Introduction

Introductions

Welcome! I'm Harry Smith (he/him/his)

- As a lecturer, my job is **teaching**—I'm here for you!
- Find me at sharry@seas.upenn.edu or in Levine 269C
- Office Hours Tue 12:15pm-1:45pm, Wed 11:30am-12:30pm
- Advising Hours Mon 11am-12pm

About Me

Undergrad	Here!
Grad School	Columbia
Academic Interests	Data Viz, CS Education
Height	6'1
Favorite Food Near Campus	Han Dynasty
Favorite Album of 2023	*Rat Saw God* by Wednesday
Favorite Authors	Rachel Cusk, Elena Ferrante, Natalia Ginzburg
Favorite Video Game	*Disco Elysium*

Introductions

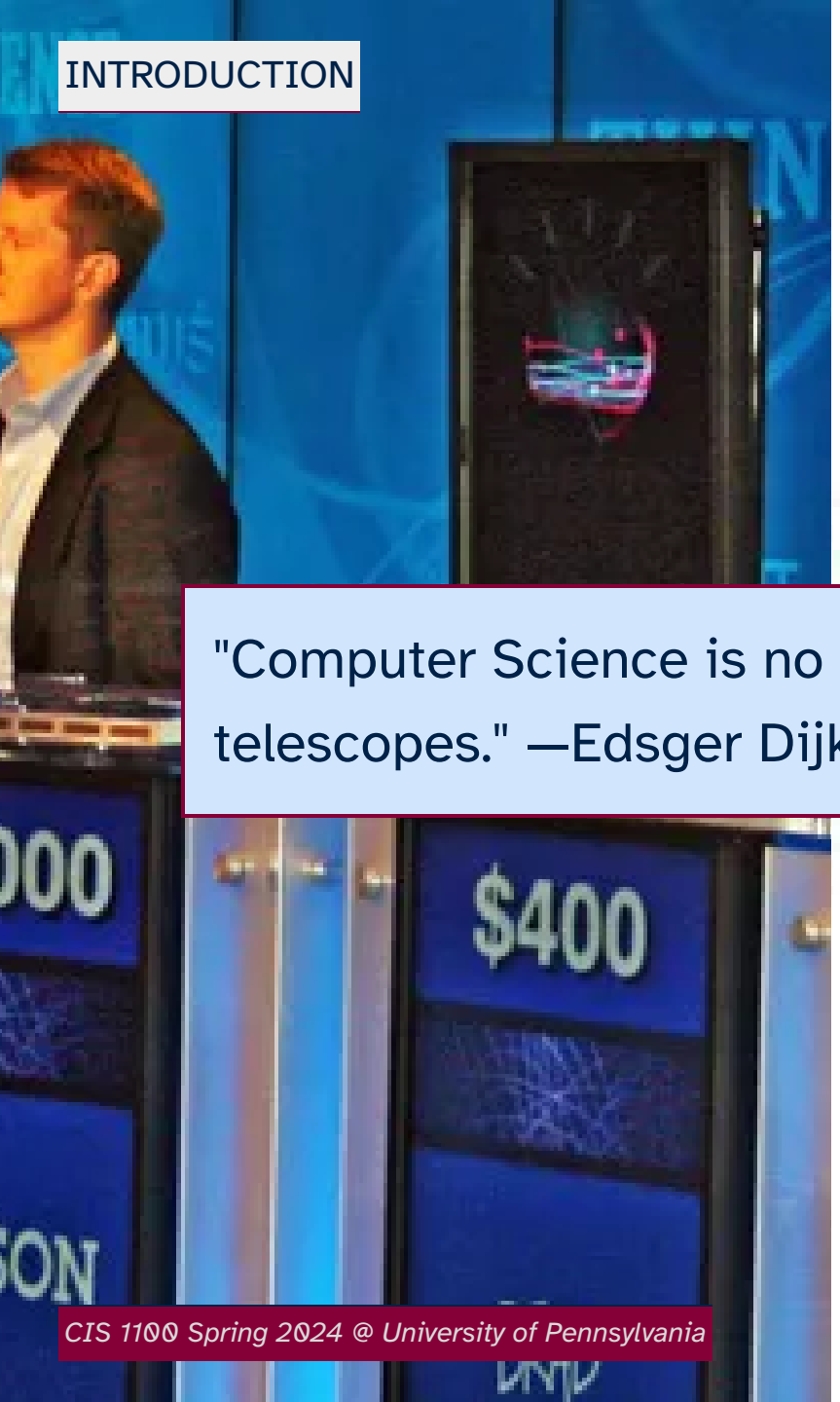
We have ~40 TAs this semester here to help you learn.

