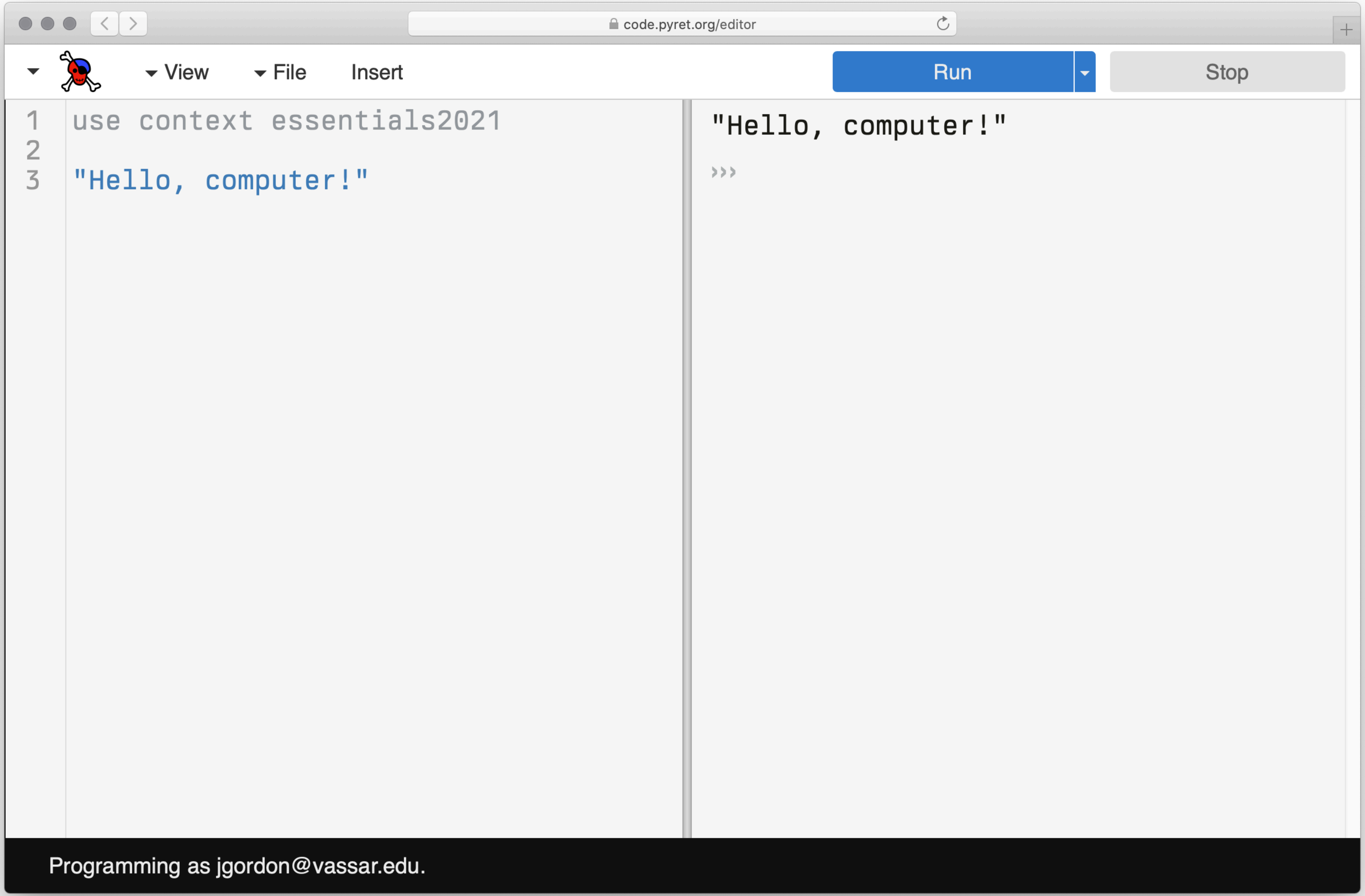
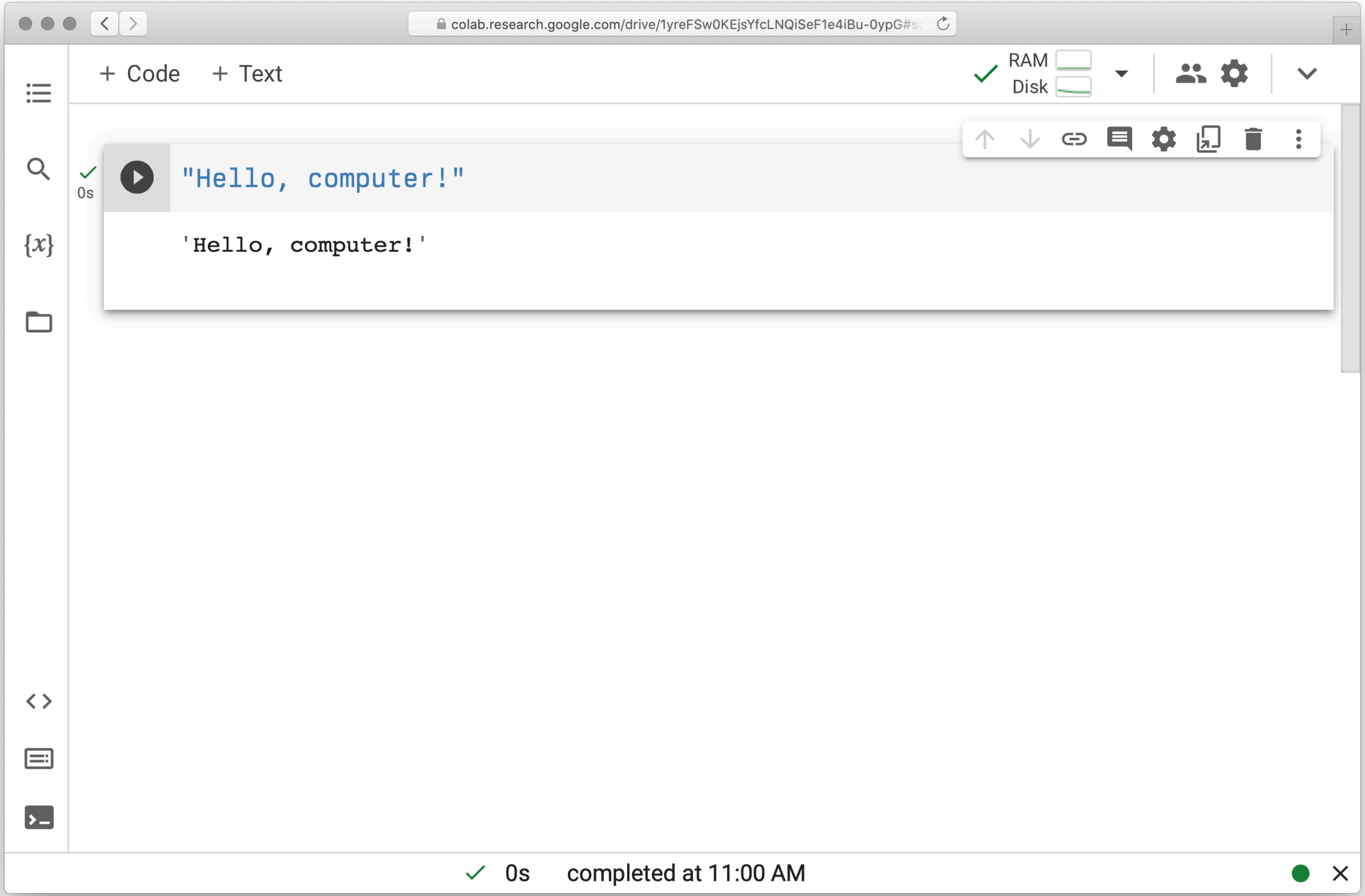
CMPU 101 § 52

Computer Science I

1 May 2023

What have we been doing this semester?

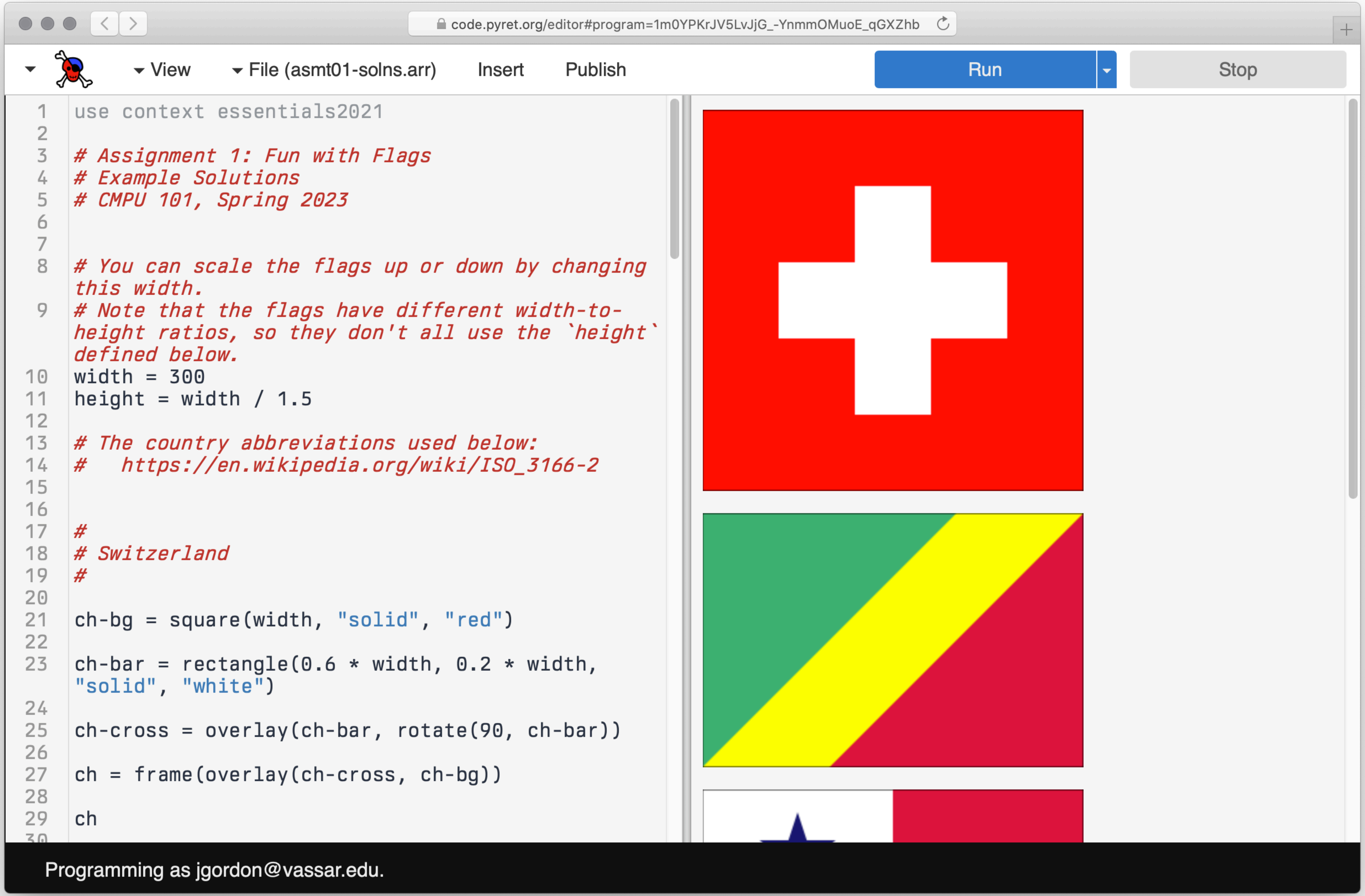

code.pyret.org/editor
View
File
Insert
Run
Stop
1 use context essentials2021
2
3 "Hello, computer!"
"Hello, computer!"
›››
Programming as jgordon@vassar.edu.


colab.research.google.com/drive/1yreFSw0KEjsYfcLNQiSeF1e4iBu-0ypG#s
+ Code
+ Text
RAM
Disk
0s
"Hello, computer!"
'Hello, computer!'
0s
completed at 11:00 AM

We're not especially interested in Pyret – ***or*** Python!

