

CS 106X, Lecture 30

Wrap-up / What's Next?



Plan For Today

- **Recap:** CS 106X
- Next Steps
- Parting Words
- Ask-Anything
- Course Evaluations

Plan For Today

- **Recap: CS 106X**
- Next Steps
- Parting Words
- Ask-Anything
- Course Evaluations

First Day

```
/*
 * hello.cpp
 * This program prints a welcome message
 * to the user.
 */
#include <iostream>
using namespace std;

int main() {
    cout << "Hello, world!" << endl;
    return 0;
}
```

We Learned About C++

- Basics of programming in the C++ language
 - functions; value vs. reference parameters
 - pointers
 - strings
 - streams
 - operator overloading
 - Templates
 - classes
 - memory management
 - Stanford C++ libraries
 - Qt Creator IDE
 - Structs
 - How to turn QT Creator off and on again

The screenshot shows the Qt Creator IDE interface. The left sidebar displays the project structure for 'FinalSample' with 'FinalSample.pro' at the top, followed by 'Headers' and 'src' directories containing various header files like 'ArrayList.h', 'BasicGraph.h', etc. Below these are 'Sources' and 'lib/StanfordCPPLib' sections with corresponding source files. The main central area shows the code editor with a snippet of C++ code for 'sorting.cpp'. The code implements a sequential search and a selection sort algorithm. The bottom right corner shows the 'Compile Output' tab displaying the terminal output of the build process, which includes compilation commands and the execution of the 'make' command.

```
181     return 0;
182 }
183 }
184
185 // Returns the index of the given value in the given vector of integers,
186 // or -1 if the value is not found in the vector.
187 int sequentialSearch(const Vector<int>& v, int value) {
188     for (int i = 0; i < v.size(); i++) {
189         if (v[i] == value) {
190             return i;
191         }
192     }
193     return -1;
194 }
195
196 // Rearranges the elements of v into sorted order using
197 // the selection sort algorithm.
198 void selectionSort(Vector<int>& v) {
199     for (int i = 0; i < v.size() - 1; i++) {
200         // find index of smallest remaining value
201         int min = i;
202         for (int j = i + 1; j < v.size(); j++) {
203             if (v[j] < v[min]) {
204                 min = j;
205             }
206         }
207         swap(v[i], v[min]);
208     }
209 }
```

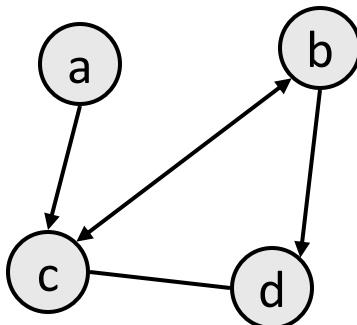
```
g++ -Wl,-subsystem,windows -mthreads -o debug\FinalSample.exe
object_script.Finalsample.Debug -lglu32 -lopengl32 -lgdi32 -luser32 -lmingw32 -
lqtmaind -LC:\Programs\Qt\5.1.1\mingw48_32\lib -lQt5Guid -lQt5Core
mingw32-make[1]: Leaving directory 'C:/Users/stepp/Documents/stanford/cs106b/
13au/exams/final/build/FinalSample-Desktop_Qt_5_1_1_MinGW_32bit-Debug'
03:19:05: The process "C:\Programs\Qt\Tools\mingw48_32\bin\mingw32-make.exe"
exited normally.
03:19:05: Elapsed time: 00:01.
```

We Learned About Good Style

- Decomposition
- Commenting
- Descriptive Naming
- Abstraction
- Unit Testing
- Debugging

We Learned About Collections

- How to use various **collections** to store and manipulate data
 - Vector, LinkedList
 - Grid
 - Stack, Queue
 - Set, HashSet, Lexicon
 - Map, HashMap
 - PriorityQueue
 - Graph



Screenshot of a web browser displaying the documentation for the `Vector<ValueType>` class from the Stanford cslib package.

The page includes the following sections:

- Constructor**:

<code>Vector()</code>	Initializes a new vector.
<code>Vector(n, value)</code>	
- Methods**:

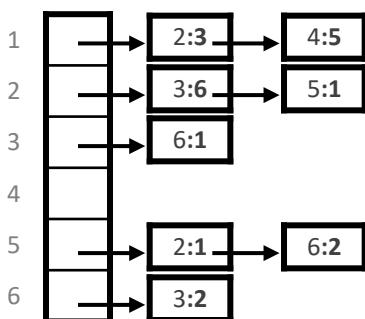
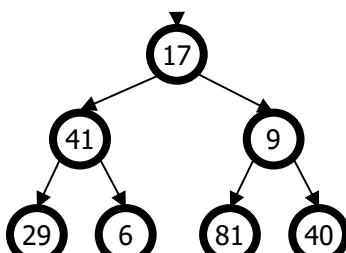
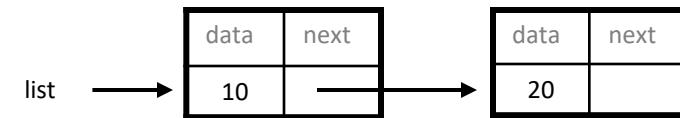
<code>add(value)</code>	Adds a new value to the end of this vector.
<code>clear()</code>	Removes all elements from this vector.
<code>get(index)</code>	Returns the element at the specified index in this vector.
<code>insert(index, value)</code>	Inserts the element into this vector before the specified index.
<code>isEmpty()</code>	Returns <code>true</code> if this vector contains no elements.
<code>mapAll(fn)</code>	Calls the specified function on each element of the vector in ascending index order.
<code>remove(index)</code>	Removes the element at the specified index from this vector.
<code>set(index, value)</code>	Replaces the element at the specified index in this vector with a new value.
<code>size()</code>	Returns the number of elements in this vector.
<code>toString()</code>	Converts the vector to a printable string representation.
- Operators**:

<code>v1 + v2</code>	Concatenates two vectors.
<code>v1 += v2;</code>	Adds all of the elements from <code>v2</code> (or the single specified value) to <code>v1</code> .
<code>vec.mapAll(fn);</code>	Adds an element to the vector passed as the left-hand operand.
<code>vec[index]</code>	Overloads <code>[]</code> to select elements from this vector.