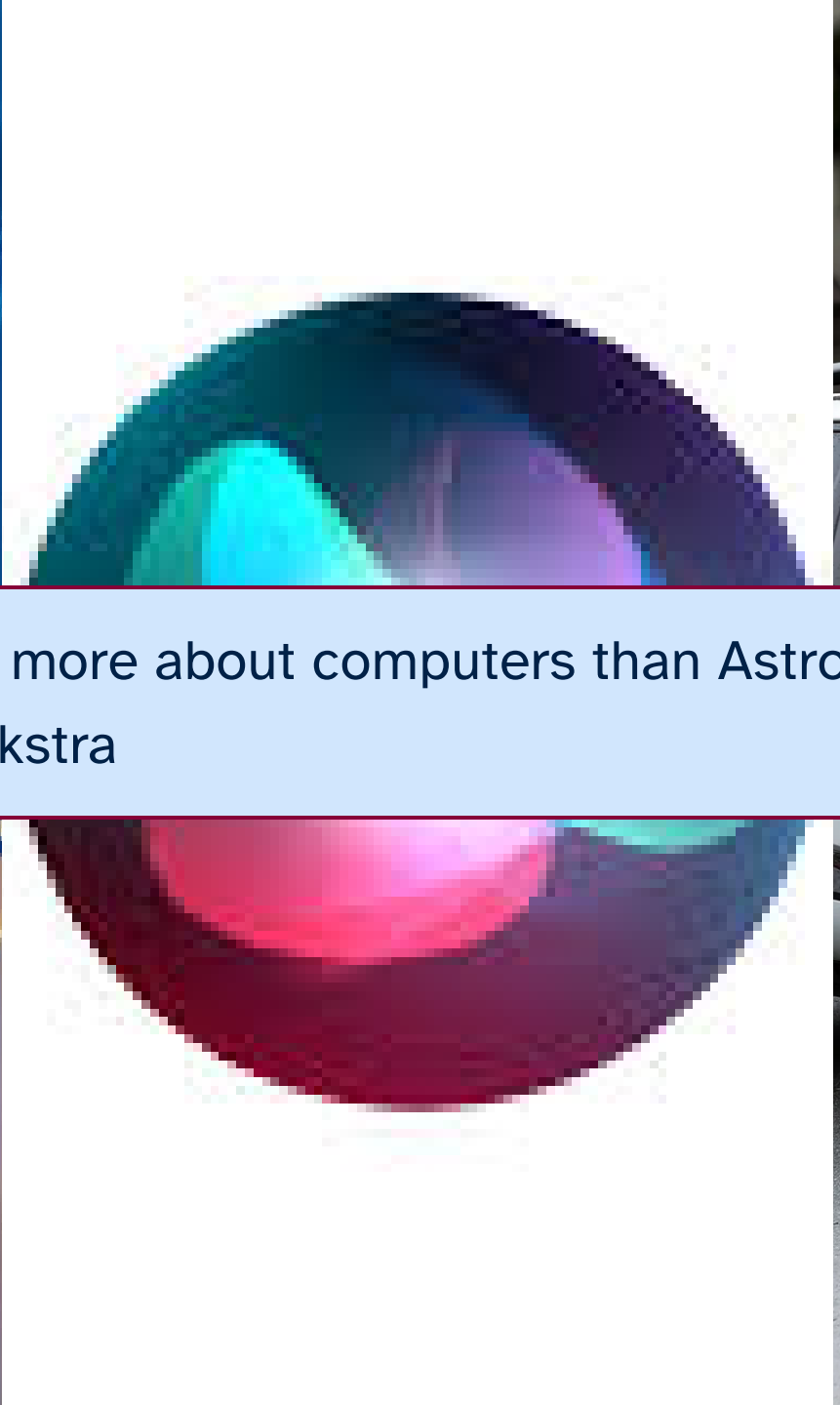
Disclaimer

I'm (kind of) reading a script for this lecture—so much to say today!

