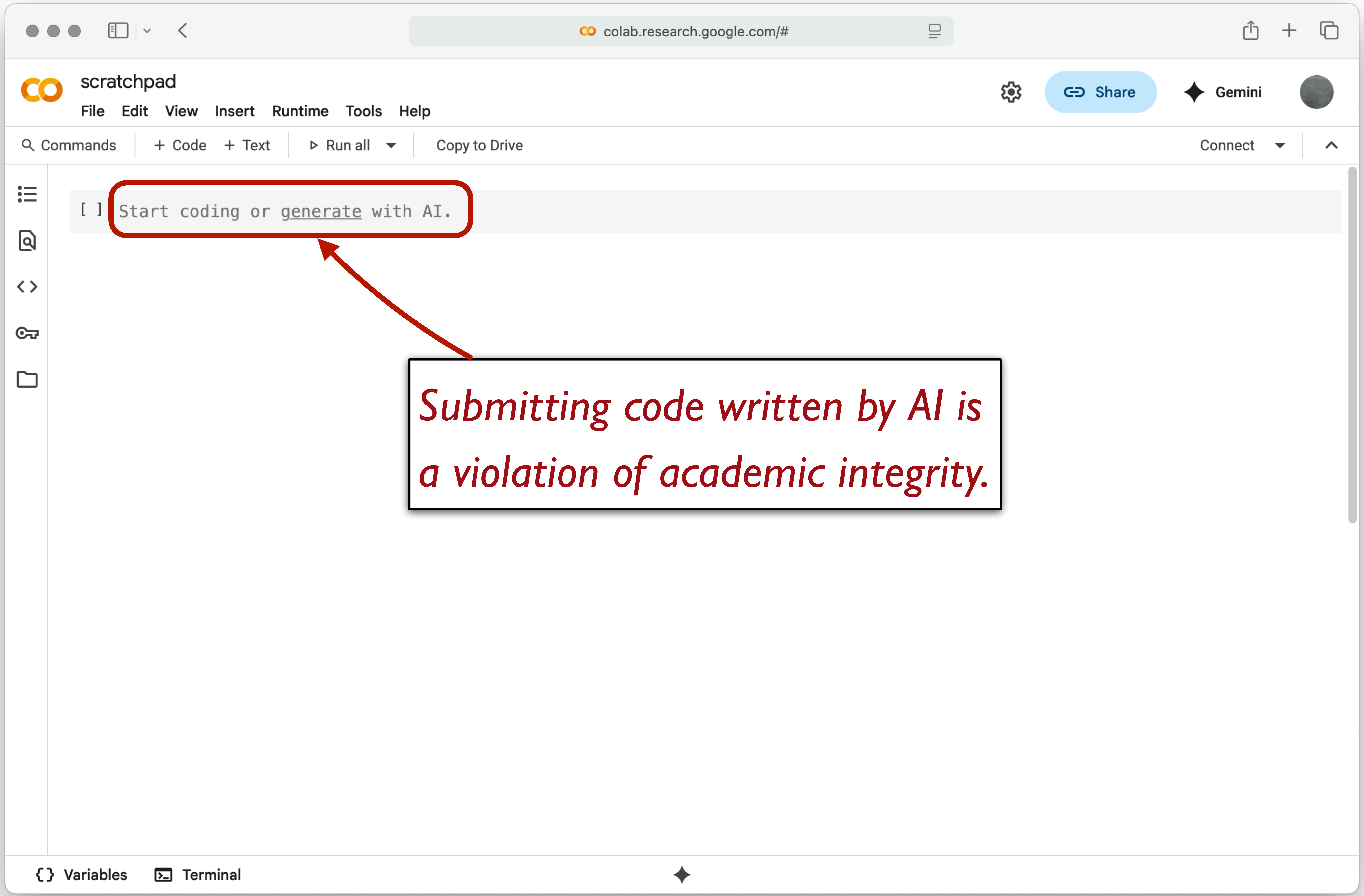
1. Policy

1.1. Guidelines for individual work

The goal of individual work is to assess the learning of each person in isolation. The guidelines are the following:

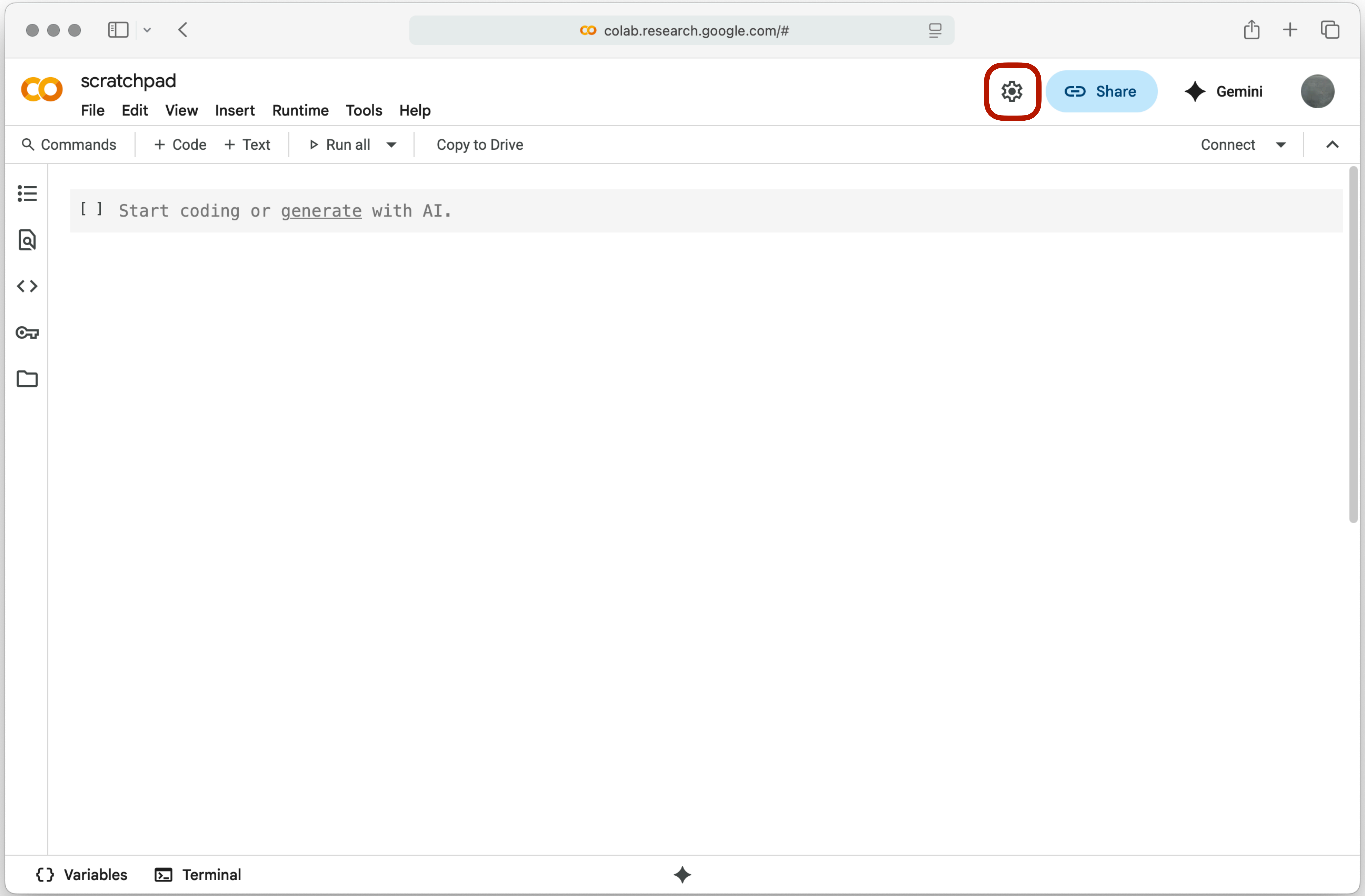
1. The work submitted should be solely authored by the person submitting it.
2. Help is to be provided, as needed, by the course's staff (i.e., the instructor, coaches, or, in some cases, the department's academic intern).
3. Unless explicitly authorized by the course instructor, source code should not be shared with other people in any way. Note that showing code on screen, paper, whiteboard, or any other medium, counts as sharing, as does publishing code on public websites or