We Learned How Those Collections Are Implemented

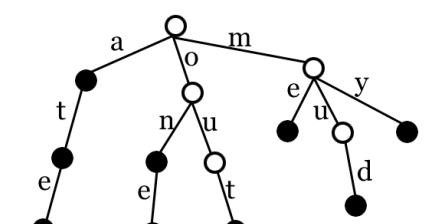
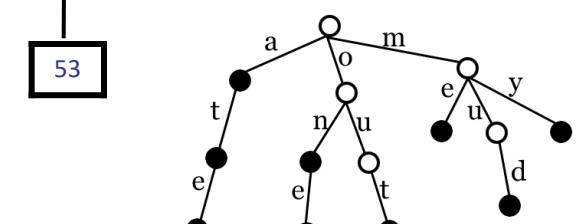
- The internal **data structures** that can implement those collections:
 - array (Vector)
 - linked list (LinkedList, Stack, Queue)
 - tree (Map, Set, Lexicon)
 - hash table (HashSet, HashMap)
 - edge list / adjacency list / matrix (Graph)

index	0	1	2	3	4	5	6	7	8	9
value	3	8	9	7	5	12	0	0	0	0
size	6	capacity 10								



1	0	3	0	5	0	0
2	0	0	6	0	1	0
3	0	0	0	0	0	1
4	0	0	0	0	0	0
5	0	1	0	0	0	2
6	0	0	2	0	0	0

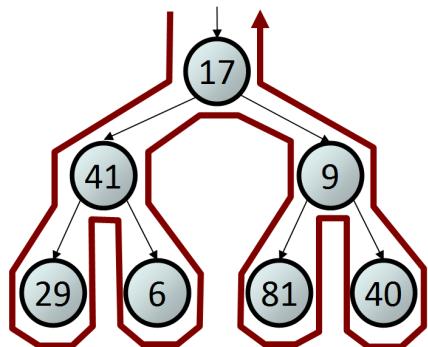
index	0	1	2	3	4	5	6	7	8	9
value										



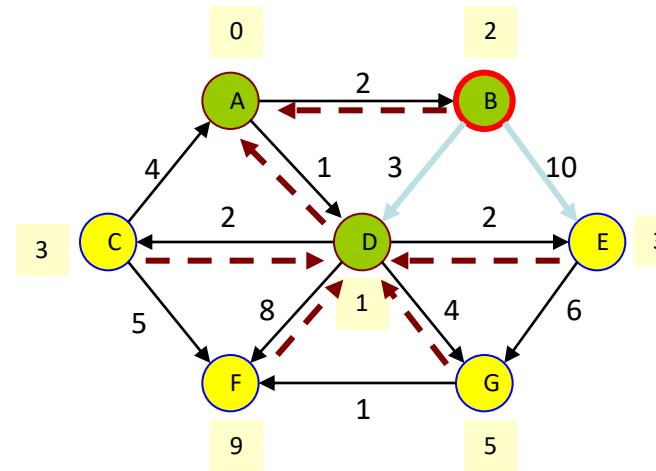
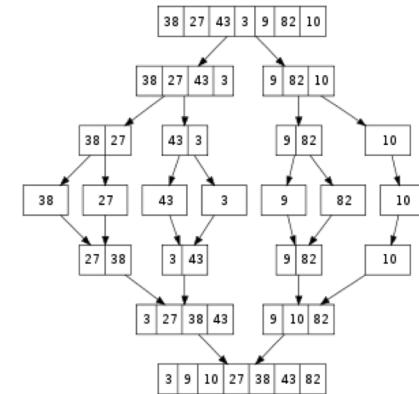
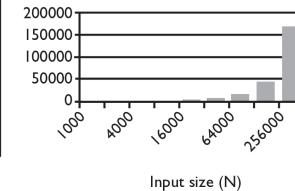
We Learned About Algorithms

- Several useful **algorithms** that operate on data:

- searching
- sorting; topological sort
- tree traversals
- graph path searching: DFS, BFS, Dijkstra's, A*
- algorithm analysis
- Big-Oh notation



N	Runtime (ms)
1000	0
2000	16
4000	47
8000	234
16000	657
32000	2562
64000	10265
128000	41141
256000	164985

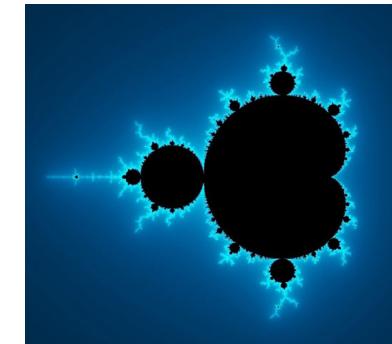
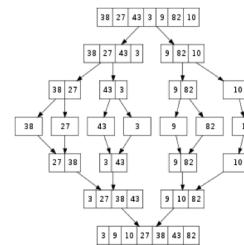
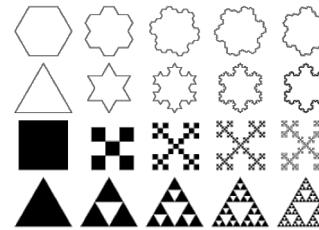


Selection Sort.

8	5	7	1	9	3
1	5	7	8	9	3
1	3	7	8	9	5
1	3	5	8	9	7
1	3	5	7	9	8
1	3	5	7	8	9

We Learned About Recursion (By Learning About Recursion)

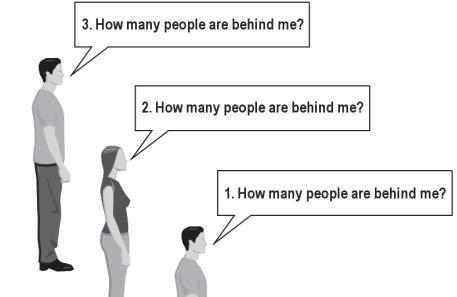
- Self-similarity and **recursion**
 - fractals
 - recursive data (e.g. files and folders)
 - exhaustive search
 - backtracking
 - recursive data structures



```
void reverseLines(ifstream& input) {      // call 1
    string line;
    if (getline(input, line)) {              // "Roses are red."
        reverseLines(input);                // make call 2 and wait
        cout << line << endl;
    }
}

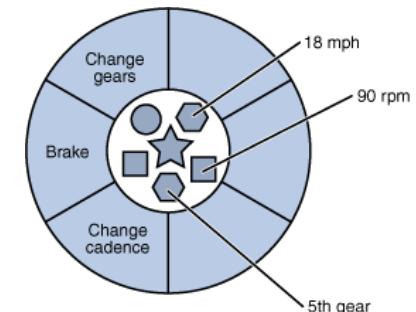
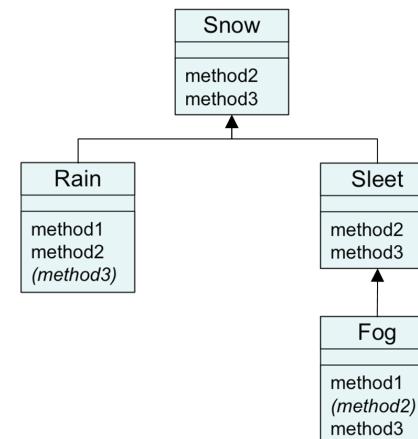
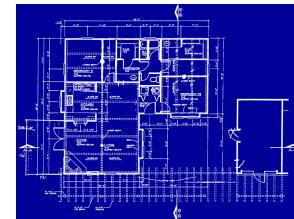
void reverseLines(ifstream& input) {      // call 2
    string line;
    if (getline(input, line)) {              // "Violets are blue."
        reverseLines(input);                // make call 3 and wait
    }
}

void reverseLines(ifstream& input) {      // call 3
    string line;
    if (getline(input, line)) {              // "CS 106X"
        reverseLines(input);                // make call 4 and wait
    }
}
```