Normally, I won't be teaching exactly like this.

What is CIS 1100?

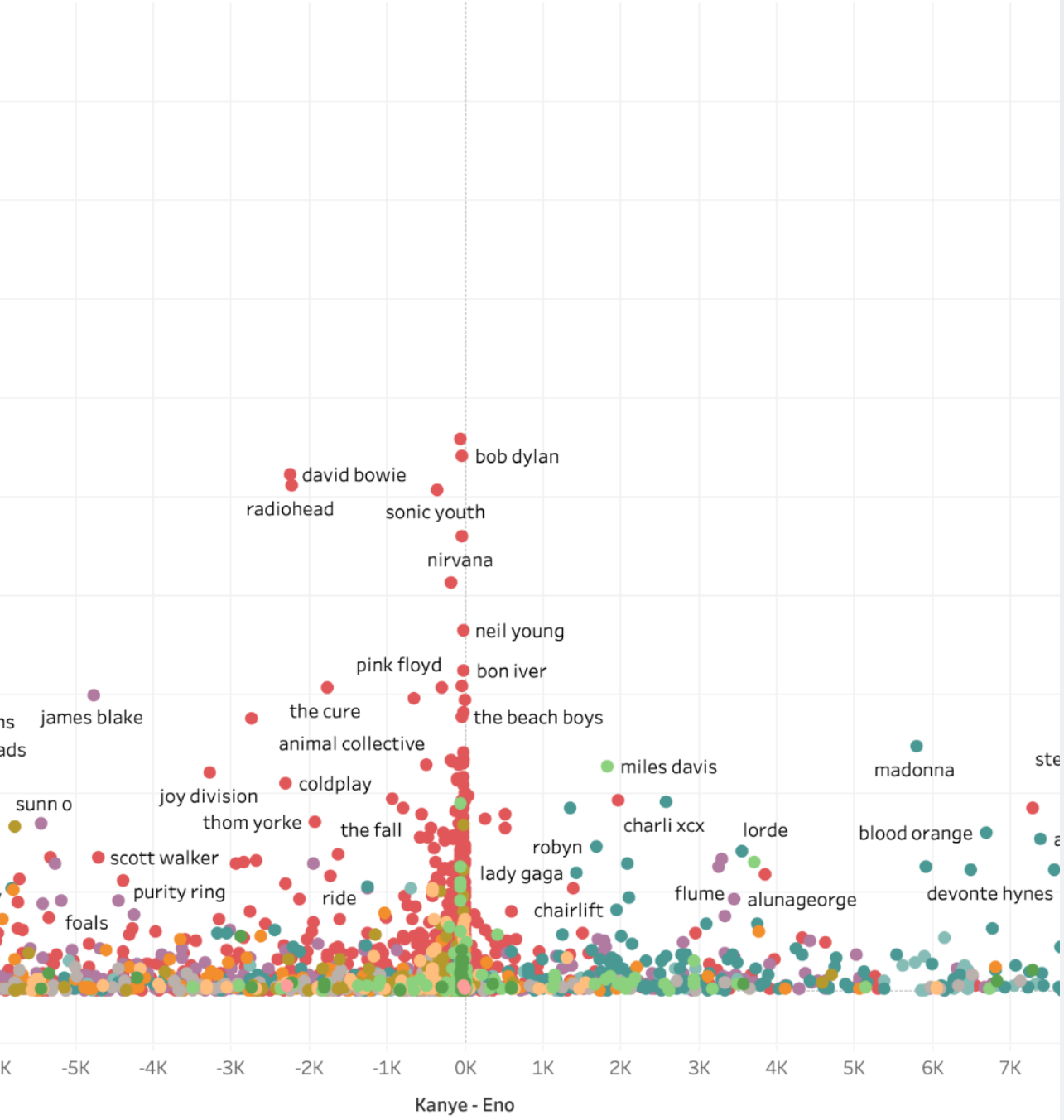




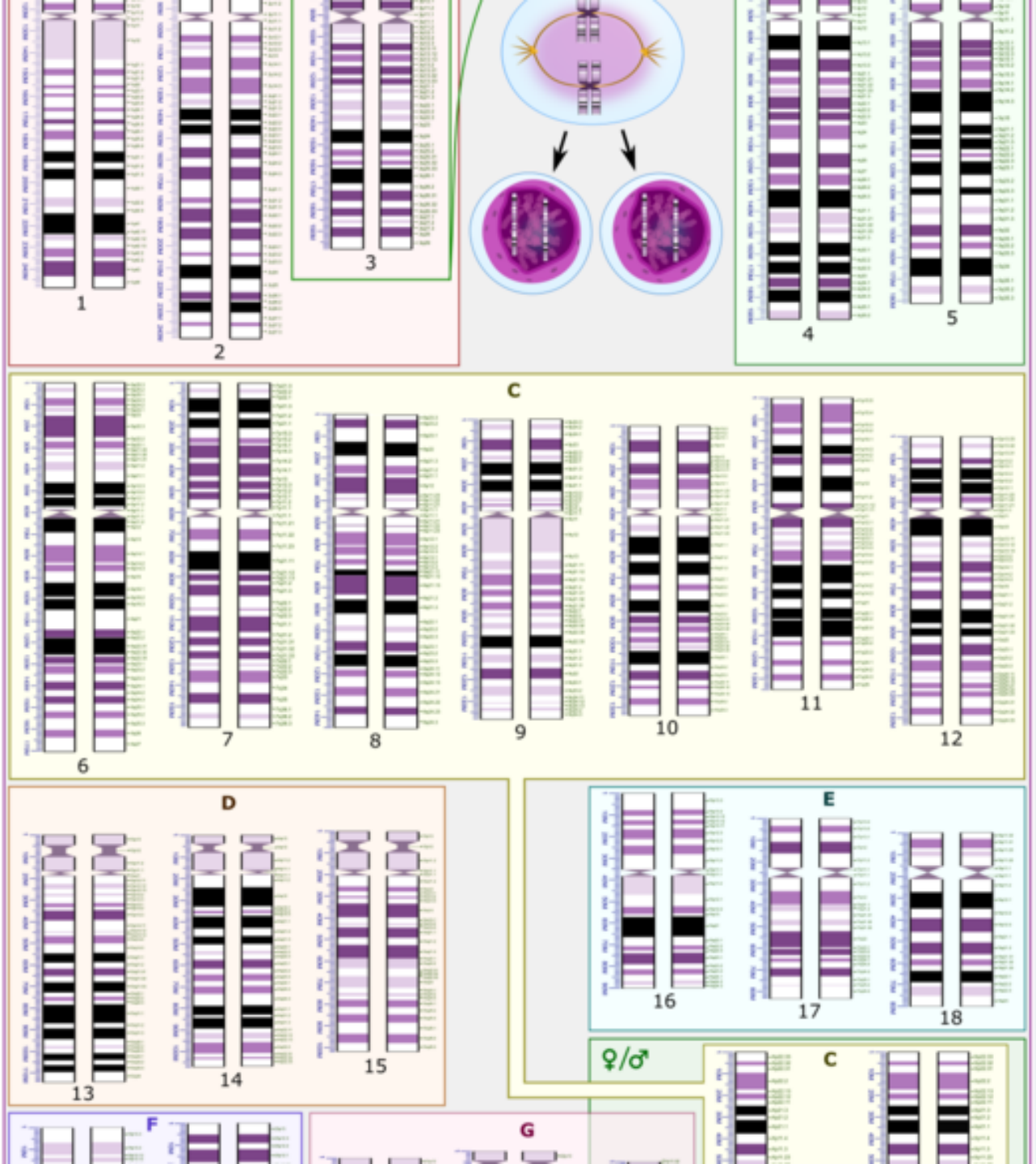
"Computer Science is no more about computers than Astronomy is about telescopes." —Edsger Dijkstra

Who is CIS 1100 For?