cs.vassar.edu/integrity

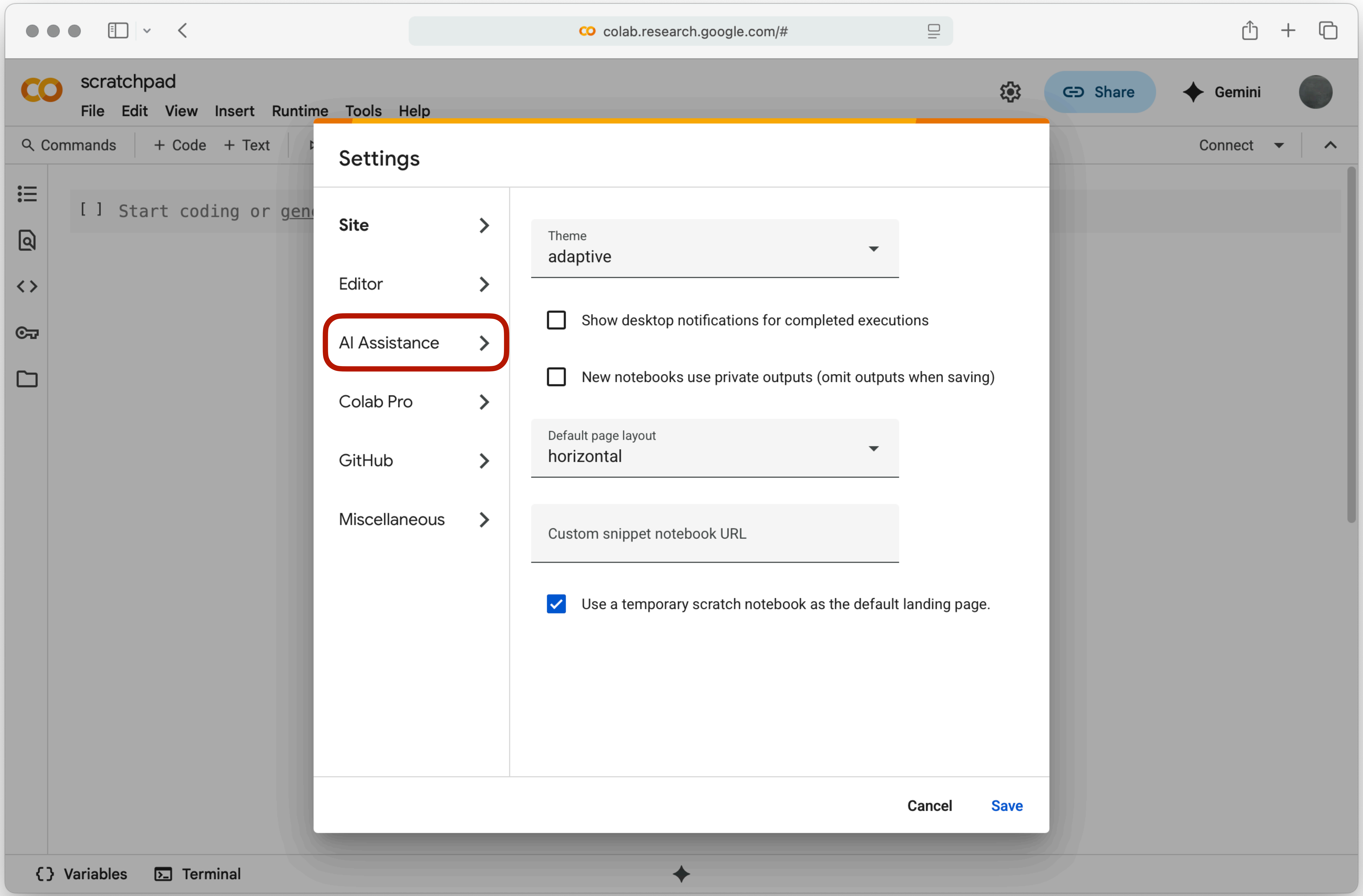


```
[ ] Start coding or generate with AI.
```

Submitting code written by AI is a violation of academic integrity.



```
[ ] Start coding or generate with AI.
```