We Learned About Object-Oriented Programming

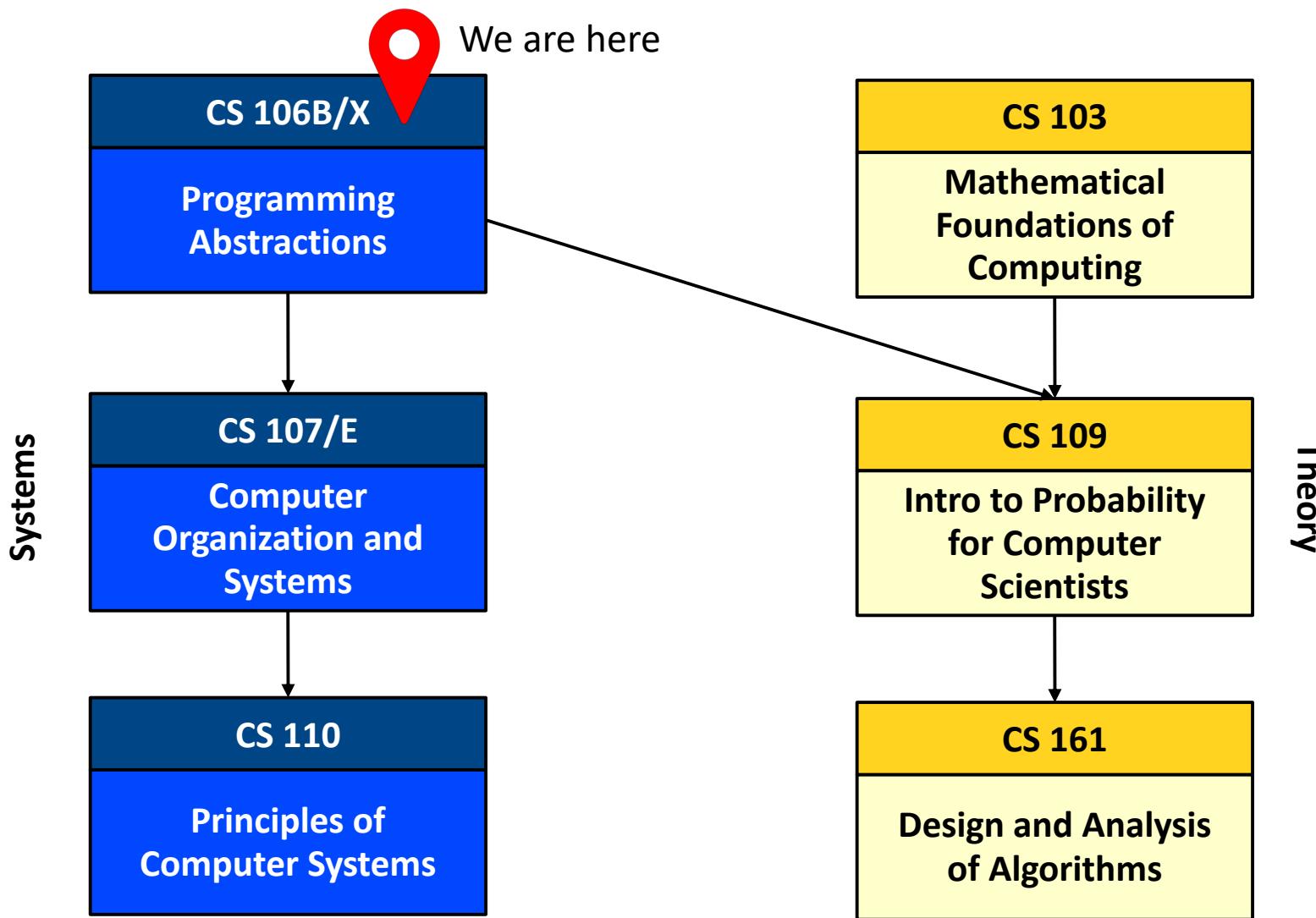
- A bit about **object-oriented** programming
 - member variables (state), member functions (behavior)
 - abstraction
 - private and public access
 - constructors and destructors
 - interface (.h) versus implementation (.cpp)
 - const
 - operator overloading
 - inheritance and polymorphism
 - template classes



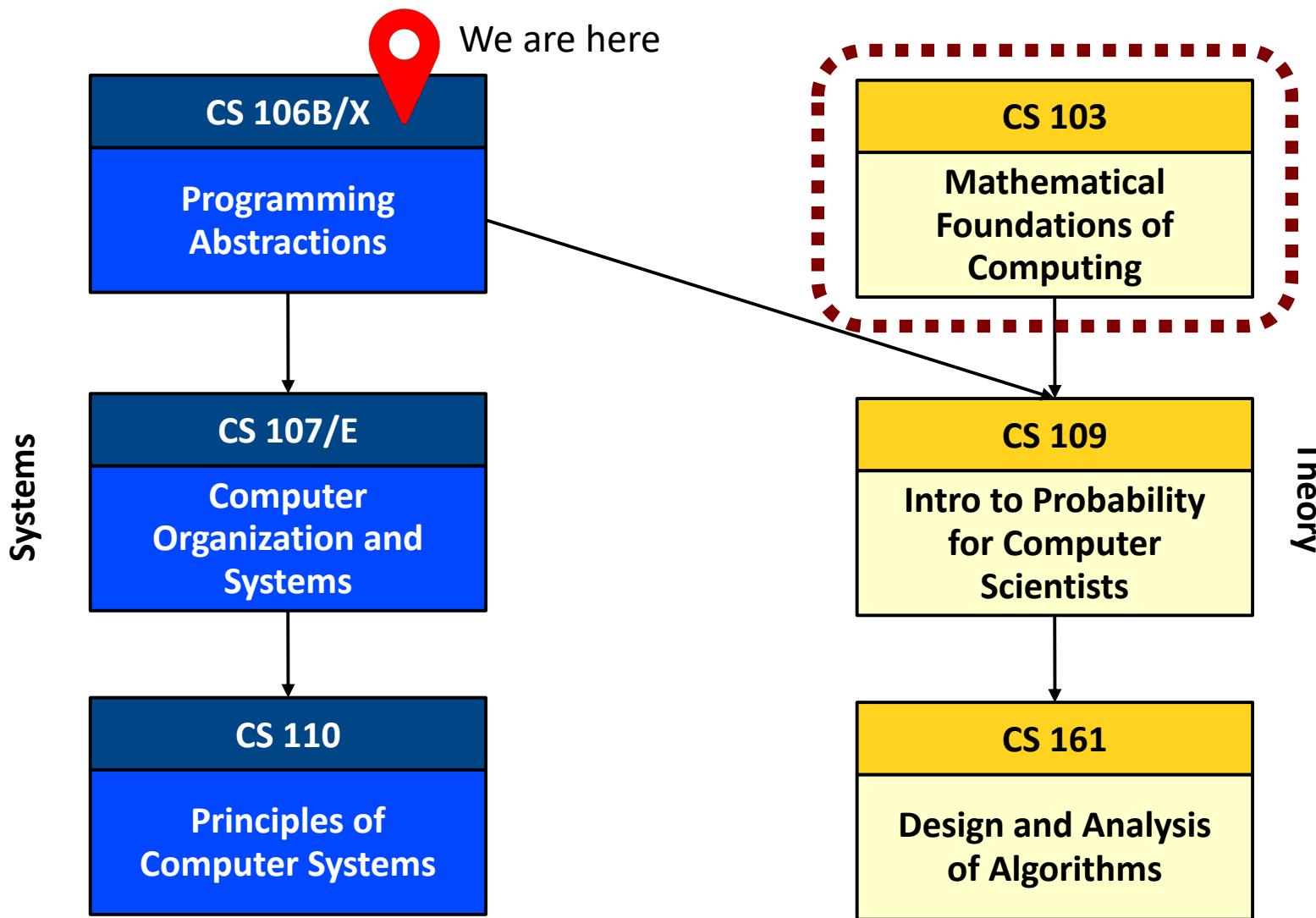
Plan For Today

- Recap: CS 106X
- **Next Steps**
 - Parting Words
 - Ask-Anything
 - Course Evaluations