If you're programming 20 years from now, it will be in a different language, using different tools.

What have we been *doing* in these languages?


code.pyret.org/editor#program=1m0YPKrJV5LvJjG_-YnmmOMuoE_qGXZhb
View
File (asmt01-solns.arr)
Insert
Publish
Run
Stop
use context essentials2021

Assignment 1: Fun with Flags
Example Solutions
CMPU 101, Spring 2023

# You can scale the flags up or down by changing this width.
# Note that the flags have different width-to-height ratios, so they don't all use the `height` defined below.
width = 300
height = width / 1.5

The country abbreviations used below:
https://en.wikipedia.org/wiki/ISO_3166-2

#
Switzerland
#

ch-bg = square(width, "solid", "red")

ch-bar = rectangle(0.6 * width, 0.2 * width, "solid", "white")

ch-cross = overlay(ch-bar, rotate(90, ch-bar))

ch = frame(overlay(ch-cross, ch-bg))

ch
Programming as jgordon@vassar.edu.

• • •

colab.research.google.com/drive/10eGzKuAay5a0Gj6bzJCHQvGv2_HssXh

2023-04-05.ipynb

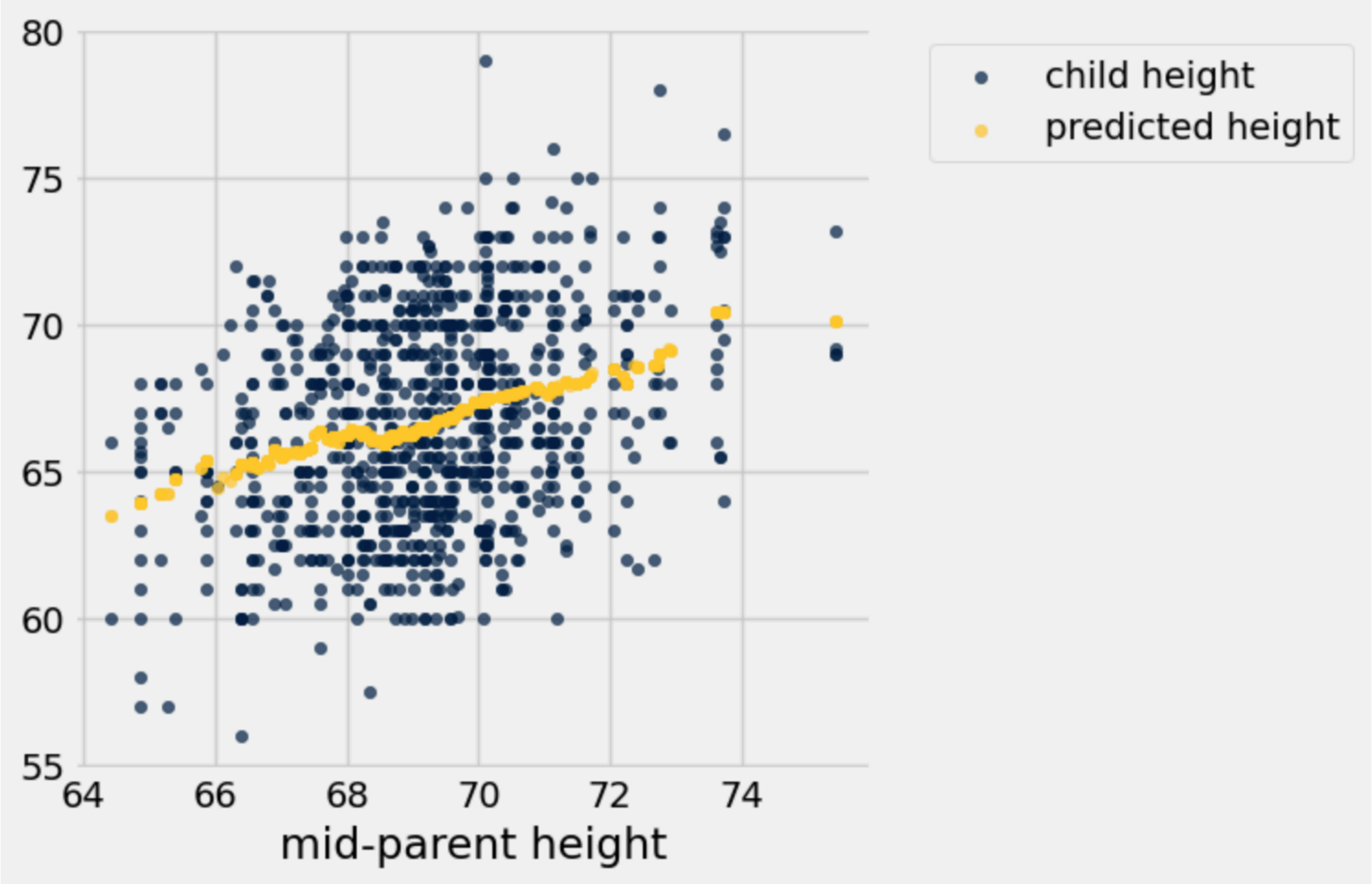
File Edit View Insert Runtime Tools Help Last edited on April 5

Comment Share

+ Code + Text

Connect

```
[ ] to_plot = galton_with_predicted.select(
        "mid-parent height", "child height", "predicted height"
    )
    to_plot.scatter("mid-parent height")
```



colab.research.google.com/drive/16h72DiJdqIm104Vyq0alHSQbYDNRXo3p

lab11 solns.ipynb

File Edit View Insert Runtime Tools Help Last saved at April 27

Comment Share

+ Code + Text

Connect

```
[ ] print(json.dumps(water_lilies_info, indent=4, ensure_ascii=False))
```


```
    "Claude Monet": [
        {
            "title": "Bridge over a Pond of Water Lilies",
            "year": "1899"
        },
        {
            "title": "Water Lilies",
            "year": "1916-19"
        },
        {
            "title": "Water Lilies",
            "year": "1919"
        }
    ],
    "Tiffany Studios": [
        {
            "title": "Cream pitcher with water lilies",
            "year": "ca. 1904-09"
        }
    ],
    "Unknown": [
        {
            "title": "Five Lotus Buds from the Mummy of Prince Amenemhat",
            "year": "ca. 1070-945 B.C."
        },
        {
            "title": "Cosmetic Spoon in the Shape of a Tilapia with Water Lilies",
            "year": "ca. 1550-1295 B.C."
        },
```

We’ve been practicing *computational thinking*.

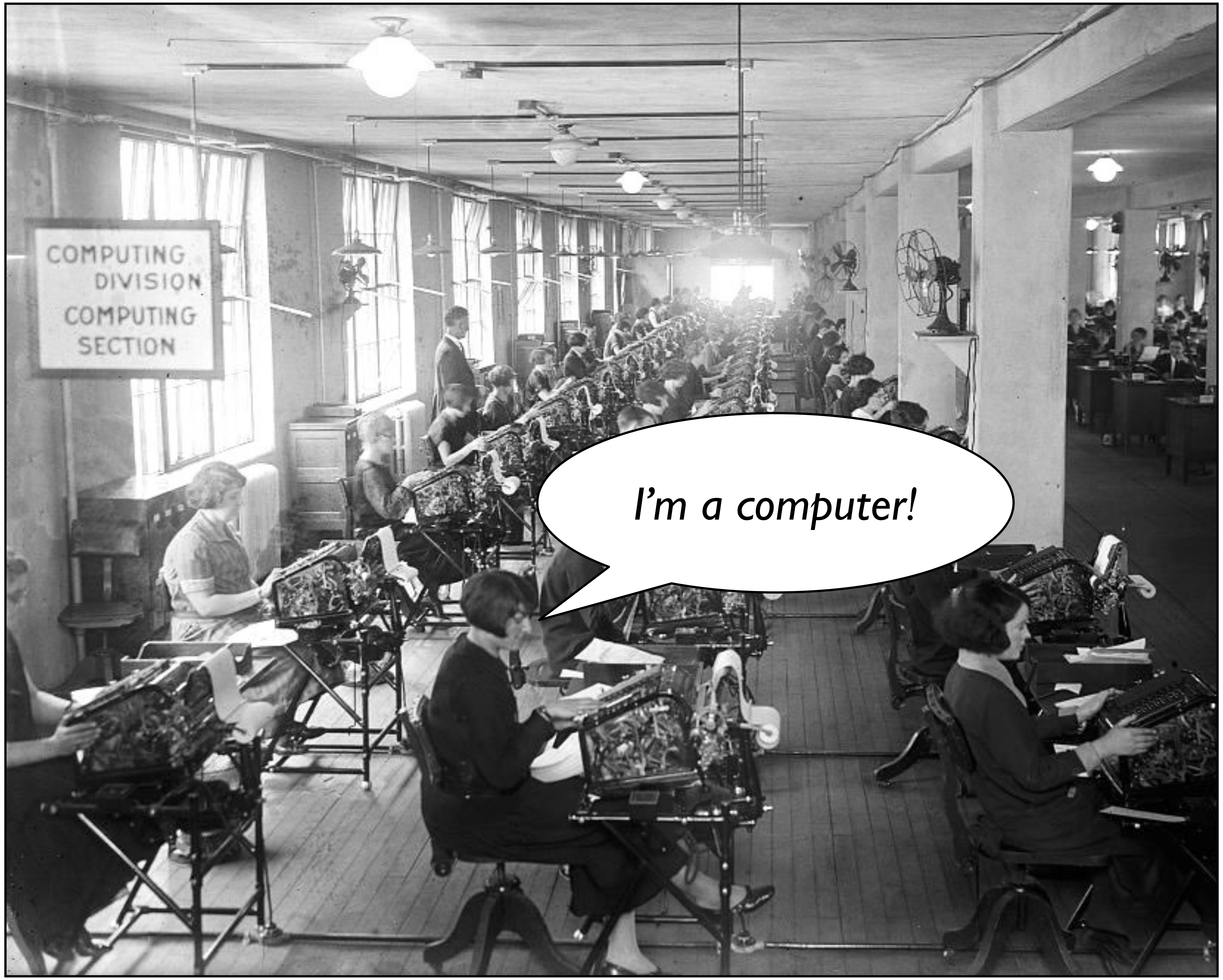
"Modern computer science is the last 1 percent of the historical timeline of computational thinking. Computer scientists inherited and then perfected computational thinking from a long line of mathematicians, natural philosophers, scientists, and engineers all interested in performing large calculations and complex inferences without error."