Settings

- Site
- Editor
- AI Assistance
- Colab Pro
- GitHub
- Miscellaneous

Theme: adaptive

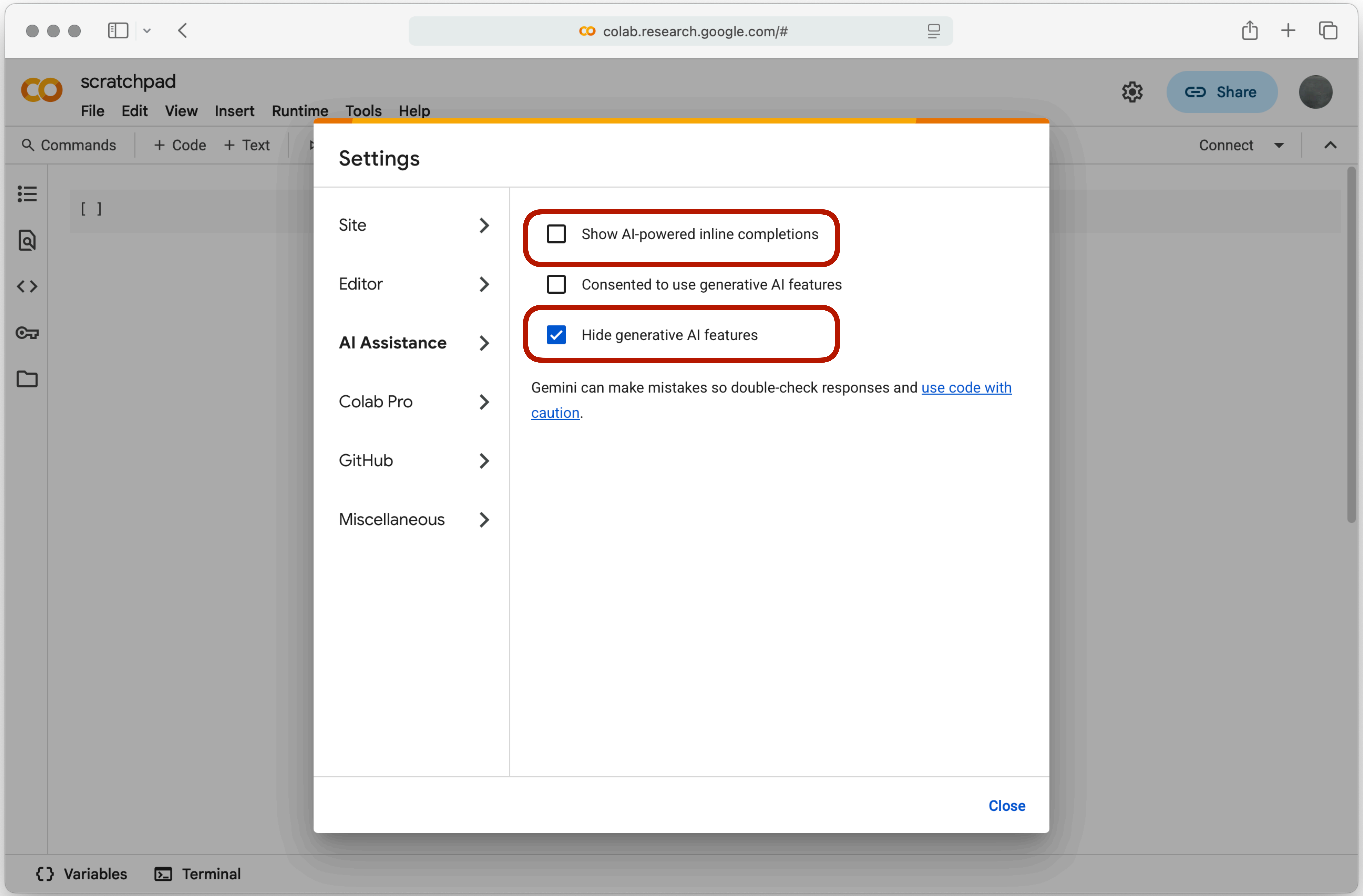
- [ ] Show desktop notifications for completed executions
- [ ] New notebooks use private outputs (omit outputs when saving)

Default page layout: horizontal

Custom snippet notebook URL

- [✓] Use a temporary scratch notebook as the default landing page.

