LEC 19

CSE 121

Victory Lap!

BEFORE WE START

Talk to your neighbors:

What is your go-to end-of-finals celebration?

Music: 🌻 CSE 121 26sp Lecture Tunes 🌻

Instructors: Matt Wang

TAs:

Abdul	Amogh	Anant	Anum	Cayden
Dalton	Ethan	Hayden	Jesse	Jessica
Johnathan	Minh	Navya	Paul	Reese
Ruslana	Sam	Savannah	Spencer	Shayna
Tamsyn	TJ	Trey		

Questions during Class?

Raise hand or send here

sli.do #cse121

Announcements, Reminders

- R7 (and R-Extra) due **Thursday, June 11th**
  - all assignments are eligible for resubmission!
  - R-Extra email has gone out!
- Today is the last day for IPL (instructor office hours already done :/)
- Final Exam: **Wednesday, June 10th from 2:30 – 4:20 PM, in GUG 220**
  - Review the Exam page of website (with policies & resources)
  - Cannot make it? Email me **ASAP**
- TA-led review session: **Mon, June 8th from 4:30 – 7 PM in BAG 154**

Evaluations and Awards

Please give us feedback!

- **Course Evaluations** are due **Sunday, June 7th at 11:59 PM**
- **TA Evaluations** are *also* due **Sunday, June 7th at 11:59 PM**
- We read *every* piece of feedback!

Bob Bandes TA Award nominations open!

- thought your TA was goated? write them a nomination!
- fun fact: some of our faculty won the award when *they* were TAs!

What is `String[] args`?

- Live demo in class ☺
- Note: this is *not* tested material!

Applications of Computer Science

or "What can I do with what I learned?" – outside of just "write code":

- Help deaf & hard-of-hearing people identify sounds
- Develop a programming language that celebrates the world's languages
- Build battery-free robots, put them on insects, then… track murder hornets!?!
- Detect and prevent toxicity online & recognize disinformation
- Create an interactive atlas of millions of refugee experiences
- Fix Olympic badminton & identify cheating in chess
- and so much more!

… including our assignments! (1/2)

- Computational Biology & Medicine (P2, P3)
  - in CSE: Chris Thachuk, Sara Mostafavi, Su-In Lee, Luis Ceze
- Computational Art (C0, C1)
  - UW CSE has many unique intersections of CS + art!
  - "Cultural-Centric Computational Embroidery" (CSE + iSchool)
  - "Computational Illusion Knitting", "How to Knit Objects Weird"
  - "WasteBanned: Supporting zero waste fashion design"
- Computer Security (C2)
  - in CSE: Franzi Roesner, David Kohlbrenner, Nirvan Tyagi

… including our assignments! (2/2)

- Games & Graphics (C1, C3)
  - at UW: many labs in CSE and iSchool's GAMER group
  - fun fact: Foldit is a crowd-sourced game for protein folding
    - David Baker shared last year's Nobel Prize in Chemistry, in part for this!!
- Social Computing (P1)
  - at UW: Amy Zhang's Social Futures Lab + so much of iSchool
- and many side quests (in lecture, section, PCM): accessibility (e.g. UW CREATE), weather, music data, chatbots, and lots of math

Closing the loop on P3 reflections – topics

Y'all listed many other applications of CS too! Some common areas:

- civil engineering & architecture
- sports
- agriculture
- environmental science & engineering
- archaeology
- 2D & 3D animation

Libraries & Archives! (1/3)

"Sometimes I enjoy volunteering online by transcribing historical documents which is really important not only for accessibility of information across large distances but also a more viable way of preserving information.

Not only is computer science really important in the actual creation of websites and digital archives to store this information but some of the basic skills can be reminiscent of coding/[pseudocode] like in the videos example of the Old McDonalds song."

Libraries & Archives! (2/3)

"Also, one of the fields I am interested in post-grad is library science and being a public librarian.

I think when most people think of librarians, they would not think about computer science--but it plays a significant role, because you have to manage databases that track not only what books the library owns, but also whether copies are checked out or on shelves, where on shelves they're located, […]"

Libraries & Archives! (3/3)

"For this I feel like we can talk about libraries! Libraries are what I would consider one of the greatest things the average person ignores [...]