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

Origins of computational thinking

Before the modern computer age, there was a profession of mathematically trained experts who performed complex calculations as teams.

They were called “computers”.



Bonus Bureau, Computing Division, 1924, loc.gov/pictures/item/2016838906

Teams of human computers engaged in computational thinking long before the invention of electronic computers.

Early computational thinking can be seen going back to the records of the Babylonians, who wrote down general procedures for solving mathematical problems starting around 1800 BCE.

Long before this class, you learned these kind of computational methods.

```
  27182818284590
+31415926535897
```

```
 27182818284590
+31415926535897
---------------
              7
```

```
             1
 27182818284590
+31415926535897
---------------
             87
```

```
          1 1
 271828182845 90
+314159265358 97
----------------
              487
```

```
            1 1 1
  27182818284590
 +31415926535897
 ---------------
            0487
```

```
      1 1 1 1
 27182818284590
+31415926535897
---------------
          20487
```

```
       1 1 1 1
 27182818284590
+31415926535897
---------------
       820487
```

```
       1   1 1 1 1
 27182818284590
+31415926535897
---------------
       4820487
```

```
       1 1111
  27182818284590
+31415926535897
--------------------
        44820487
```

$$
\begin{array}{r}
1\;\;\;1\;\;\;1\,1\,1\,1\;\; \\
2718281828459 0 \\
+31415926535897 \\
\hline
744820487
\end{array}
$$

```
    1   1   1 1 1 1
 27182818284590
+31415926535897
---------------
     8744820487
```

```
     1 1 1111
 27182818284590
+31415926535897
---------------
    98744820487
```

```
     1 1 1111
 27182818284590
+31415926535897
---------------
    598744820487
```

```
     1 1 1111
 27182818284590
+31415926535897
---------------
  8598744820487
```

```
     1 1 1111
 27182818284590
+31415926535897
---------------
 58598744820487
```

```
     1 1 1111
 27182818284590
+31415926535897
---------------
 58598744820487
```

Euclid's algorithm

Around 300 BCE, the Greek mathematician Euclid gave a method to find the *greatest common divisor* (GCD) of two numbers, which is the largest integer that divides both numbers.

Euclid’s algorithm

Euclid noticed that the GCD of two numbers divides their difference.

So, he repeatedly replaced the larger number with their difference until both were the same.

Euclid's algorithm

Euclid noticed that the GCD of two numbers divides their difference.

So, he repeatedly replaced the larger number with their difference until both were the same.

```
  gcd(48, 18)
→ gcd(30, 18)
→ gcd(12, 18)
→ gcd(12,  6)
→ gcd( 6,  6)
→ 6
```

Sieve of Eratosthenes

This is another famous method dating back to the ancient Greeks, used to find all the prime numbers up to some limit.

Sieve of Eratosthenes

2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18

We begin with a list of all the integers, from 2 to the limit.

Sieve of Eratosthenes

2 3 ~~4~~ 5 ~~6~~ 7 ~~8~~ 9 ~~10~~ 11 ~~12~~ 13 ~~14~~ 15 ~~16~~ 17 ~~18~~

We cross out all the multiples of 2.

Sieve of Eratosthenes

2 **3** ~~4~~ 5 ~~6~~ 7 ~~8~~ ~~9~~ ~~10~~ 11 ~~12~~ 13 ~~14~~ ~~15~~ ~~16~~ 17 ~~18~~

We cross out all the multiples of 2.

Then all the multiples of 3.

Sieve of Eratosthenes

2 3 ~~4~~ **5** ~~6~~ 7 ~~8~~ ~~9~~ ~~10~~ 11 ~~12~~ 13 ~~14~~ ~~15~~ ~~16~~ 17 ~~18~~

We cross out all the multiples of 2.

Then all the multiples of 3. And 5.

Sieve of Eratosthenes

2 3 ~~4~~ 5 ~~6~~ 7 ~~8~~ ~~9~~ ~~10~~ 11 ~~12~~ 13 ~~14~~ ~~15~~ ~~16~~ 17 ~~18~~

We cross out all the multiples of 2.

Then all the multiples of 3. And 5.

And so on, leaving you with only the primes between 2 and the limit you chose.

Sieve of Eratosthenes

After each round of elimination, a new prime will be revealed, and the next round crosses out all its multiples.

These are computational procedures, carried out by hand!

Programmable computers

No matter how simple and unambiguous the steps are made, human computers make mistakes – and lots of them!

So, inventors through the ages have sought to make computing *machines* to allow people to perform longer computations with fewer errors.

This was a slow process, taking us from…

Slide rule

c. 1620

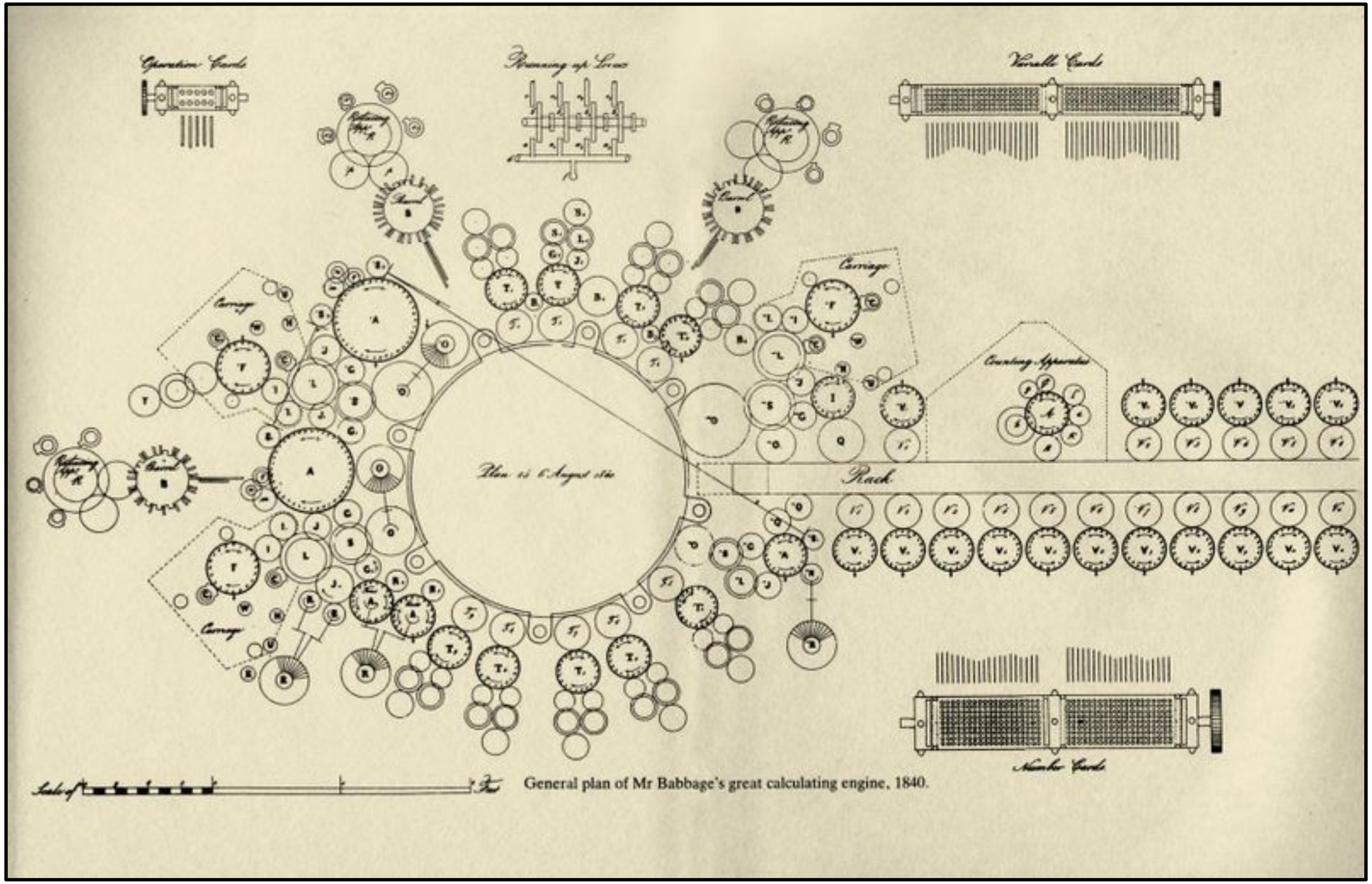
Blaise Pascal's mechanical calculator

1642

Precursors to the idea of a *programmable* computer originated well before the electronic computing age.

In the early 1700s, French textile weavers experimented with machines that could weave complex patterns using an automatic loom.

One of the more well known is the Jacquard loom, which was controlled by long chains of punched cards.



Plan for Babbage's Analytical Engine

1840

Babbage collaborated with a gifted mathematician, Ada Lovelace, who designed algorithms for the Analytical Engine, even though there was no machine to run them on.

Lovelace saw the Analytical Engine not as a mere calculator but as a processor of *any* information that could be encoded in symbols.