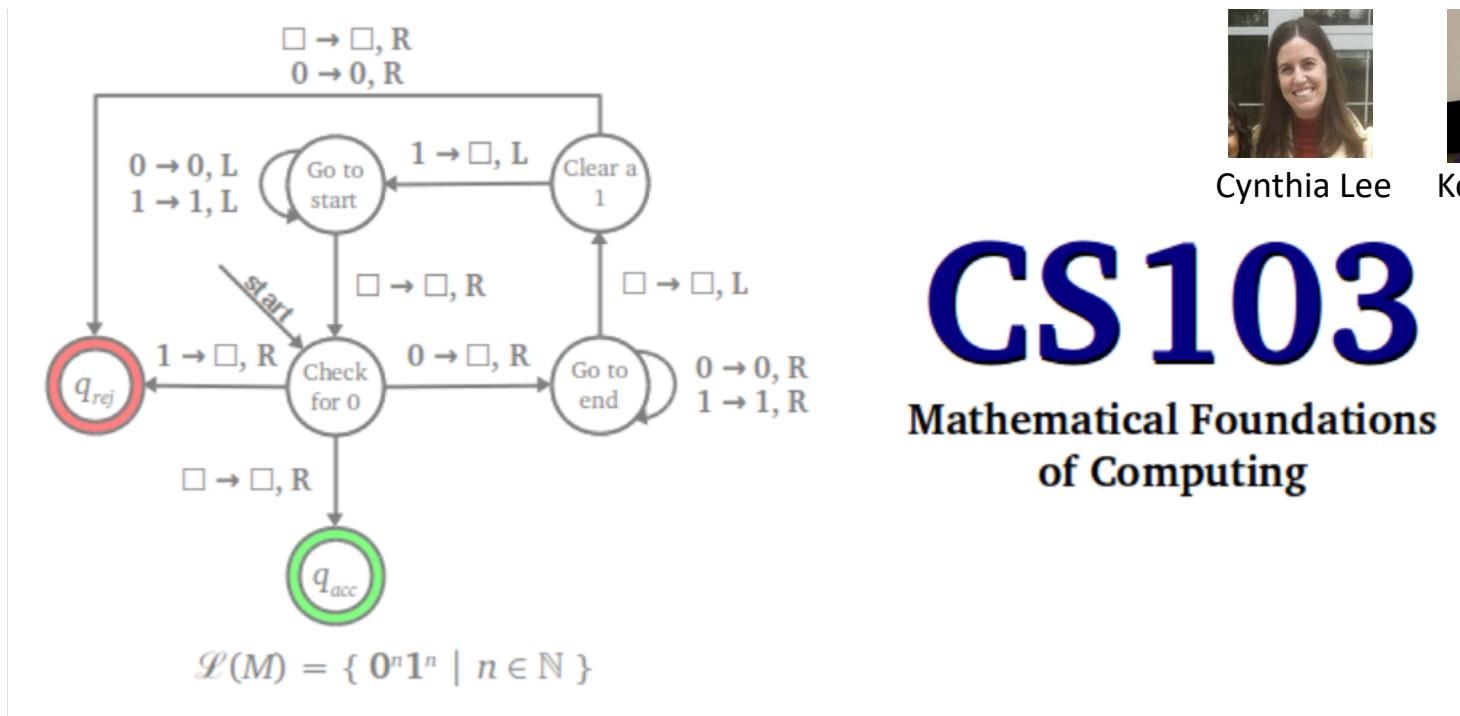
What's Next?



What's Next?



CS 103



- Can computers solve any problem?
 - How can we prove this?
 - What makes some problems harder to solve than others?



Cynthia Lee

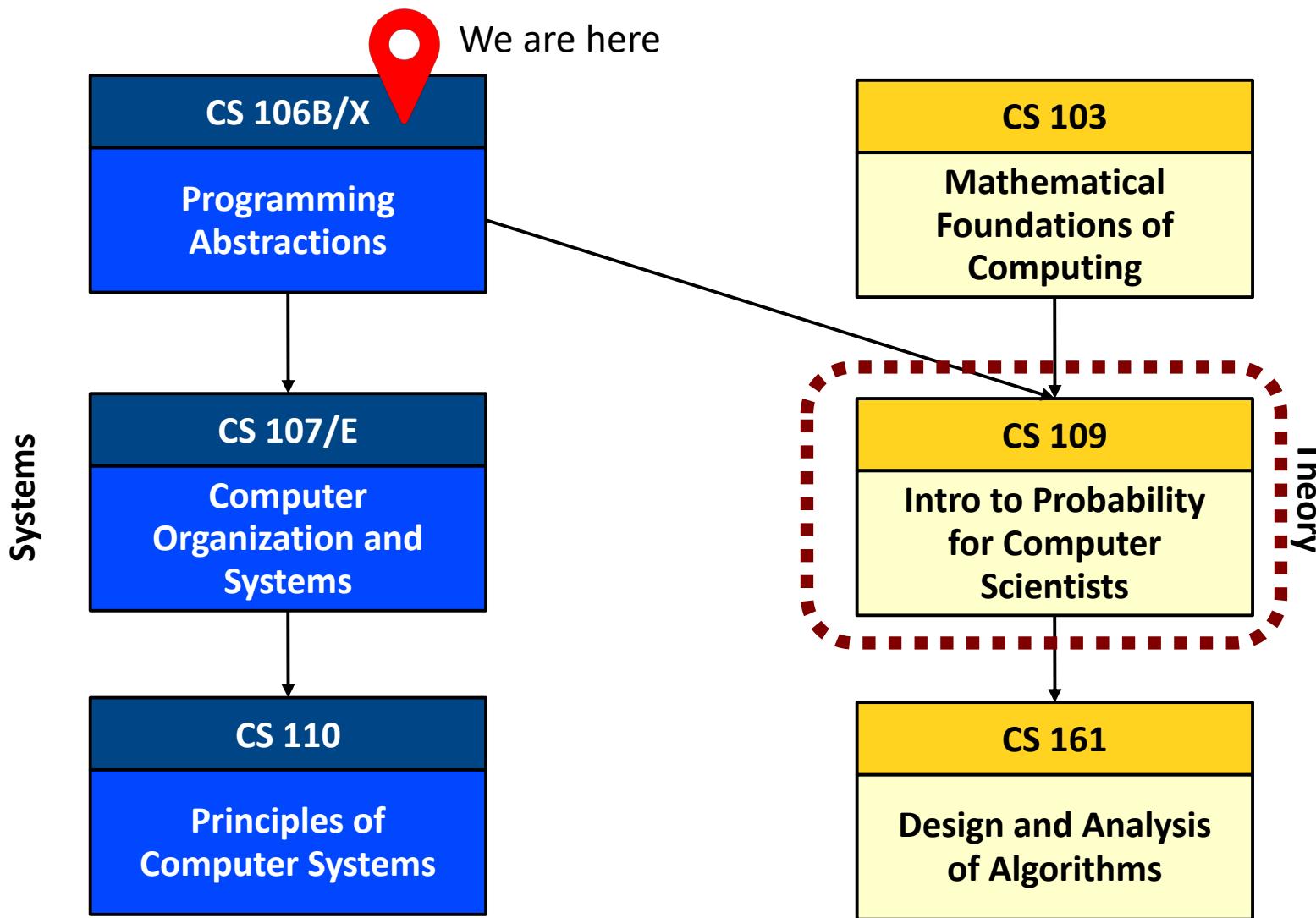


Keith Schwarz

CS103

Mathematical Foundations of Computing

What's Next?