Specifically library digitization and organization have been incorporating computer science to streamline access to knowledge.

For example with the UW library you can check the status of a book and where you can find it thanks to the digitization of what began as a physical information system."

Benjamin Charles Germain Lee

- Assistant Professor at UW's iSchool
- previous PhD student in CSE at UW; undergrad in astrophysics & math (not CS!)
- *very cool* work related to libraries & archives
  - Newspaper Navigator (Library of Congress)
  - Govscape – searching > 10 million US government PDFs!!
  - archival work at US Holocaust Memorial Museum (see: "Uncanny Testimony")

Future Courses

or "What can I do next?"

Non-majors

Course	Overview
CSE 154	Intro to web programming (several languages)
CSE 160	Intro programming, data analysis (Python)
CSE 163	Intermediate programming, data analysis (Python)
CSE 180	Introduction to data science (Python)
CSE 373	Data structures and algorithms (in Java)
CSE 374	Low-level programming and tools (C/C++)
CSE 412	Intro to Data Visualization
CSE 416	Intro. to Machine Learning
CSE 493E	Accessibility

More 12X!

Course	Overview
CSE 122	Data structures, object-oriented programming
CSE 123	More OOP, recursion

Majors

Course	Overview
CSE 311	Mathematical foundations
CSE 331	Software design/implementation
CSE 340	Interaction programming (apps)
CSE 341	Programming languages
CSE 351	Hardware / Software Interface
CSE 480	Social impacts of computing

Related majors: Informatics, ACMS, HCDE, ECE, …

See: https://www.cs.washington.edu/academics/ugrad/current-students and https://www.cs.washington.edu/academics/ugrad/nonmajor-options/nonmajor-courses

Closing the loop on P3 reflections – journey

We also asked you to reflect on your journey learning CS!

Impossible to adequately summarize them all, but the biggest theme was **combating myths about computer science:**

- what programming & computer science is
- what makes computer science hard (it's not just syntax!!)
- what you need to be a "good" programmer (not always math!!)
- what programmers & computer scientists look like
- **that one bad experience means you'll *always* be bad at it**

Metacognition & Growth Mindset

“Before taking this class, I thought that some people were naturally good at computer science and just "got it." I thought it would be a skillset that it really hard to learn.

Reflecting back now, I still think that it comes much easier to some people and its still hard to learn and understand what to do from a specification but I also think that I've improved my problem solving skills and understand of coding.”

… and some honest (and funny) answers!

"I honestly didn't think I would enjoy it that much. I got sick of the weed-out classes at UW and expected this one to be the same as the rest. I ended up really enjoying coding and I definitely want to look into a career that revolves around it.... hopefully there are some out there."

"I thought my friends who understood [CS] were people speaking a gibberish language at me that I would never understand. After taking this course half of this is true! I kinda get it now! It is totally still gibberish though lol."

"Matt was one of the funniest teachers of all time! would love to see him like posts videos or clips of his teaching."

– trust me, you don't ;)

Generalizing *beyond* Computer Science

Some of you said, ”I’m glad I took this class, but no more CS for me”

That’s totally valid!

Some lessons from this class that *could* apply more broadly:

- how to break big problems into smaller subproblems
- how to isolate what part of a system is broken
- attention to detail
- understanding basics of how software works
- how to learn (and reflect) effectively

Frequently Asked Questions

How can I get better at programming?

- Practice!

How can I learn to X?

- Classes, books, videos, or self-learn!
- CS (as a field) has lots of free resources :)

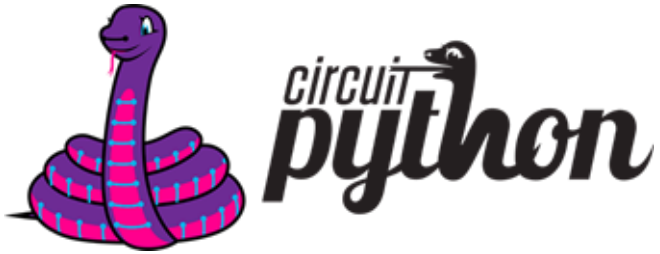
What should I do next?