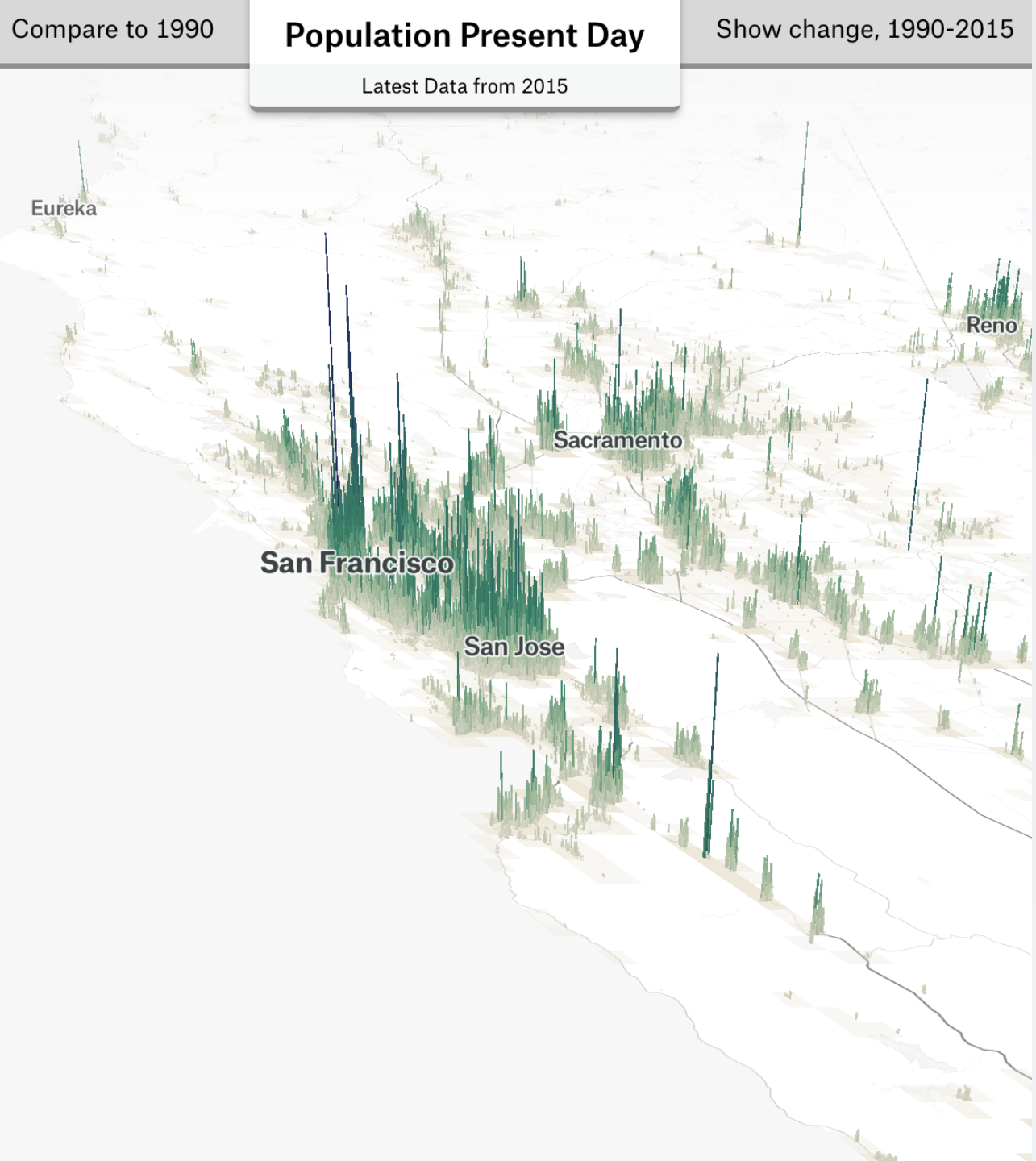
🎉 🎉 🎉 *Everyone!* 🎉 🎉 🎉



Digital Humanities



Biology



Data Journalism

Course Structure (the highlights)