Cancel Save



Settings

- Site
- Editor
- **AI Assistance**
- Colab Pro
- GitHub
- Miscellaneous

☐ Show AI-powered inline completions

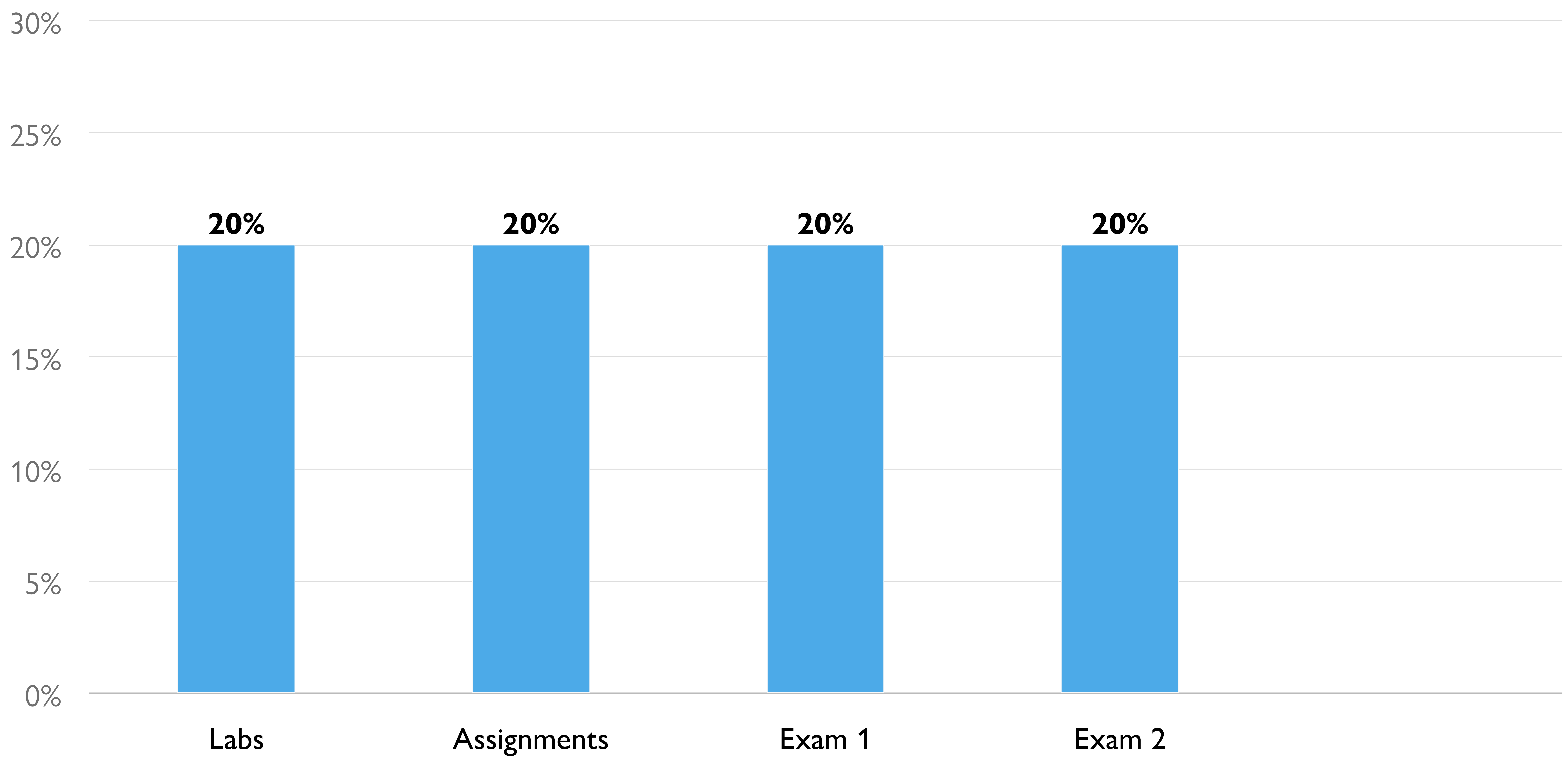
☐ Consented to use generative AI features