CS 109

- How can we better model and analyze data?
- How can we program computers to learn?
- How can we predict the likelihood of certain outcomes?



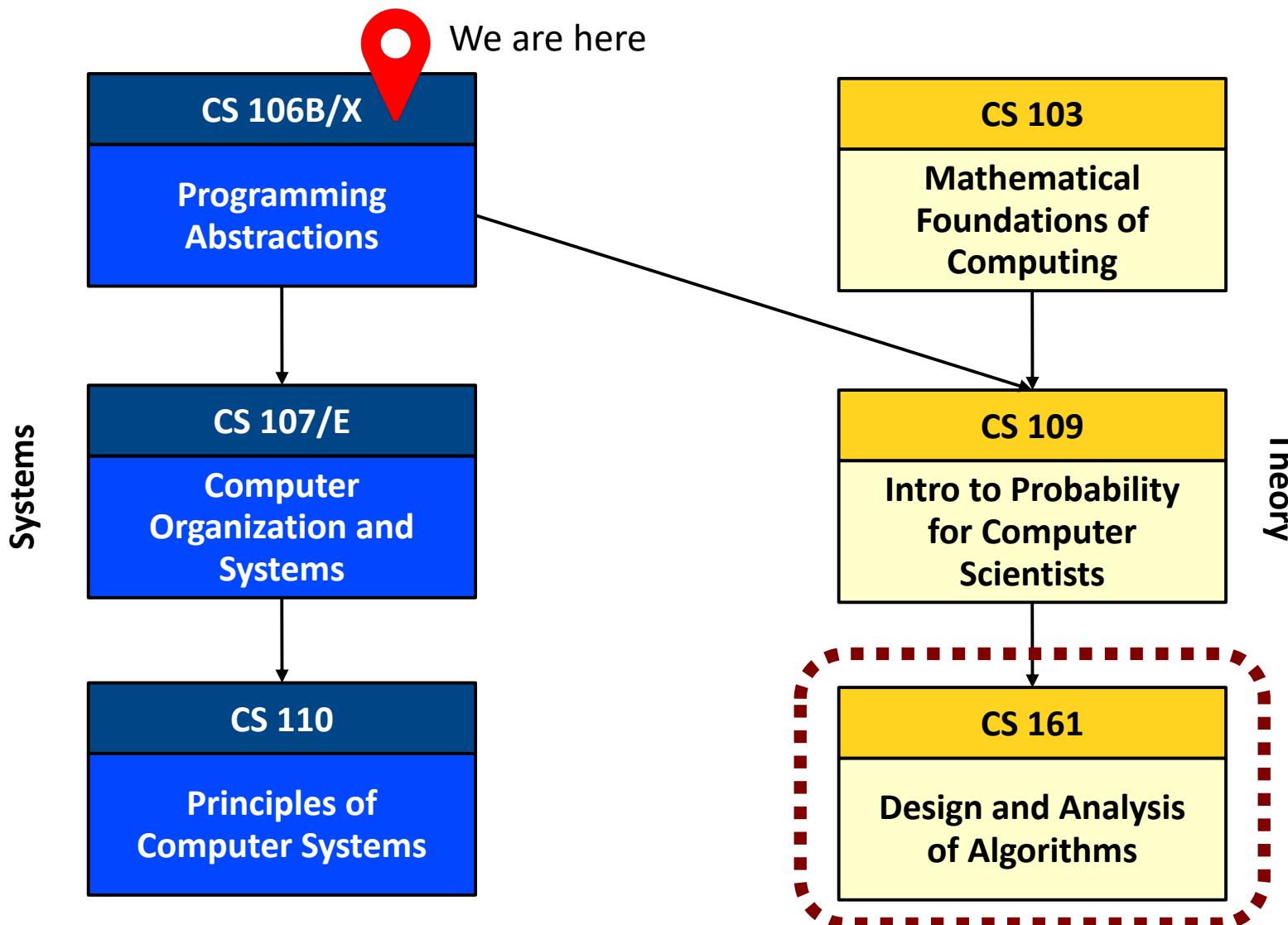
Chris Piech



Mehran Sahami



What's Next?



CS 161

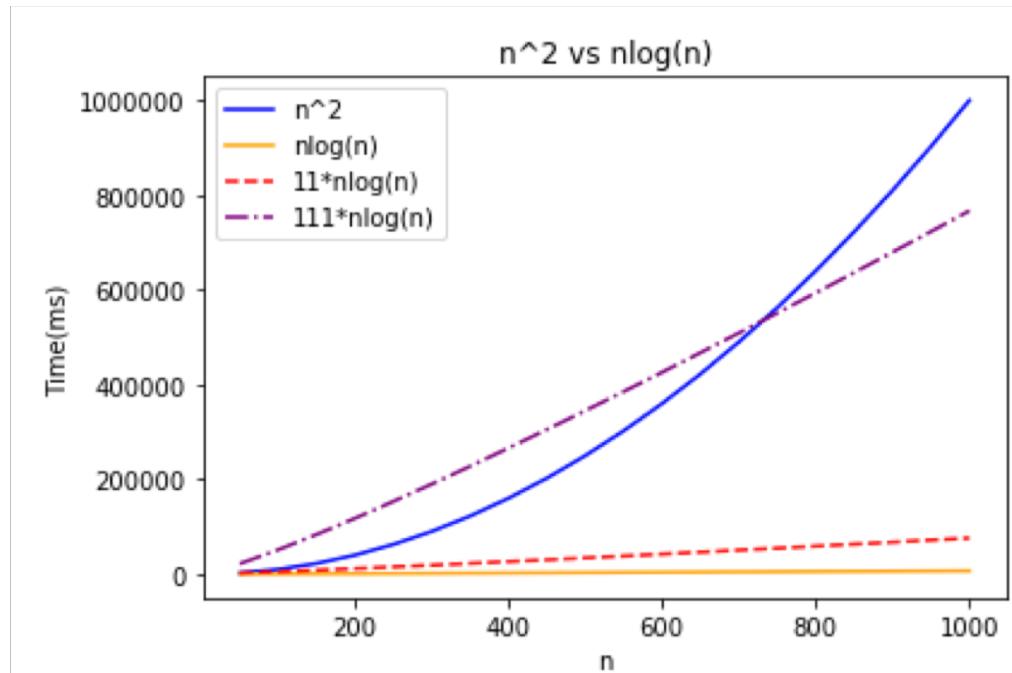
- How can we measure the efficiency of algorithms?
- How can we design our own algorithms?
- What useful algorithms are there?
- How can we prove that algorithms are correct?