- Programming is not a spectator sport.
- This course is taught in **Java** - suitable for beginners.

Lecture

- Three times a week.
- Lecturing, with additional live demos, Q&A, and polls.
- **Recordings:** Available on Canvas.
- You are expected to engage, ask questions, practice *metacognition*.

Recitation

- These are **mandatory** weekly sessions
- **Format:** Review, group work, TA interaction.
- You must be registered for **one** section; 1 hour or 2 hours both fine.
- Attendance & participation included in your final grade!

Assignment Types

- You will be assessed in various ways outside of lecture and recitation.
- There are four main types of assignments: 📝

Weekly Check-In Activities

- Short worksheets or small programming tasks.
- Complete them individually.
- Submit and repeat as needed.
- Get guidance from TAs.
- Collaboration with peers allowed if needed.

Programming Projects (Homeworks)

- One- or two-week assignments.
- Apply course material to real programs.
- Must be done individually.
- Seek TA assistance for understanding and debugging.
- Challenging, may take 10-15 hours.

More Homework Policies

- Notable policies: lateness and collaboration. ⚖️
- You have four "late days" for Homeworks.
- Each "late day" allows up to 24 hours delay.
- At most two late days per assignment.
- No submissions after the deadline without a late day. ❌
- Homeworks must be completed individually.

Midterm Exams

- Two exams during the semester.
- Held in class:
  - February 26 & April 15
  - Mondays! Including before Spring Break.
- Closed-book exams. 🚫📚
- Together, they account for 12% of your final grade.
- Test your Java fluency under time constraints.

Final Exam

- Scheduled during the official finals period. 🗓️
- Worth 12% of your final grade.
- Similar to midterms, assesses Java fluency.
- Date and time will be announced soon.

Tips for Success

Other quick tips for success:

1. Start your assignments as soon as they're released to you.
2. Come to Office Hours all the time, even if you don't have a question.
3. If you ever need support, either academic or otherwise, don't hesitate to reach out.

TA Resources

Sunday Review Sessions

- Weekly on Sundays! Time TBD
- Useful for exam review, interactive small group work

Code Reviews

- One of these is mandatory in the first month of the course
- After that, you can sign up for them as needed!
- Great for 1:1 time with a TA

Online Resources

- Course Website: cis1100.com
  - hw assignments, schedule, recitation info, course policies
- Canvas
  - class recordings
- Gradescope (coming soon)
  - assignment submission & grading
- Codio (use course token `trust-plato`)
  - writing code

LLMs & Other Online Resources

- Good: discuss course content or plan homework strategies at a high level with your classmates
- Bad:
  - writing code together or reading each other's code
  - searching for solutions on the internet
  - using an old solution that a friend gave you
  - using Large Language Models like ChatGPT or CoPilot to do your assignments for you.

First Program: `HelloWorld.java`

We'll write a Java program that prints out the message `"Hello, World!"` when compiled and run.

Definitions:

- Java program
  - Code, written in the Java language, that specifies a series of instructions for a computer to execute
- Printing 🖨️
  - Displaying text to the screen
  - (not actually putting ink on a piece of paper)
- Compiling
  - A process that turns Java code (stuff that we write) into specialized instructions that a computer knows how to execute
- Running 🏃‍♀️
  - Executing the compiled instructions!

File Naming Conventions

When you create a new file in Codio, you'll have to name it in a special way to make sure it's run and recognized as a Java program.