☑ Hide generative AI features

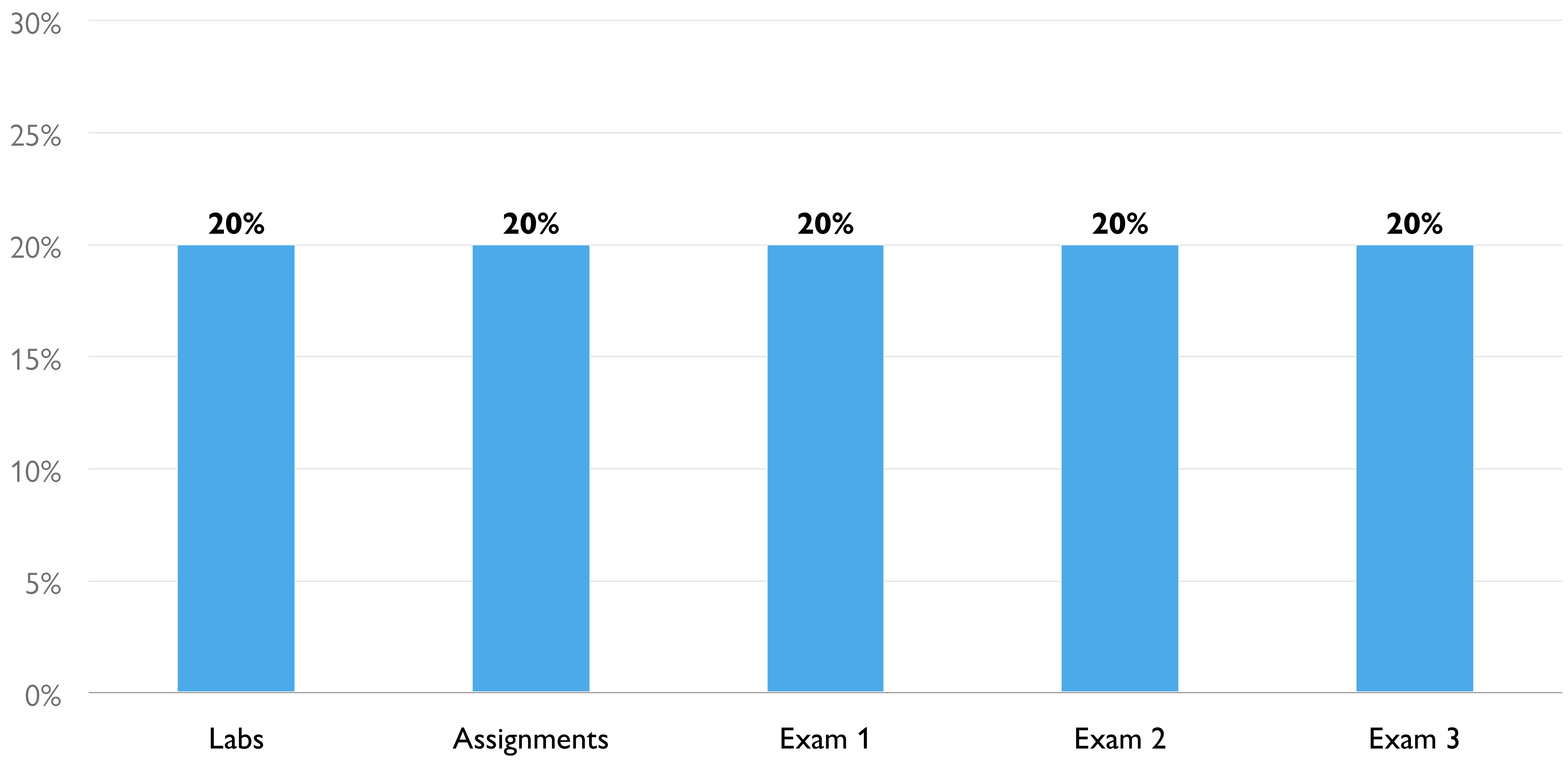
Gemini can make mistakes so double-check responses and use code with caution.