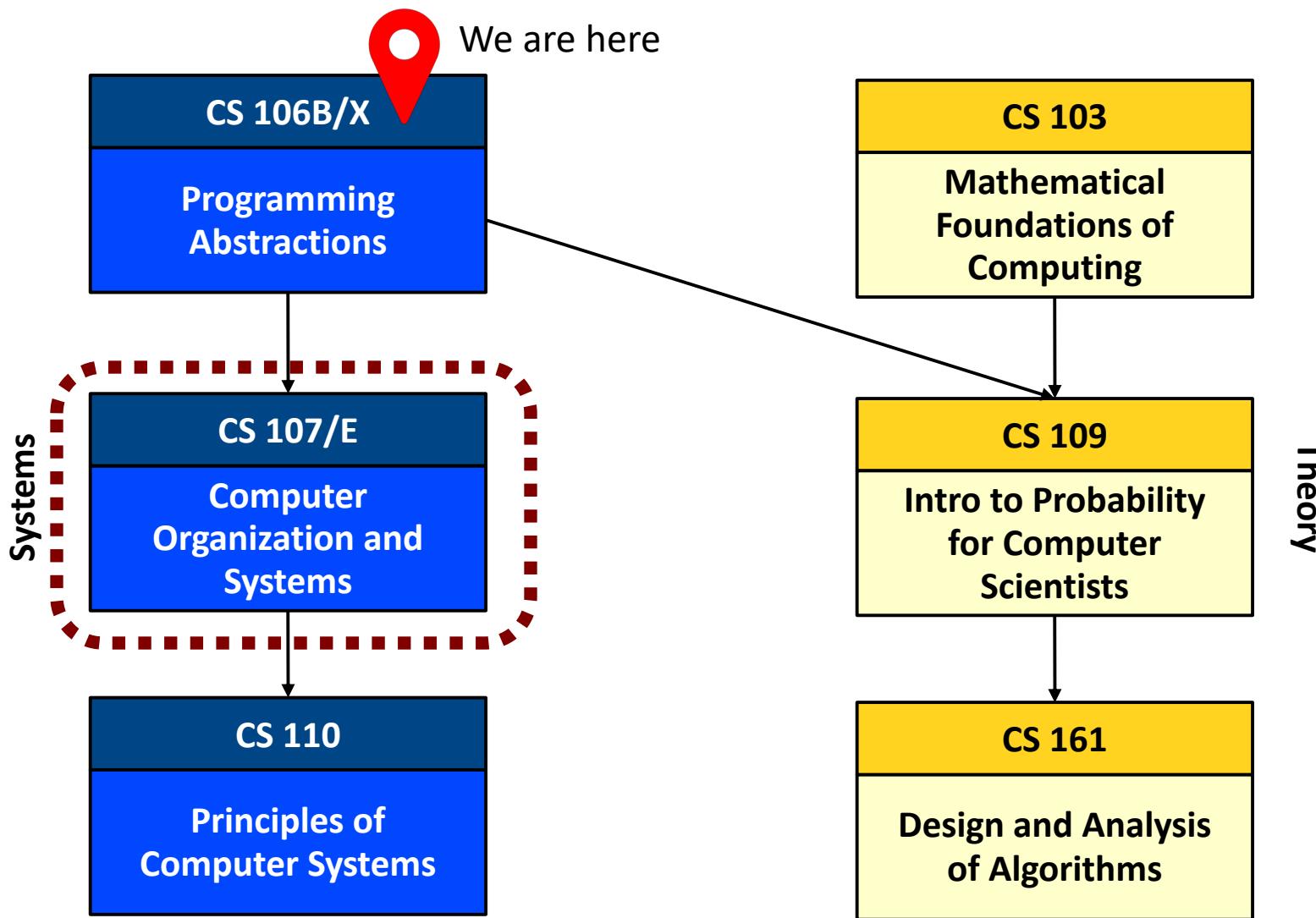
Aviad
Rubinstein



Mary Wootters



What's Next?



CS 107

- How do computers represent data under the hood?
- How can we effectively manage bits, bytes, and memory?
- How does a processor execute our code?



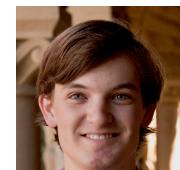
Julie Zelenski



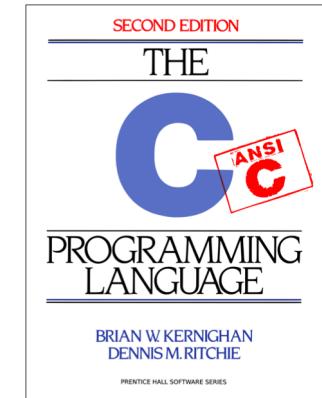
Chris Gregg



Cynthia Lee



Nick Troccoli



CS 107E

- **CS 107E**

- develop for a **Raspberry Pi**
- similar concepts, but in a setting of embedded devices
- entry by **application** (limited enrollment)