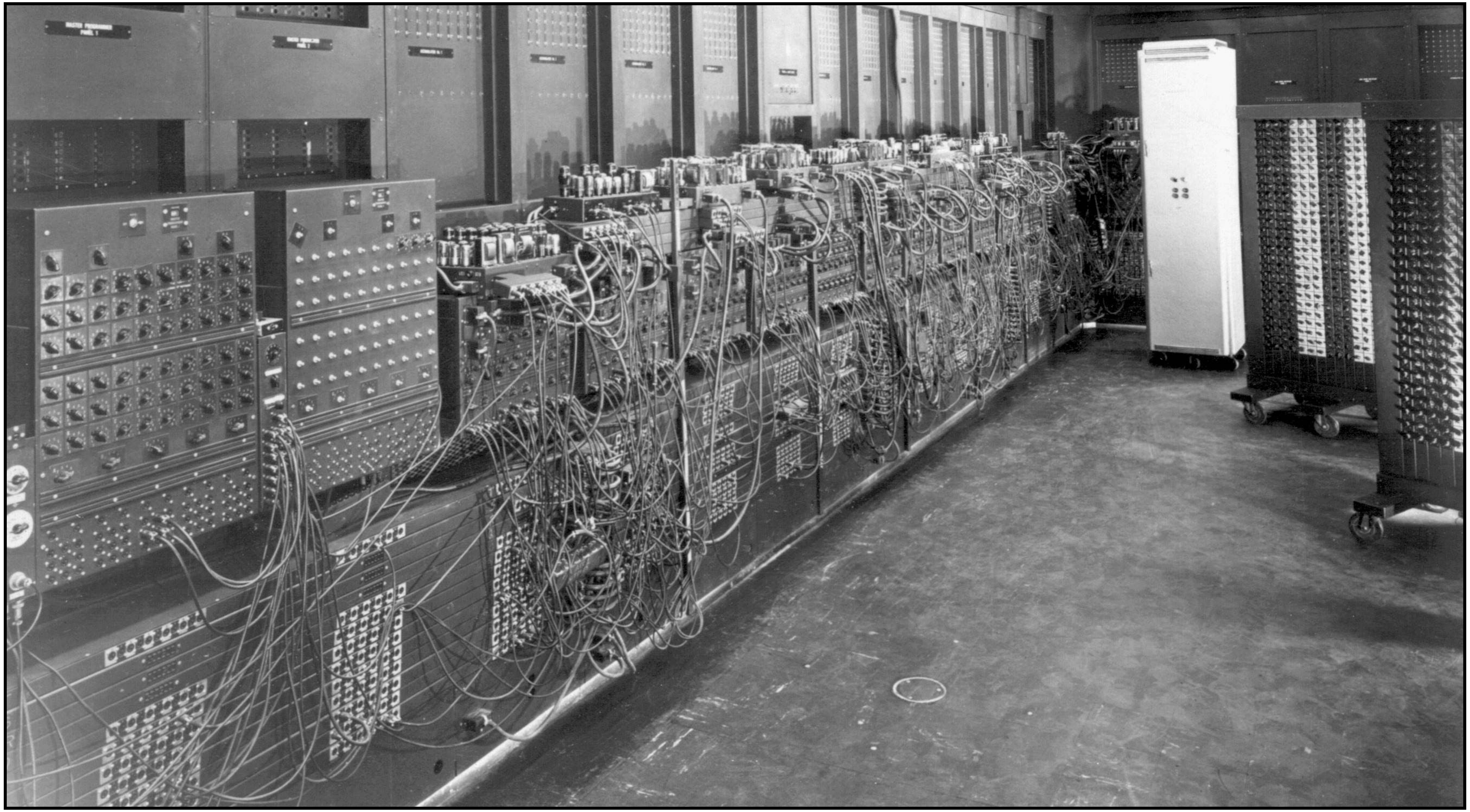
This insight, that computing programs can calculate not only over numbers but over symbols that can stand for anything in the world, anticipated by a hundred years a key tenet of the modern computer age.

Lovelace saw the computer as an *information machine*.

While Babbage's designs for a programmable computer weren't realized at the time, the age of electronics opened new possibilities.

Harvard Mark I

1944



ENIAC

c. 1945

Early computers were very difficult to program, working in languages that were closely tied to the hardware.

Grace Hopper ’28 popularized the idea of a compiler for machine-independent programming languages and defined FLOW-MATIC, the first English-like data processing language in the early 1950s.

Those ideas were later folded into the popular COBOL language (1959).

Since the 1950s, many programming languages have been defined, experienced popularity, and then been supplanted by new designs.

Today, Python is the programming language most often used for work in data science and artificial intelligence.

Programming no longer involves plugging in wires or punching cards, but it's still hard!

“*The programmer, like the poet, works only slightly removed from pure thought-stuff.* He builds castles in the air, from air, creating by exertion of the imagination…

Frederick Brooks, *The Mythical Man-Month*, 1975

"Few media of creation are so flexible, so easy to polish and rework, so readily capable of realizing grand conceptual structures. *Yet the program construct, unlike the poet's words, is real in the sense that it moves and works*, producing visible outputs separate from the construct itself…

Frederick Brooks, *The Mythical Man-Month*, 1975

"One types the correct incantation on a keyboard, and a display screen *comes to life*, showing things that never were nor could be… It prints results, draws pictures, produces sounds, moves arms. The magic of myth and legend has come true in our time…

Frederick Brooks, *The Mythical Man-Month*, 1975

"The computer resembles the magic of legend in this respect, too. If one character, one pause, of the incantation is not strictly in proper form, the magic doesn't work. Human beings are not accustomed to being perfect, and few areas of human activity demand it. *Adjusting to the requirement for perfection is, I think, the most difficult part of learning to program.*"

Frederick Brooks, *The Mythical Man-Month*, 1975

Computing with data

We’ve seen some cool datasets during this semester – and you got to explore data that was of interest to you for your mini-projects – but there are many, many more datasets you can explore.

www.data-is-plural.com

Data Is Plural

... is a weekly newsletter (and seasonal podcast) of useful/curious datasets, published by Jeremy Singer-Vine. There have been 331 editions, dating from October 21, 2015 to April 26, 2023. To receive future editions, sign up here:

Enter your email address | Subscribe

Some Nice Things People Have Said

- *“a treasure trove of interesting datasets”* — Julia Silge
- *“delivers exactly what it promises, it's delightful”* — Simon Willison
- *“required reading for anyone interested in data journalism”* — Julia Angwin
- *“consistently fascinating”* — Robin Sloan
- *“the only newsletter I open immediately”* — Paul Ford
- *“the newsletter I forward most frequently to colleagues who like datasets”* — Melody Joy Kramer
- *“definitely my favorite newsletter about data and possibly my favorite newsletter period”* — Melanie Walsh
- *“the best data newsletter ever”* — François Briatte

Recent Editions

2023.04.26 • Drinking water violations, childcare prices, kinship terms, art history allocations, and “the oldest experimental crop field in America.”

2023.04.19 • Municipal zoning rules, AI incidents, state bill trajectories, rare-earth mining projects, and “pirate radio” enforcement.

2023.04.12 • Jail rosters, sanctions enforcement, border surveillance, flash flooding in urban England, and Dutch textile shipments.

data-is-plural.com

github.com/public-apis/public-apis

README.md

Public APIs

A collective list of free APIs for use in software and web development

Status

Number of Categories 51 | Number of APIs 1425

Tests of push & pull failing | Validate links failing | Tests of validate package passing

The Project

Contributing Guide • API for this project • Issues • Pull Requests • License

Alternative sites for the project (unofficials)

Free APIs • Dev Resources • Public APIs Site • Apihouse • Collective APIs

---

[**Become a sponsor** and support Public APIs and their maintainers]

Special thanks to:

APILayer

The fastest way to integrate APIs into any product

Explore, discover and consume public APIs as simpler programmable building blocks all on one platform for a 10x developer experience.

Report repository

Used by 1

@KasperiP / **KasperiP**

Contributors 1,266

+ 1,255 contributors

Languages

Python 96.8% **Shell** 3.2%

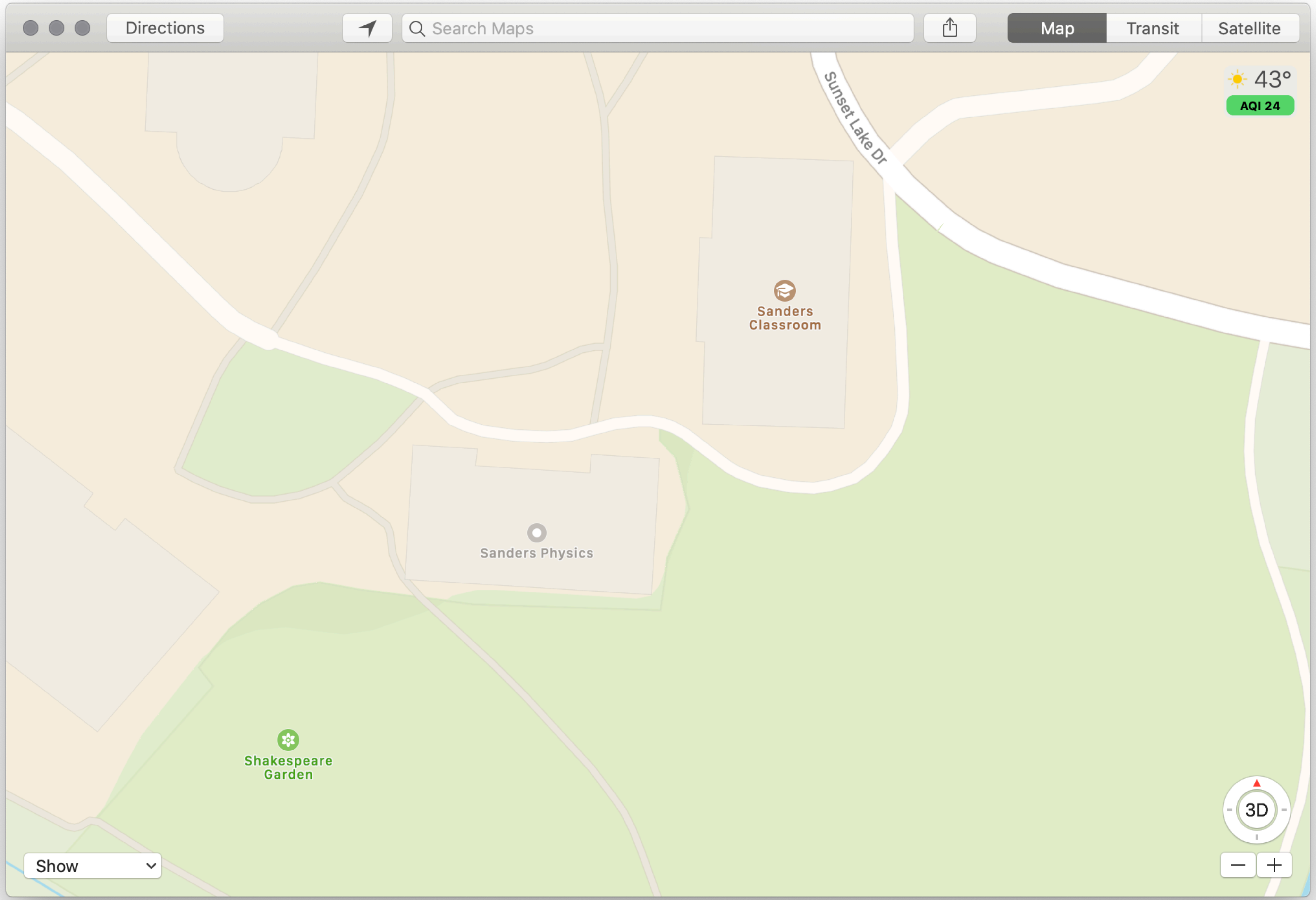
github.com/public-apis/public-apis

Building data models

Data is situated in the environment where it was gathered.

Consider Galton's child-height data.

- He gathered the data in England c. 1886.
- What would happen if you tried to use it to predict heights in Poughkeepsie today? In Guatemala? In China?



When we collect data, it's like making a map:

We're constructing a model, where we choose what to represent, and how to represent it.

"…most of the data and data models we have inherited deal with structures of power, like gender and race, with a crudeness that would never pass muster in a peer-reviewed humanities publication."

Miriam Posner, "What's Next: The Radical, Unrealized Potential of Digital Humanities", 2016 colab.research.google.com/drive/10eGzKuAay5a0Gj6bzJCHQvGv2_HssXhD

2023-04-05.ipynb

File Edit View Insert Runtime Tools Help Last edited on April 5

Comment Share

+ Code + Text

Connect

```
galton
```

family num	father height	mother height	children	child num	gender	child height
1	78.5	67	4	1	male	73.2
1	78.5	67	4	2	female	69.2
1	78.5	67	4	3	female	69
1	78.5	67	4	4	female	69
2	75.5	66.5	4	1	male	73.5
2	75.5	66.5	4	2	male	72.5
2	75.5	66.5	4	3	female	65.5
2	75.5	66.5	4	4	female	65.5
3	75	64	2	1	male	71
3	75	64	2	2	female	68

... (924 rows omitted)

Family size

We noticed that some of these Victorian families were *big*, so we decided to check the distribution of the number of children.