- Anything you're interested in!
- but: hard to tell what's easy and what's hard

Should I learn another language? Which one?

- That depends – what do you want to do?

Aside: Cute Programming Language Logos



Deno

Summer Break Project: Tic Tac Toe

Build your own Tic Tac Toe game + "AI"!

1. How would you represent a Tic Tac Toe game in Java?

   (hint: arrays will be very, very helpful!)

2. Write a method that tells you if a Tic Tac Toe game is won.

3. Write a method that gets input from the user and "makes" a move.

4. Wrap it all up – into a nice two-player game!

Summer Break Project: Tic Tac Toe++

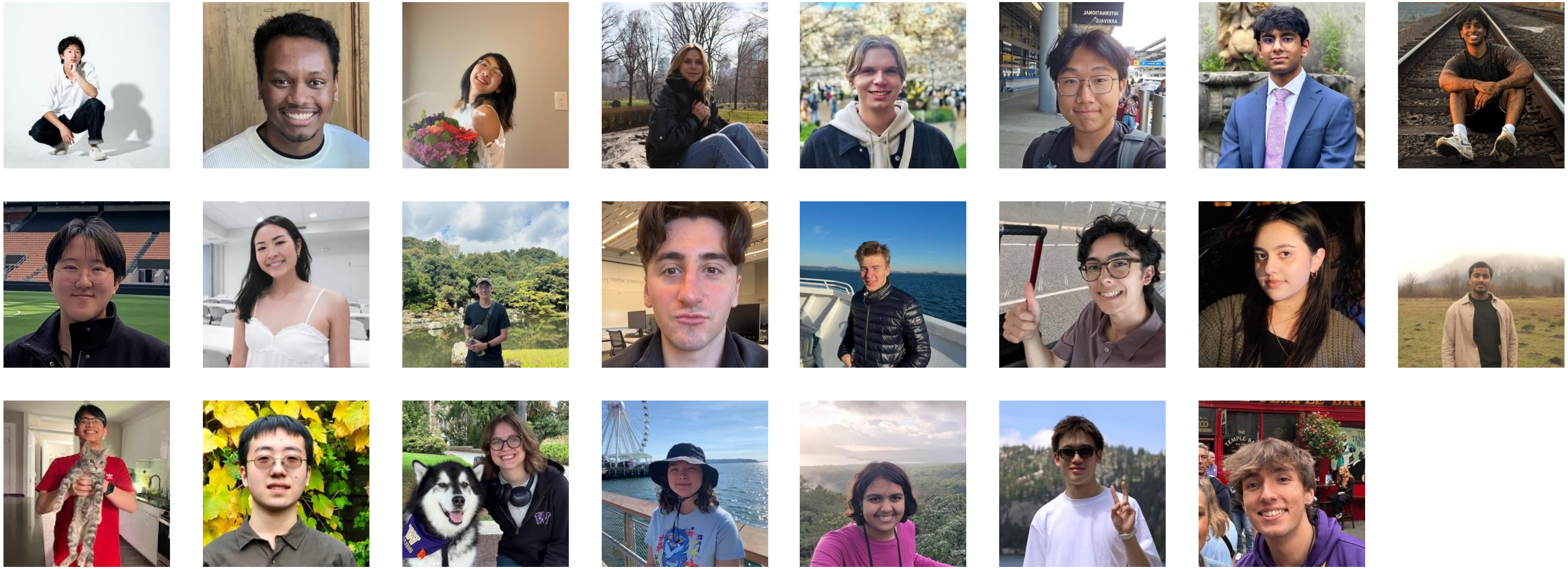
Make some "AI" that…

- just makes a random valid move (you should be able to beat this!)
- tries to make a "good" move (~ some if statements)
- never loses
  - Tic Tac Toe is a "solved game": a perfect player will never lose.

Or, extend this idea to other grid-based games!

- similar-ish: connect four, checkers, battleship
- much harder: sudoku, chess, go, othello

Thank your 23 <u>fabulous</u> TAs!



Thank you!

Ask Me (Almost) Anything!

sli.do #cse121