Julie Zelenski



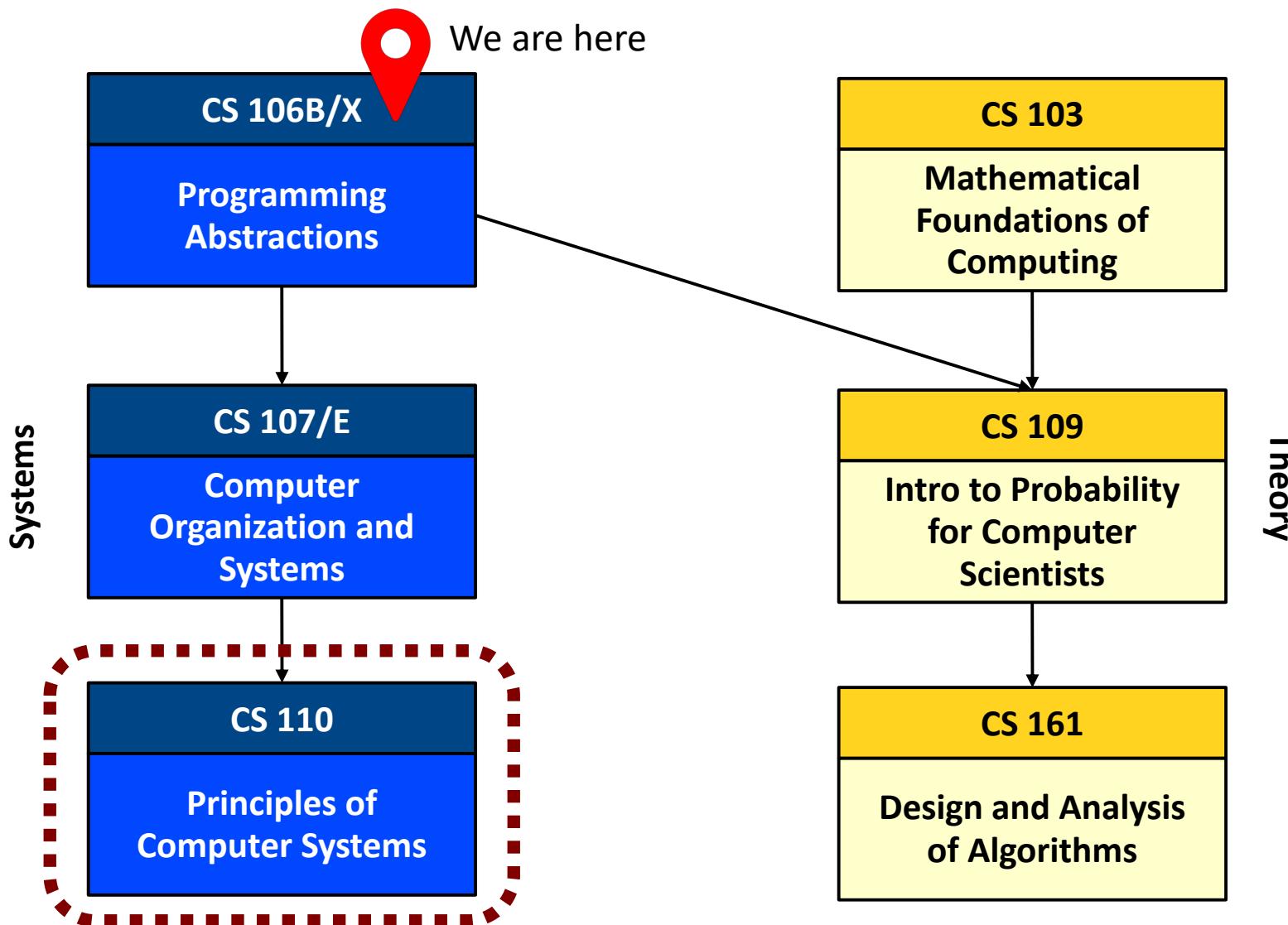
Chris Gregg



Pat Hanrahan



What's Next?



CS 110

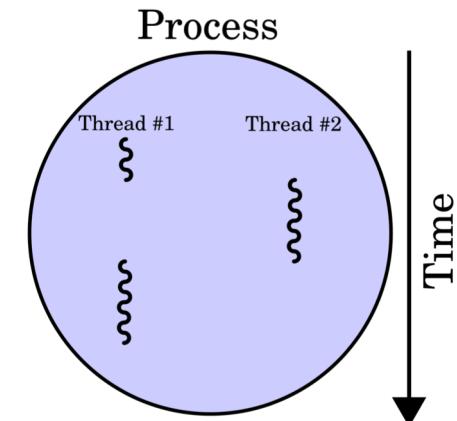
- How can we implement multithreading in our programs?
- How can multiple programs communicate with each other?
- How can we implement distributed software systems to do things like process petabytes of data?
- How can we maximally take advantage of the hardware and operating system software available to us?



Jerry Cain



Chris Gregg



Other CS Courses

- **CS 108:** Object-Oriented System Design
 - how to build large OO applications in Java; testing; GUIs; threads
- **CS 142:** Web Development
- **CS 221:** AI: Principles and Techniques
- **CS 147:** Human-Computer Interaction (HCI)
- **CS 181:** Computers, Ethics, and Public Policy
 - piracy, hacking, online privacy, wiretapping, ...
- **CS 193:** Special Topics
 - mobile apps, web dev, big data, ...
- **CS 40/41/42/92SI:** Student Initiated (JS, web, Python, ...)



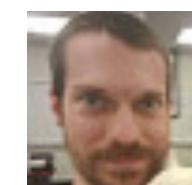
Patrick Young
(CS 108)



Christina Wodtke
(CS 247; CS 377G)



Emma Brunskill
(CS 234)



Marty Stepp
(CS 193A)



Percy Liang
(CS 221)

cs9: job prep

Studying CS

- Major or Minor: <https://cs.stanford.edu/degrees/ug/>
- Coterminal Degree: <https://cs.stanford.edu/degrees/ug/Coterm.shtml>
- other related majors:
 - **Symbolic Systems** (CS + psychology + linguistics + AI + Human-Computer Interaction)
 - **MCS** (Mathematical and Computational Science)
 - **MS&E** (Management Science and Engineering)
 - **EE** (Electrical Engineering)

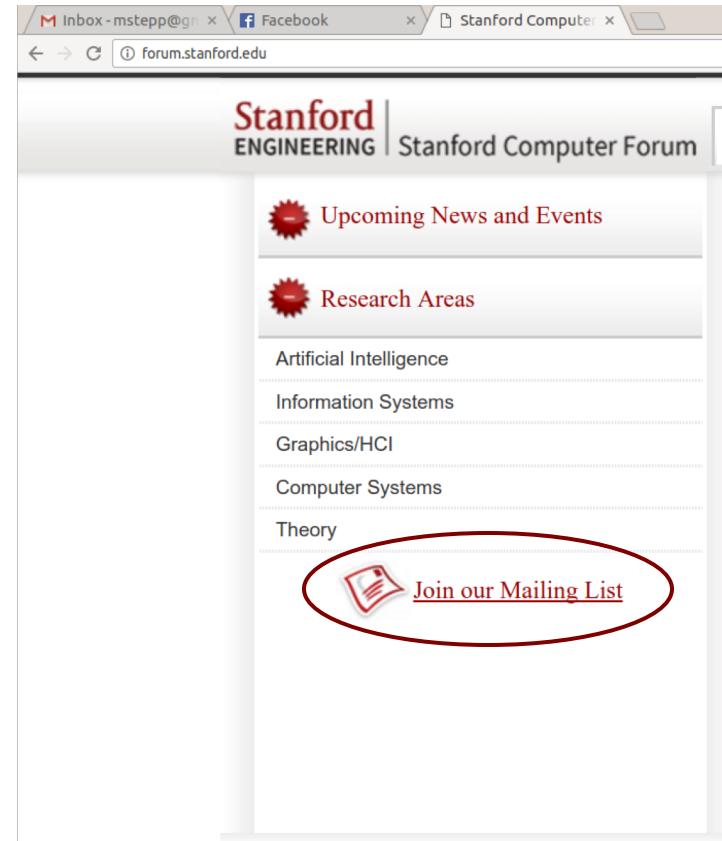
Outside of Stanford

- Online courses
 - Coursera, Udacity, edX
 - Stanford CS lectures on YouTube
- Online tutorials
 - Codecademy
 - Khan Academy
 - Stack Overflow
 - W3 Schools
- Practice coding
 - TopCoder
 - Project Euler
 - CodeFights
 - HackerRank
 - CodeStepByStep :)



Internships

- You could get a summer job based on what you learned this year.
 - A lot of CS interview questions are about 106X-esque topics:
 - pointers / linked lists, recursion, binary trees, graphs, ...
- Computer Forum (forum.stanford.edu)
 - Job Fairs twice a year
 - Mailing list for job opportunities
 - Many companies would want to hire you ☺