We made a histogram:

```
[ ] galton.hist("children")
```

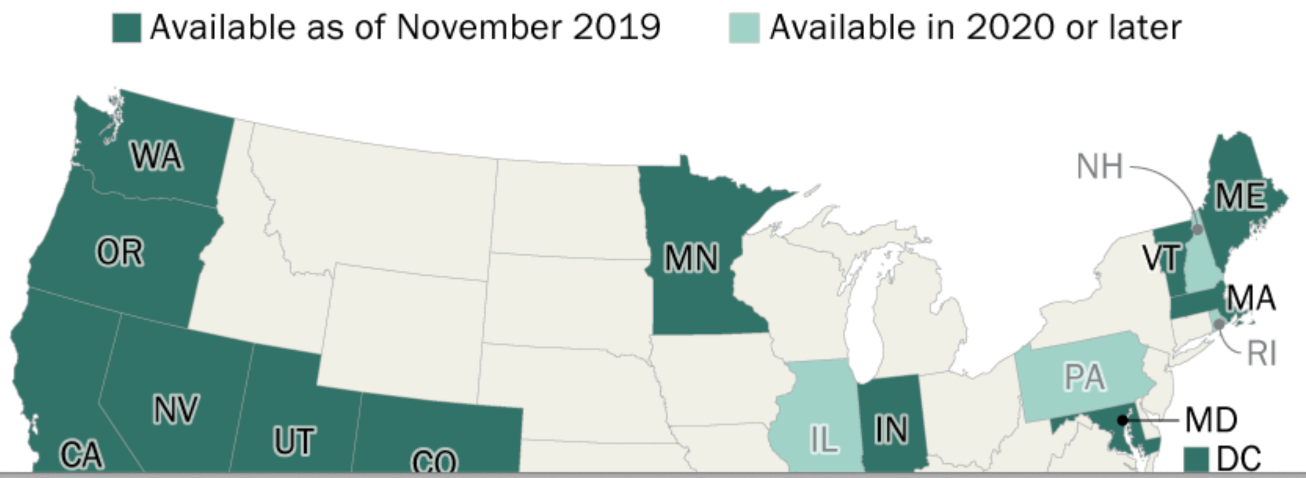
DECEMBER 18, 2019

About four-in-ten U.S. adults say forms should offer more than two gender options

BY NIKKI GRAF

In 2019, at least seven states have started offering a third gender option on driver's licenses for people who don't identify as male or female, and at least four more plan to do so in 2020. A number of states have also added a third gender option on birth

States that offer a third gender option on driver's licenses



RELATED

SHORT READ | JUL 27, 2021

Rising shares of U.S. adults know someone who is transgender or goes by gender-neutral pronouns

SHORT READ | DEC 18, 2019

About four-in-ten U.S. adults say forms should offer mo options

SHORT READ | SEP 5

About one-in-fiv someone who go pronoun

"I want us to be more ambitious, to hold ourselves to much higher standards when we are claiming to develop data-based work that depicts people's lives."

Miriam Posner, "What's Next: The Radical, Unrealized Potential of Digital Humanities", 2016

Data and privacy

Not Secure — help.sentiment140.com

Sentiment140

Search this site

- General Information
- Site Functionality
- For Academics
- API
- Sentiment Analysis Sites
- Contact Us

Return to Sentiment140

General Information

What is Sentiment140?

Sentiment140 allows you to discover the sentiment of a brand, product, or topic on Twitter.

How does this work?

You can read about our approach in our technical report: Twitter Sentiment Classification using Distant Supervision. There are also additional features that are not described in this paper.

How is this different?

Our approach is different from other sentiment analysis sites because:

- We use classifiers built from machine learning algorithms. Other products use a simpler keyword-based approach which may have higher precision but lower recall.
- We provide transparency for the classification results of individual tweets. Other sites only surface aggregated metrics, which makes it difficult to assess the accuracy of their classifiers.

Who created this?

Sentiment140 was created by Alec Go, Richa Bhayani, and Lei Huang, who were Computer Science graduate students at Stanford University.

What are the use cases?

1. Brand management (e.g. windows 10)
2. Polling (e.g. obama)
3. Planning a purchase (e.g. kindle)

Can you help me?

sentiment140.com

testdata.manual.2009.06.14 — Edited

150%

Sheet 1

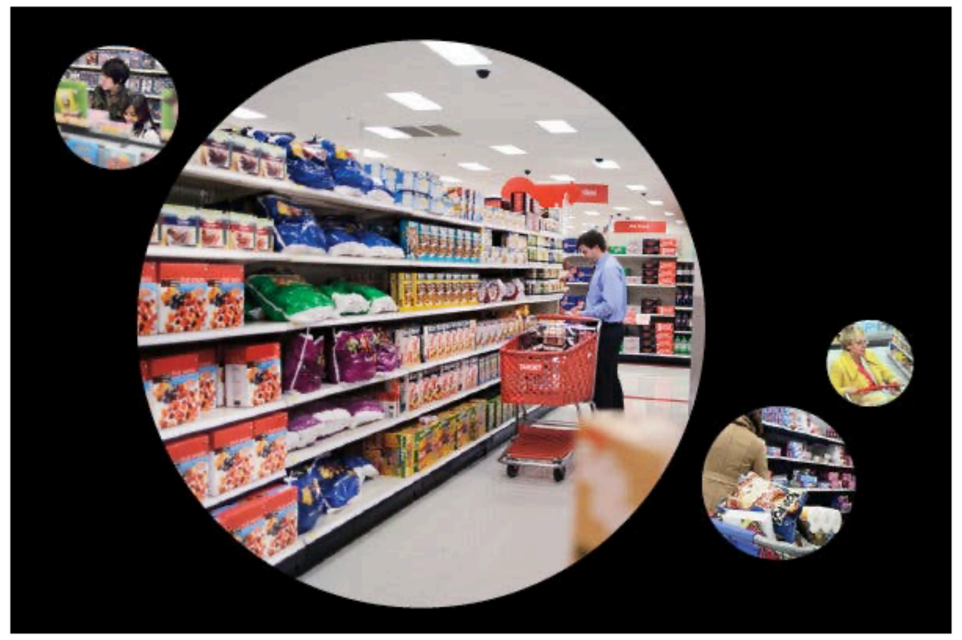
	A	B	C	D	E	F
37	0	122	Sat May 16 22:42:25 UTC 2009	itchy	robloposky	I'm itchy and miserable!
38	0	123	Sat May 16 22:42:44 UTC 2009	itchy	EdwinLValencia	@sekseemess no. I'm not itchy for now. Maybe later, lol.
39	4	124	Sat May 16 23:48:15 UTC 2009	stanford	imusicmash	RT @jessverr I love the nerdy Stanford human biology videos - makes me miss school. http://bit.ly/13t7NR
40	4	125	Sat May 16 23:58:34 UTC 2009	lyx	drewloewe	@spinuzzi: Has been a bit crazy, with steep learning curve, but LyX is really good for long docs. For anything shorter, it would be insane.
41	4	131	Sun May 17 15:05:03 UTC 2009	Danny Gokey	VickyTigger	I'm listening to "P.Y.T" by Danny Gokey <3 <3 <3 Aww, he's so amazing. I <3 him so much :)
42	4	132	Sun May 17 17:27:45 UTC 2009	sleep	babblyabbie	is going to sleep then on a bike ride:]
43	0	133	Sun May 17 17:27:49 UTC 2009	sleep	kisjoaquin	cant sleep... my tooth is aching.
44	0	134	Sun May 17 17:28:02 UTC 2009	sleep	Whacktackular	Blah, blah, blah same old same old. No plans today, going back to sleep I guess.
45	0	135	Sun May 17 17:29:50 UTC 2009	san francisco	Adrigonzo	glad i didnt do Bay to Breakers today, it's 1000 freaking degrees in San Francisco wtf
46	2	136	Sun May 17 17:30:19 UTC 2009	san francisco	sulu34	is in San Francisco at Bay to Breakers.
47	2	137	Sun May 17 17:30:23 UTC 2009	san francisco	schuyler	just landed at San Francisco
48	2	138	Sun May 17 17:30:56 UTC 2009	san francisco	MattBragoni	San Francisco today. Any suggestions?
49	0	139	Sun May 17 17:32:00 UTC 2009	aig	KennyTRoland	?Obama Administration Must Stop Bonuses to AIG Ponzi Schemers ... http://bit.ly/2CUIg
50	0	140	Sun May 17 17:32:30 UTC 2009	aig	aMild	started to think that Citi is in really deep s&^t. Are they gonna survive the turmoil or are they gonna be the next AIG?
51	0	141	Sun May 17 17:32:36 UTC 2009	aig	Trazor1	ShaunWoo hate'n on AiG
52	4	142	Sun May 17 17:35:17 UTC 2009	star trek	mimknits	@YarnThing you will not regret going to see Star Trek. It was AWESOME!
53	2	143	Sun May 17 17:35:28 UTC 2009	star trek	GeeRen	On my way to see Star Trek @ The Esquire.
54	2	144	Sun May 17 17:35:45 UTC 2009	star trek	checkyesjess	Going to see star trek soon with my dad.
55	0	145	Mon May 18 01:13:27 UTC 2009	Malcolm Gladwell	renano	annoying new trend on the internets: people picking apart michael lewis and malcolm gladwell.

SUM 122 | AVERAGE 61 | MIN 0 | MAX 122 | COUNTA 6

"The words and phrases we search for on Google, the times of day we are most active on Facebook, and the number of items we add to our Amazon carts are all tracked and stored as data – data that are then converted into corporate financial gain."

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

How Companies Learn Your Secrets



Antonio Bolfo/Reportage for The New York Times

By Charles Duhigg

Feb. 16, 2012

Andrew Pole had just started working as a statistician for Target in 2002, when two colleagues from the marketing department stopped by his desk to ask an odd question: "If we wanted to figure out if a customer is pregnant, even if she didn't want us to know, can you do that? "

"As Pole's computers crawled through the data, he was able to identify about 25 products that, when analyzed together, allowed him to *assign each shopper a 'pregnancy prediction' score*. More important, he could also *estimate her due date* to within a small window, so Target could send coupons timed to very specific stages of her pregnancy.

"One Target employee I spoke to provided a hypothetical example. Take a fictional Target shopper named Jenny Ward, who is 23, lives in Atlanta and in March bought cocoa-butter lotion, a purse large enough to double as a diaper bag, zinc and magnesium supplements and a bright blue rug. There's, say, an 87 percent chance that she's pregnant and that her delivery date is sometime in late August."

Kashmir Hill, "How Target Figured Out a Teen Girl was Pregnant Before Her Father Did", *Forbes*, 2012

Computing with data is complex, and it's not just "technical" issues we need to concern ourselves with!

Computing with data gives us a lot of power!

We can do a lot of harm, but we can also do a lot of good!

Computer Science I
—or, where do you go from here?