- For our first program, we'll call it `HelloWorld.java`.
  - 🚨Make sure to use this name exactly!🚨
- Since the filename of our program is `HelloWorld.java`, our program should start with `public class HelloWorld`

```
public class HelloWorld {

}
```

The "Class"

- Further study required to really understand this term 😕
  - This is one of the (myriad) miseries of Java!
- For now, we'll say that the concept of a *class* is synonymous with that of a *program*.
  - A bunch of code that lives in a file that is identified with a particular name
  - In our first case, `HelloWorld`

The `main` method

Next up, we'll add some more jargon to our program.

```
public class HelloWorld {
  public static void main(String[] args) {

  }
}
```

This new stuff in the middle of the braces we wrote after the class declaration is called the "`main` method". It has its own curly braces which will contain all of the statements that actually give our program interesting behavior.

Finishing up the Program

```
public class HelloWorld {
  public static void main(String[] args) {
    System.out.println("Hello, world!");
  }
}
```

Here, we've added a *print statement* to our `main` method that tells the program to print out the message `"Hello, world!"`.

Print Statements

```
System.out.println("Your message goes here! Don't forget the quotes.");
```

- Writing `System.out.println` followed by a pair of parentheses allows you put a message in those parentheses to be printed out.
    - Make sure to put the text of the message between a pair of double quotes (these `"`).
- Remember to put a semicolon (`;`) at the end of the line, too!

Comments

These are used as notes that describe your program, contained within the program.

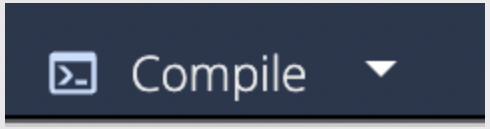
They are ignored by the computer and are meant only for other humans (you, your teammates) to read.

```
// this is a comment on one line, denoted by the double slashes

/* You can also write multi-line comments this way,
 * starting them with "slash-star" and ending with
 * "star-slash".
 */
```

Compiling and Running

To see what the program does, you have to *run* it! To run it, you must first *compile* it.



(click this button here!)

Compiling

README.md | .codio | HelloWorld.java | Compile

```
Welcome to Ubuntu 14.04.5 LTS (GNU/Linux 5.19.0-1028-aws x86_64)

 * Documentation:  https://help.ubuntu.com/
 *
 * Welcome to the Codio Terminal!
 *
 * https://docs.codio.com/develop/develop/ide/boxes/overview
 *
 * Your Codio Box domain is: rentminus-lasermayday.codio.io
 *
Last login: Tue Jul 25 01:23:37 2023 from 192.168.10.93
codio@rentminus-lasermayday:~/workspace$ javac -cp cis110.jar:junit-platform-console-st
andalone-1.3.2.jar:. *.java
codio@rentminus-lasermayday:~/workspace$
```

Successful!

Compiling

README.md | .codio | HelloWorld.java | Compile

```
codio@rentminus-lasermayday:~/workspace$ javac -
andalone-1.3.2.jar:. *.java
HelloWorld.java:3: error: cannot find symbol
    System.out.println("Hello, world!");
               ^
  symbol:   method println(String)
  location: variable out of type PrintStream
1 error
codio@rentminus-lasermayday:~/workspace$
```

Failure! The message reports `"1 error"`.

Compiling

```
codio@rentminus-lasermayday:~/workspace$ jav
andalone-1.3.2.jar:. *.java
HelloWorld.java:1: error: '{' expected
public class Hello World {
                      ^
1 error
codio@rentminus-lasermayday:~/workspace$
```

A Different Failure! (again, `"1 error"`).

Running

Once you have a successful compilation (no errors reported), you can run the program by typing **into the terminal window** `java HelloWorld`.

```
codio@rentminus-lasermayday:~/workspace$ java HelloWorld
Hello, world!
codio@rentminus-lasermayday:~/workspace$
```

See the message that got printed?

For You to Try

1. Try changing `"Hello, World!"` to something else—but remember your quotation marks!
2. Try adding another line of `System.out.println();` with a different message.
3. Replace the first `System.out.println()` with `System.out.print()`—what's the difference?