Close

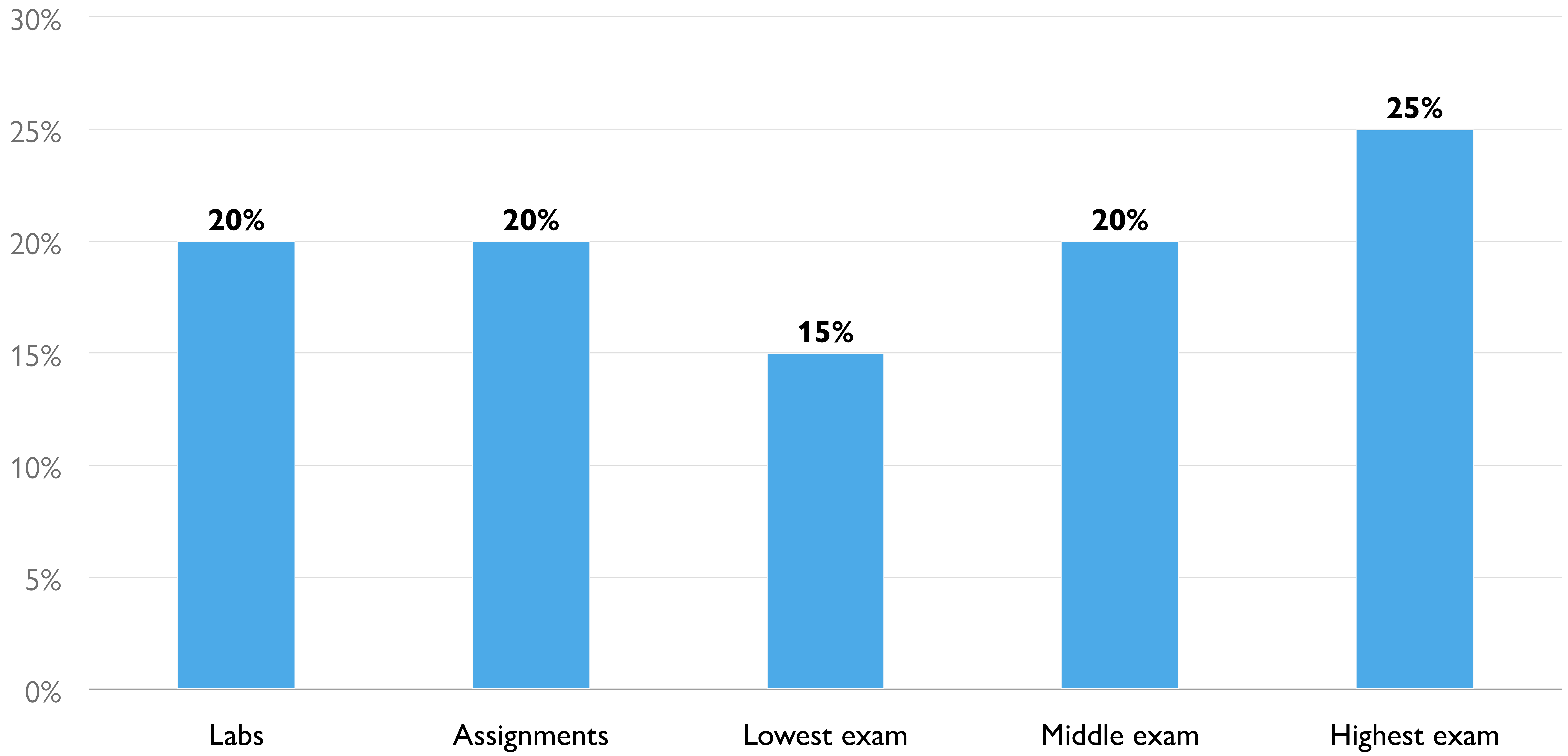
Grading

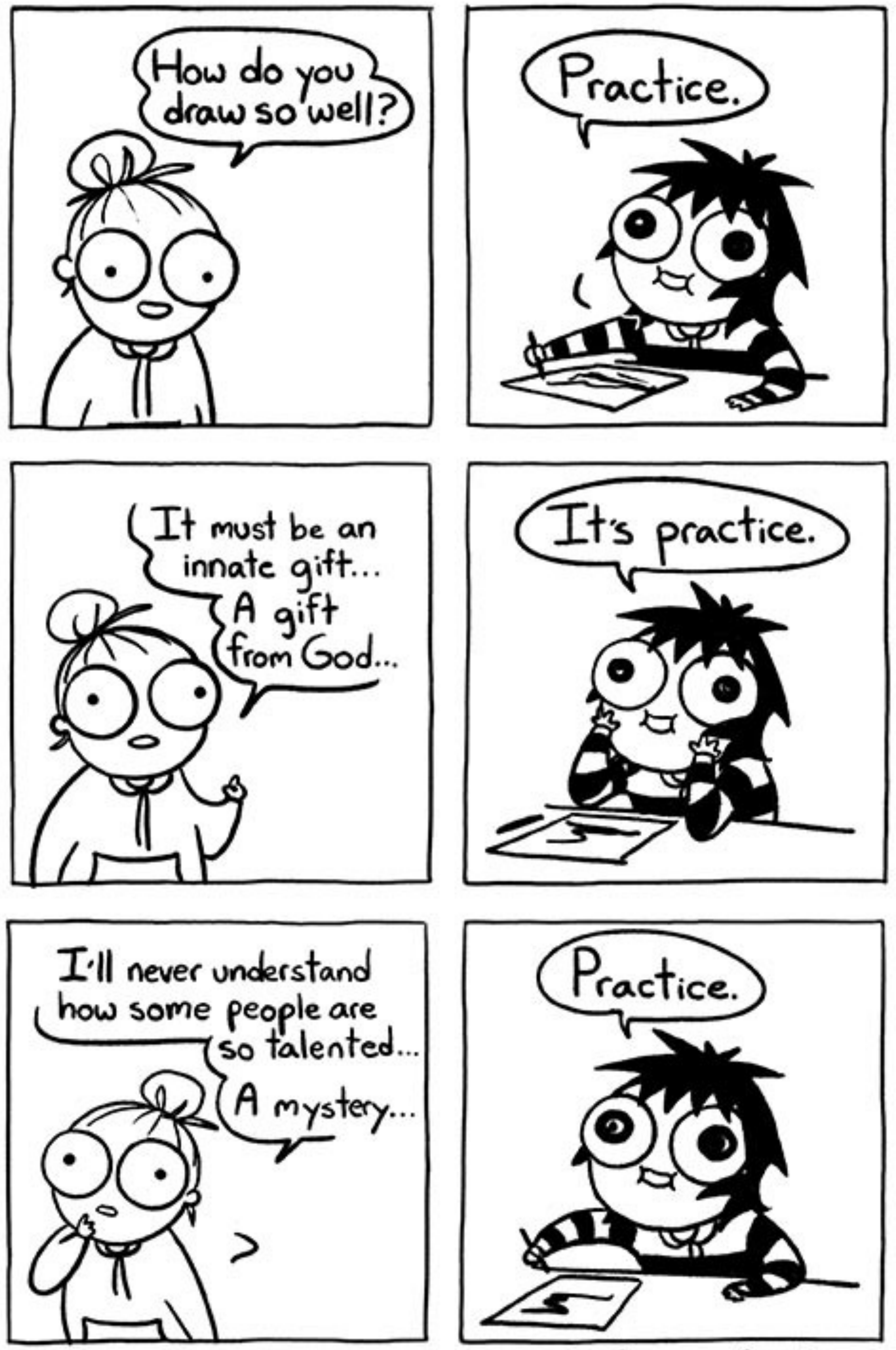


Grading


30%
25%
20%
15%
10%
5%
0%
20%
20%
20%
20%
20%
Labs
Assignments
Exam 1
Exam 2
Exam 3

Grading


30%
25%
20%
15%
10%
5%
0%
20%
20%
15%
20%
25%
Labs
Assignments
Lowest exam
Middle exam
Highest exam


How do you draw so well?
Practice.
It must be an innate gift... A gift from God...
It's practice.
I'll never understand how some people are so talented... A mystery...
Practice.

©Sarah Andersen

“All through our education, we are being taught a kind of reverse mindfulness. A kind of Future Studies where – via the guise of mathematics, or literature, or history, or computer programming, or French – we are being taught to think of a time different to the time we are in. Exam time. Job time. When-we-are-grown-up time.

“To see the act of learning as something not for its own sake but because of what it will *get* you reduces the wonder of humanity. We are thinking, feeling, art-making, knowledge-hungry, marvelous animals, who understand ourselves and our world through the act of learning. It is an end in itself. It has far more to offer than the things it lets us write on application forms. It is a way to love living right now.”

Matt Haig, *Notes on a Nervous Planet*

We’ve got a big journey ahead of us. I hope you’re excited!

Acknowledgments

This class incorporates material from:

Peter J. Denning and Matti Tedre, *Computational Thinking*

W. Daniel Hillis, *The Pattern on the Stone*