Data types

Naming values

Evaluation

Conditionals (**if** and **cases**)

Function signatures

Testing functions

Tabular data

Higher-order functions

Lambda expressions

Linked lists

Defining structured data

Structurally recursive data and functions

Trees (binary, *n*-ary)

Reactive programs

List and string slicing

List comprehensions

Numpy arrays

Sanitizing real-world data

Visualization

Dataclasses

Side effects and mutation

Functional vs imperative languages

Iteration (**for**)

Accumulators

Debugging using **print** statements

Memory and aliasing

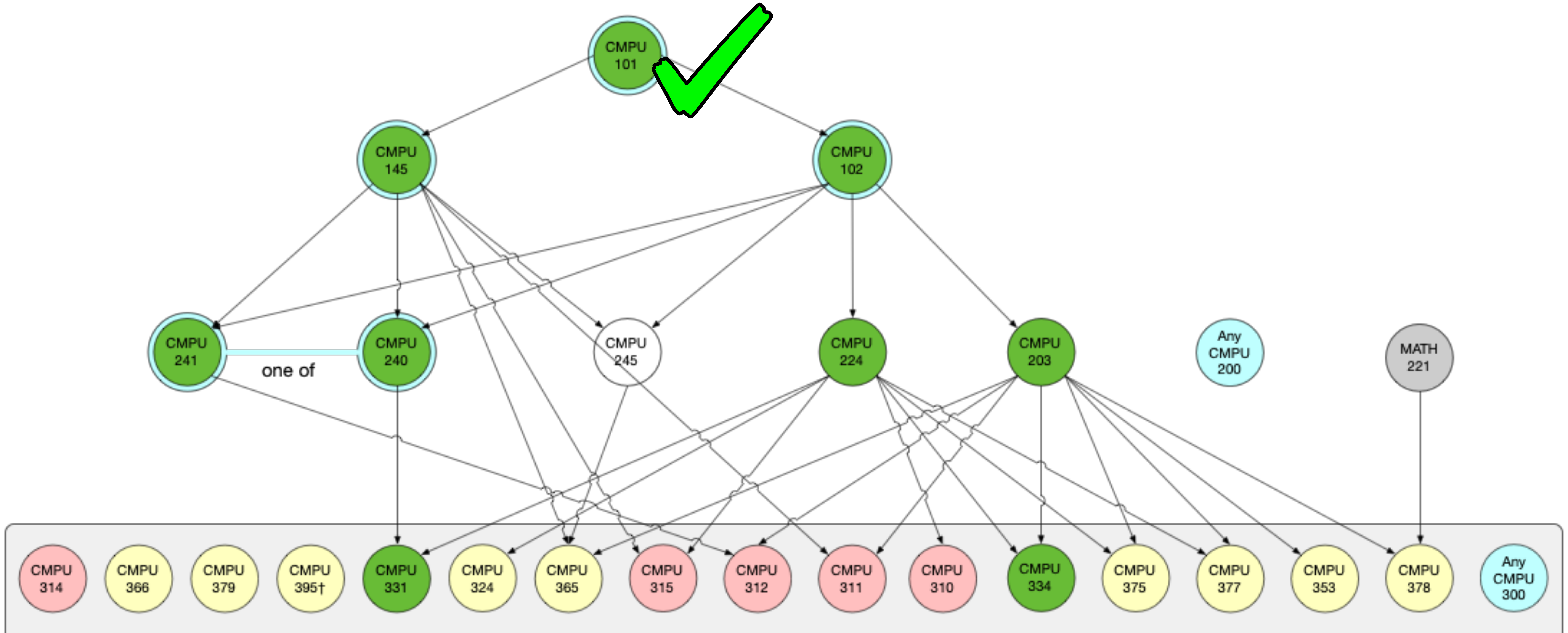
Dictionaries (hash tables)

Web APIs and JSON

Congratulations on making it this far!

CS courses at Vassar

Vassar Computer Science Course Map



*At least two CMPU-200 level courses required for every CMPU-300 level course †Prerequisites vary depending on topic

Major-required courses

CMPU 101 - Computer Science I: Problem-Solving and Abstraction
CMPU 102 - Computer Science II: Data Structures and Algorithms
CMPU 145 - Foundations of Computer Science
CMPU 203 - Computer Science III: Software Design and Implementation
CMPU 224 - Computer Organization
CMPU 240 - Theory of Computation
CMPU 241 - Analysis of Algorithms
CMPU 331 - Compilers
CMPU 334 - Operating Systems

300-level electives (at least one for major)

CMPU 324 - Computer Architecture
CMPU 353 - Bioinformatics
CMPU 365 - Artificial Intelligence
CMPU 366 - Computational Linguistics
CMPU 375 - Computer Networks
CMPU 377 - Parallel Programming
CMPU 378 - Graphics
CMPU 379 - Computer Animation: Art, Science and Criticism
CMPU 395 - Advanced Special Topics

Intensives (at least one for major)

CMPU 310 - Topics in Virtualization
CMPU 311 - Database Systems
CMPU 312 - Applications of Artificial Intelligence
CMPU 314 - Projects in Digital Media Production
CMPU 315 - Computer Security

Correlate-required courses

CMPU 101 - Computer Science I: Problem-Solving and Abstraction
CMPU 102 - Computer Science II: Data Structures and Algorithms
CMPU 145 - Foundations of Computer Science
CMPU 240 or 241 - Theory of Computation or Analysis of Algorithms
CMPU 2xx - Any other 200-level course
CMPU 3xx - Any 300-level course

200-level electives (not required for major)

CMPU 245 - Declarative Programming Models

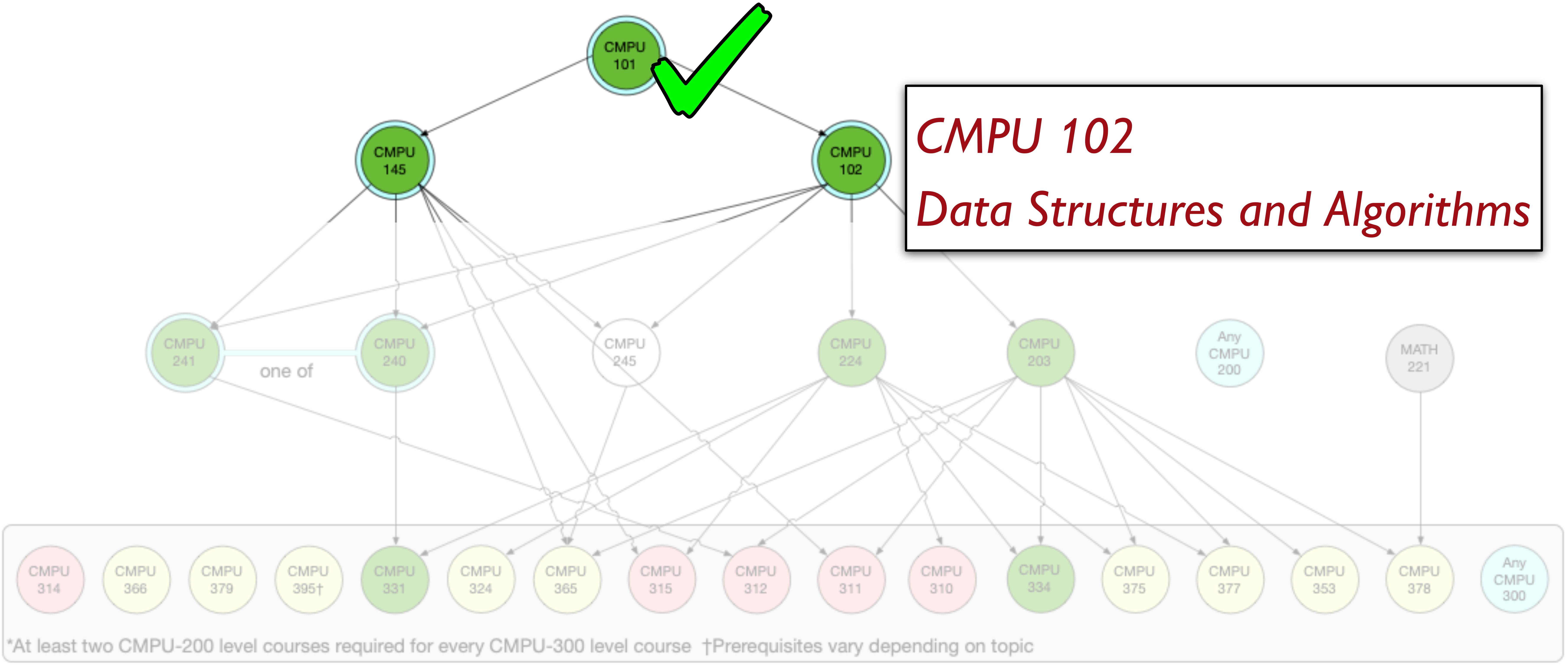
Extra-departmental

MATH 221 - Linear Algebra

Vassar Computer Science Course Map

CMPU 102

Data Structures and Algorithms



Major-required courses

CMPU 101 - Computer Science I: Problem-Solving and Abstraction
CMPU 102 - Computer Science II: Data Structures and Algorithms
CMPU 145 - Foundations of Computer Science
CMPU 203 - Computer Science III: Software Design and Implementation
CMPU 224 - Computer Organization
CMPU 240 - Theory of Computation
CMPU 241 - Analysis of Algorithms
CMPU 331 - Compilers
CMPU 334 - Operating Systems

300-level electives (at least one for major)

CMPU 324 - Computer Architecture
CMPU 353 - Bioinformatics
CMPU 365 - Artificial Intelligence
CMPU 366 - Computational Linguistics
CMPU 375 - Computer Networks
CMPU 377 - Parallel Programming
CMPU 378 - Graphics
CMPU 379 - Computer Animation: Art, Science and Criticism
CMPU 395 - Advanced Special Topics

Intensives (at least one for major)

CMPU 310 - Topics in Virtualization
CMPU 311 - Database Systems
CMPU 312 - Applications of Artificial Intelligence
CMPU 314 - Projects in Digital Media Production
CMPU 315 - Computer Security

Correlate-required courses

CMPU 101 - Computer Science I: Problem-Solving and Abstraction
CMPU 102 - Computer Science II: Data Structures and Algorithms
CMPU 145 - Foundations of Computer Science
CMPU 240 or 241 - Theory of Computation or Analysis of Algorithms
CMPU 2xx - Any other 200-level course
CMPU 3xx - Any 300-level course

200-level electives (not required for major)

CMPU 245 - Declarative Programming Models

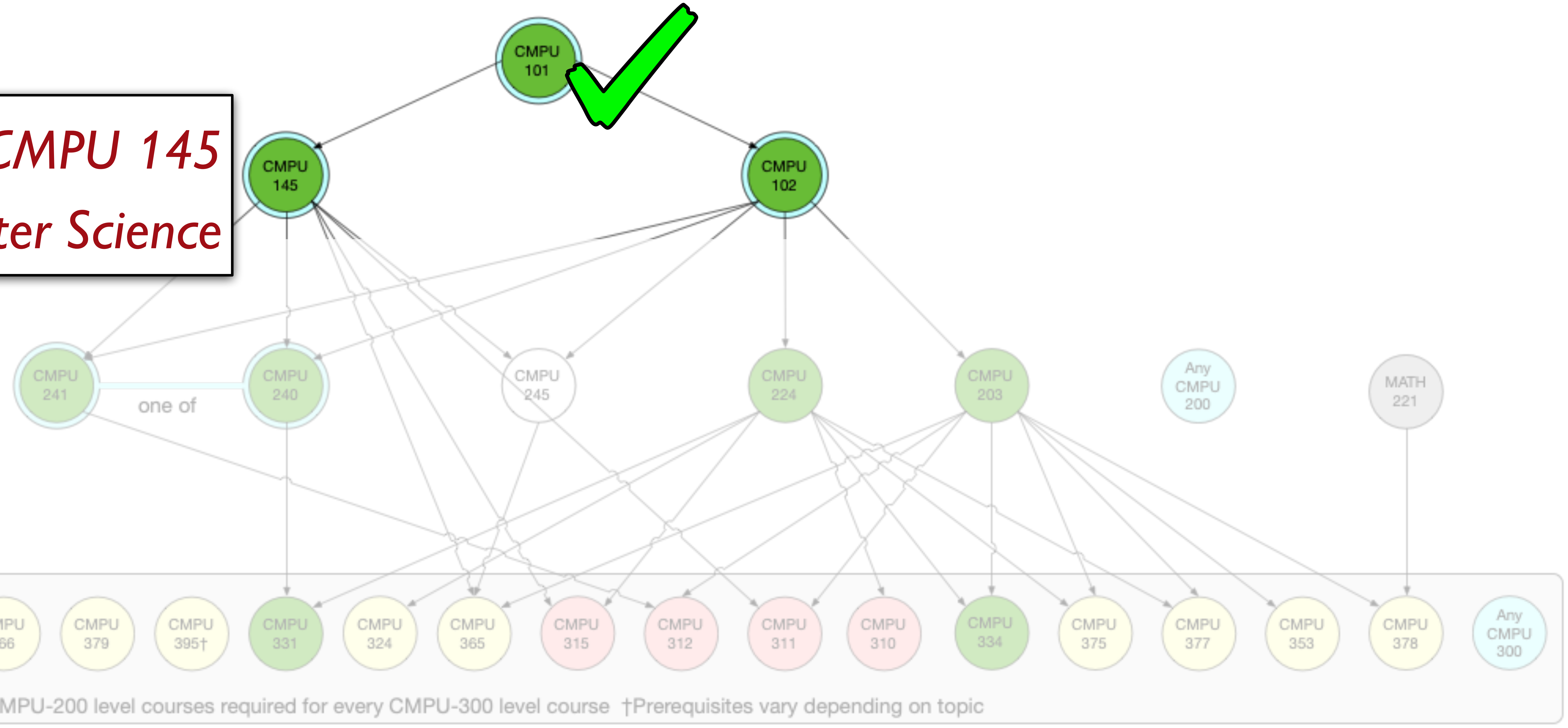
Extra-departmental

MATH 221 - Linear Algebra

Vassar Computer Science Course Map

CMPU 145

Foundations of Computer Science



Major-required courses

CMPU 101 - Computer Science I: Problem-Solving and Abstraction
CMPU 102 - Computer Science II: Data Structures and Algorithms
CMPU 145 - Foundations of Computer Science
CMPU 203 - Computer Science III: Software Design and Implementation
CMPU 224 - Computer Organization
CMPU 240 - Theory of Computation
CMPU 241 - Analysis of Algorithms
CMPU 331 - Compilers
CMPU 334 - Operating Systems

300-level electives (at least one for major)

CMPU 324 - Computer Architecture
CMPU 353 - Bioinformatics
CMPU 365 - Artificial Intelligence
CMPU 366 - Computational Linguistics
CMPU 375 - Computer Networks
CMPU 377 - Parallel Programming
CMPU 378 - Graphics
CMPU 379 - Computer Animation: Art, Science and Criticism
CMPU 395 - Advanced Special Topics

Intensives (at least one for major)

CMPU 310 - Topics in Virtualization
CMPU 311 - Database Systems
CMPU 312 - Applications of Artificial Intelligence
CMPU 314 - Projects in Digital Media Production
CMPU 315 - Computer Security

Correlate-required courses

CMPU 101 - Computer Science I: Problem-Solving and Abstraction
CMPU 102 - Computer Science II: Data Structures and Algorithms
CMPU 145 - Foundations of Computer Science
CMPU 240 or 241 - Theory of Computation or Analysis of Algorithms
CMPU 2xx - Any other 200-level course
CMPU 3xx - Any 300-level course

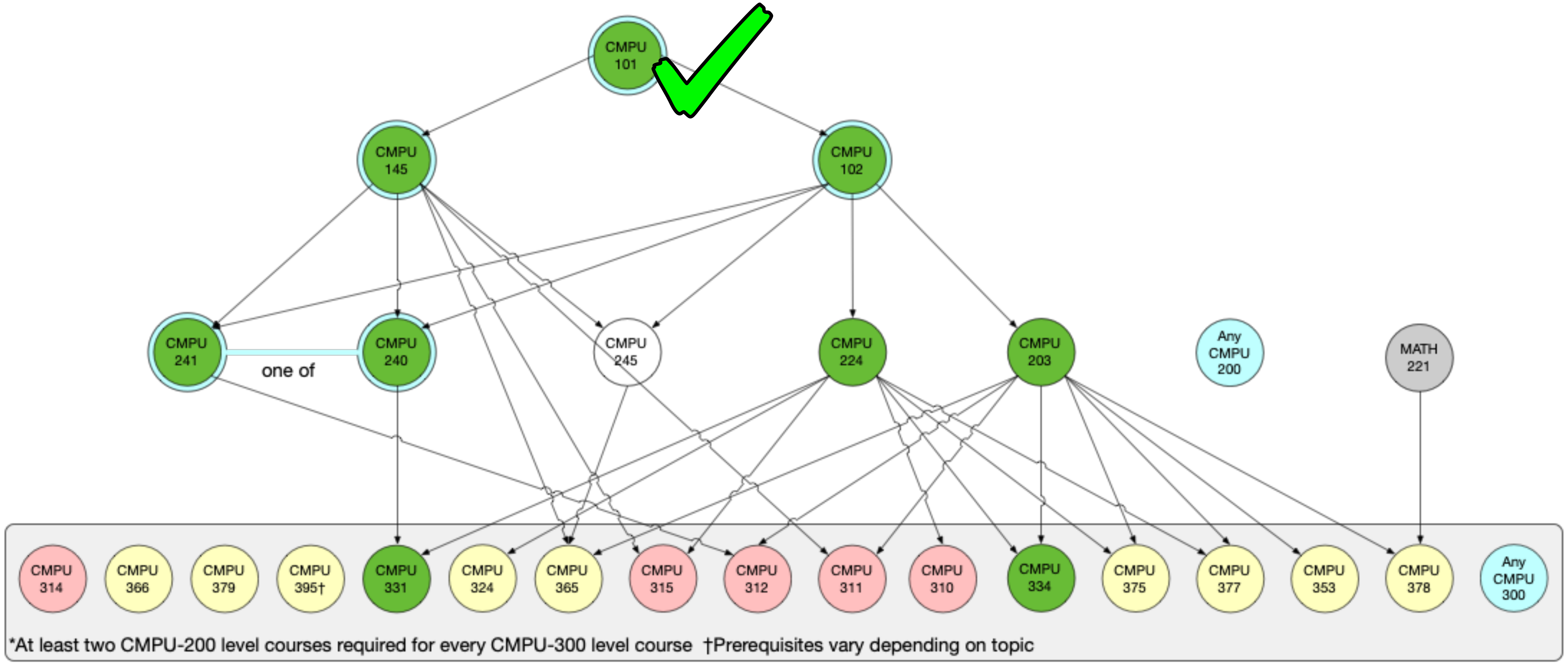
200-level electives (not required for major)

CMPU 245 - Declarative Programming Models

Extra-departmental

MATH 221 - Linear Algebra

Vassar Computer Science Course Map



Major-required courses

CMPU 101 - Computer Science I: Problem-Solving and Abstraction
CMPU 102 - Computer Science II: Data Structures and Algorithms
CMPU 145 - Foundations of Computer Science
CMPU 203 - Computer Science III: Software Design and Implementation
CMPU 224 - Computer Organization
CMPU 240 - Theory of Computation
CMPU 241 - Analysis of Algorithms
CMPU 331 - Compilers
CMPU 334 - Operating Systems

300-level electives (at least one for major)

CMPU 324 - Computer Architecture
CMPU 353 - Bioinformatics
CMPU 365 - Artificial Intelligence
CMPU 366 - Computational Linguistics
CMPU 375 - Computer Networks
CMPU 377 - Parallel Programming
CMPU 378 - Graphics
CMPU 379 - Computer Animation: Art, Science and Criticism
CMPU 395 - Advanced Special Topics

Intensives (at least one for major)

CMPU 310 - Topics in Virtualization
CMPU 311 - Database Systems
CMPU 312 - Applications of Artificial Intelligence
CMPU 314 - Projects in Digital Media Production
CMPU 315 - Computer Security

Correlate-required courses

CMPU 101 - Computer Science I: Problem-Solving and Abstraction
CMPU 102 - Computer Science II: Data Structures and Algorithms
CMPU 145 - Foundations of Computer Science
CMPU 240 or 241 - Theory of Computation or Analysis of Algorithms
CMPU 2xx - Any other 200-level course
CMPU 3xx - Any 300-level course

200-level electives (not required for major)

CMPU 245 - Declarative Programming Models

Extra-departmental

MATH 221 - Linear Algebra

Try them out!

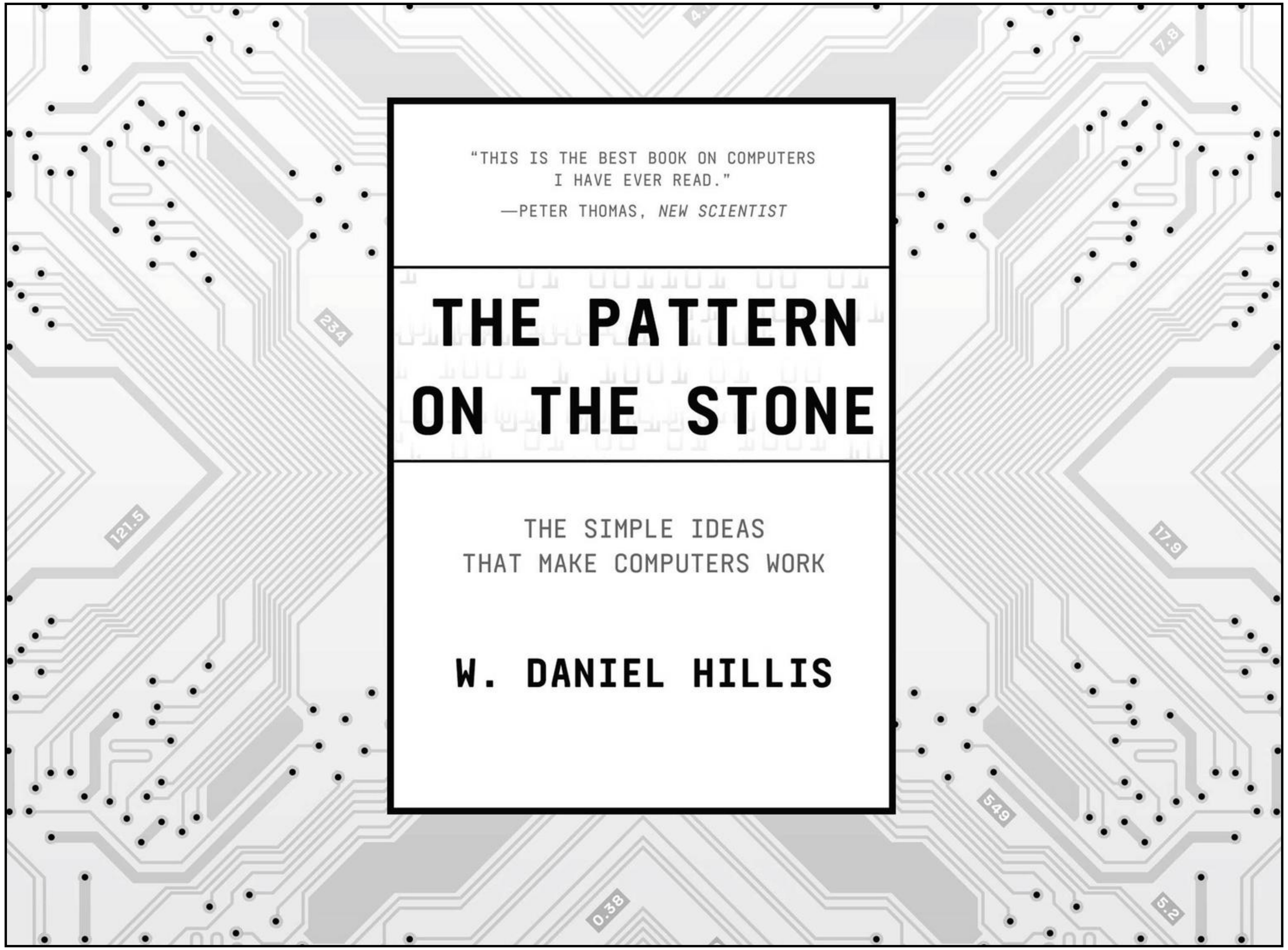
If you keep going with the CS major sequence, you work your way up to some really exciting courses, including…

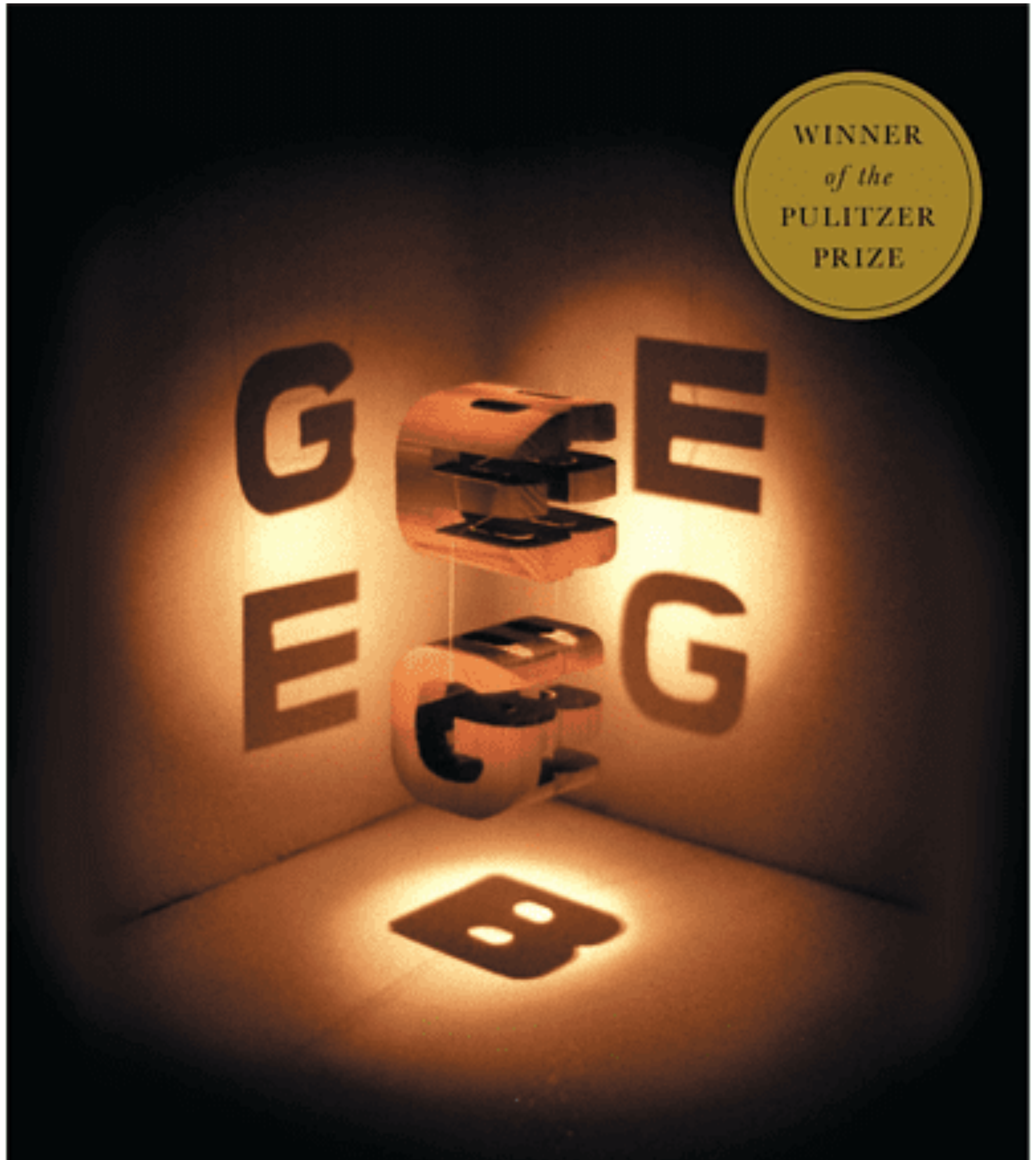
CMPU 353 Bioinformatics

CMPU 377 Parallel Programming

And, you know, probably some cool courses I don't teach as well!

Further reading


"THIS IS THE BEST BOOK ON COMPUTERS I HAVE EVER READ."
—PETER THOMAS, *NEW SCIENTIST*
THE PATTERN ON THE STONE
THE SIMPLE IDEAS THAT MAKE COMPUTERS WORK
W. DANIEL HILLIS

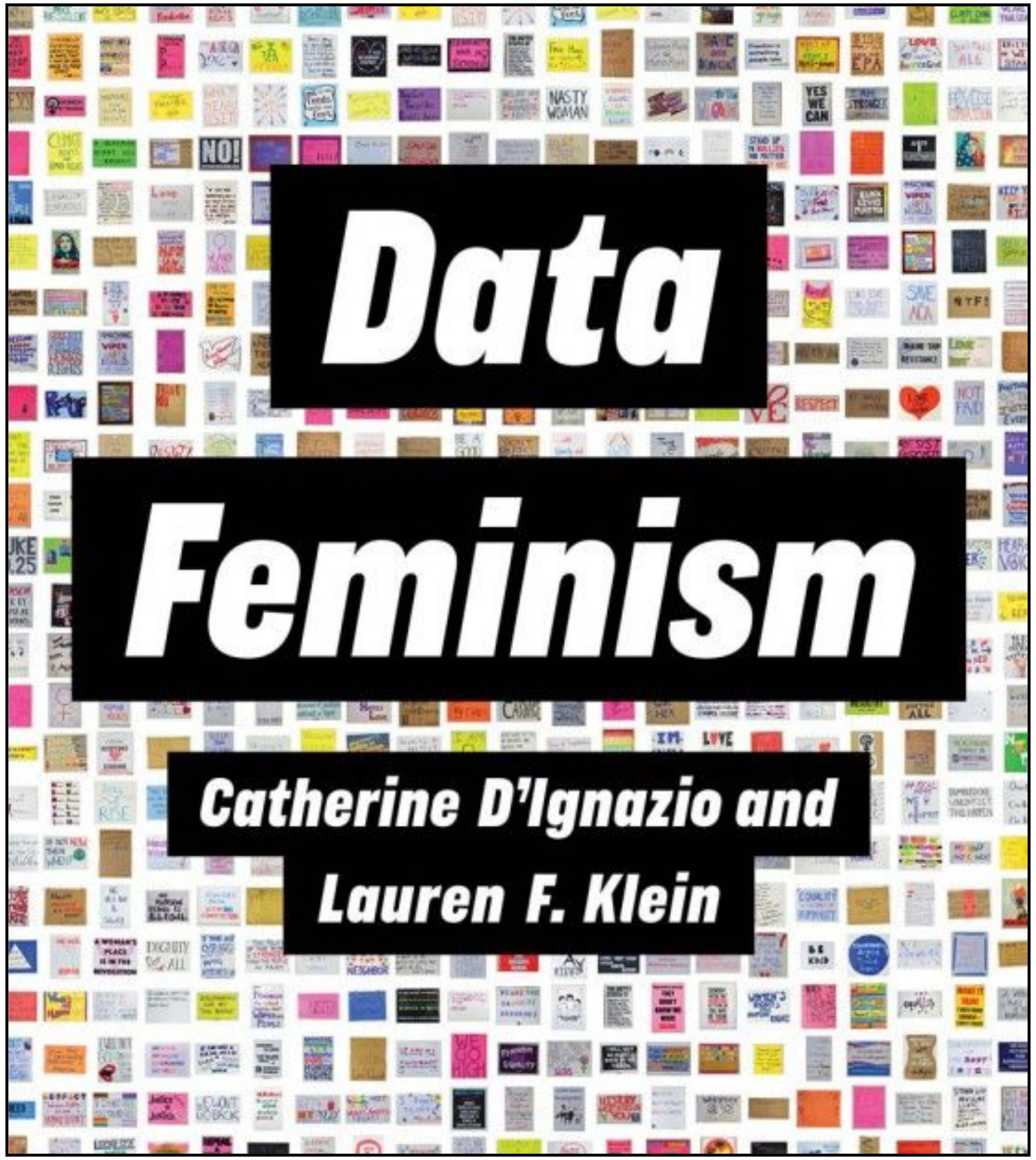


GÖDEL, ESCHER, BACH:

an Eternal Golden Braid

DOUGLAS R. HOFSTADTER

A metaphorical fugue on minds and machines in the spirit of Lewis Carroll



data-feminism.mitpress.mit.edu

That’s it!

We'll meet some time during study week to review for Exam 2, working through practice problems and answering your questions.

go.vassar.edu/course/evals

Acknowledgments

This lecture incorporates material from:

Frederick P. Brooks, Jr, *The Mythical Man-Month*

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

Miram Posner, UCLA

Jonathan Gordon, Vassar College