Freshman internships

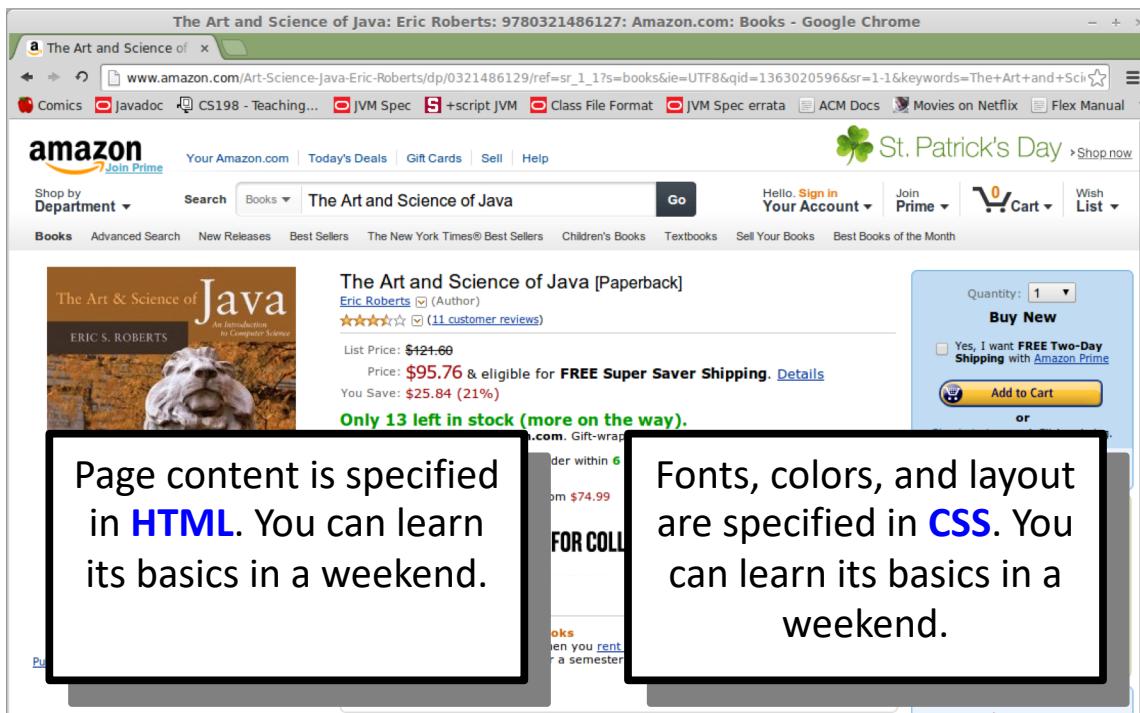
- Microsoft Explore
 - <https://careers.microsoft.com/students/explore>
- Google Engineering Practicum
 - <https://careers.google.com/jobs#!t=jo&jid=/google/engineering-practicum-intern-summer-2018>
- Google Summer of Code
 - <https://developers.google.com/open-source/gsoc/>
- Facebook University for Engineering
 - <https://m.facebook.com/careers/university/fbueng>
- Twitter Freshman Engineering Internships
 - <https://careers.twitter.com/en/university.html>
- Intel Early Internship for Software Engineering (IRISE)
 - <http://www.intel.com/content/www/us/en/jobs/locations/united-states/students/internships/>
- Amazon University Internships
 - <http://www.amazon.jobs/team/university-internships>
- Pinterest Engage Internship
 - <https://careers.pinterest.com/careers/details/>

Freshman internships 2

- **Khan Academy internships**
 - <https://www.khanacademy.org/careers/interns>
- **NSA internships**
 - <https://www.nsa.gov/careers/>
- **Breakout Mentors** (mentor students age 9-14 as they learn programming)
 - <http://breakoutmentors.com/>
- possible others
 - <http://www.computersciencedegreehub.com/internships-fortune-500-companies/>
- **Quora:** What tech companies offer internships to freshmen?
 - <https://www.quora.com/What-tech-companies-offer-internships-to-freshmen>
- **Quora:** What can I do to get an internship over the summer?
 - <https://www.quora.com/As-a-college-freshman-majoring-in-CS-what-can-I-do-to-potentially-get>

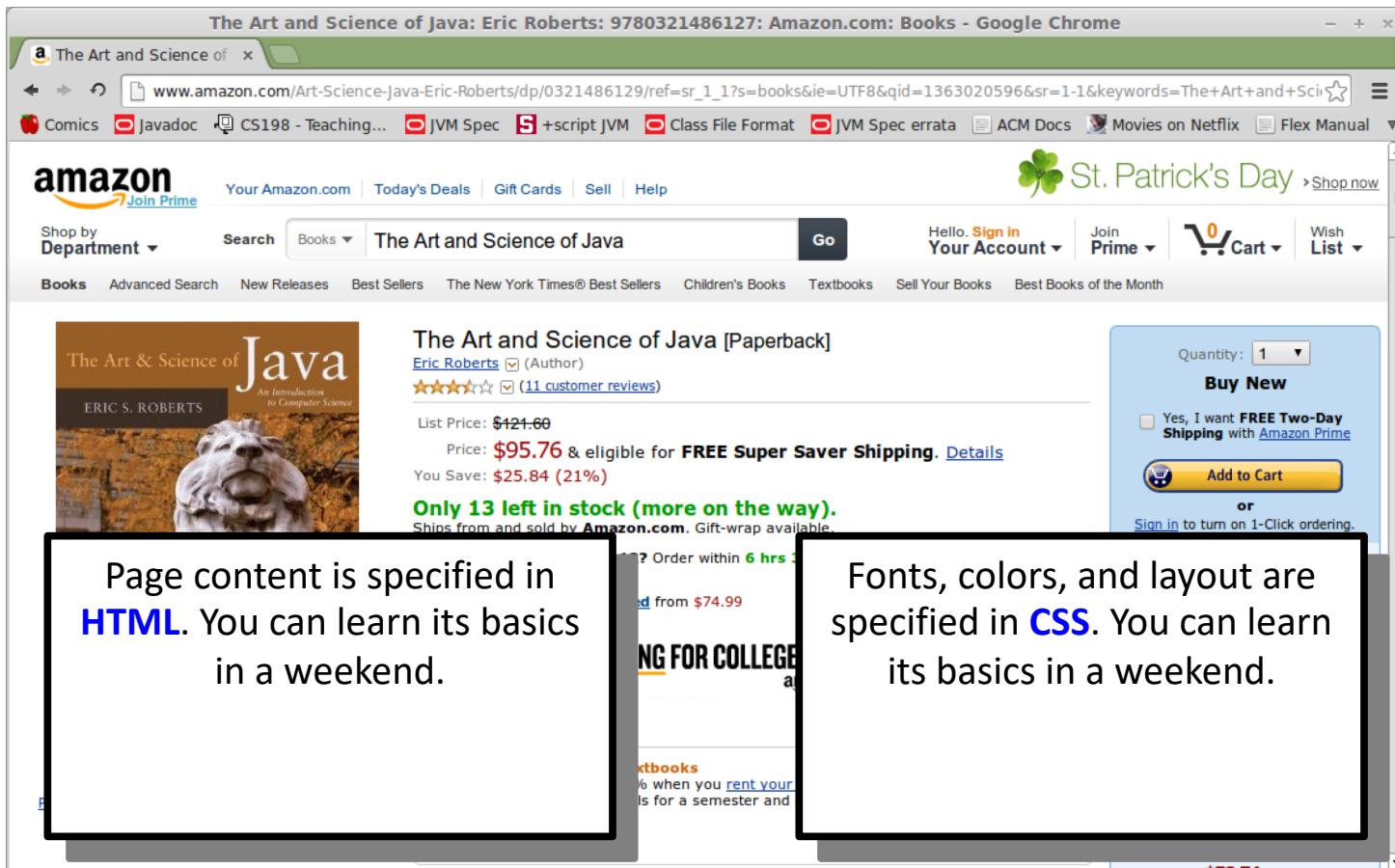
Teach yourself

- If there is a CS topic that interests you, go learn more about it!
 - Example: Learn how to make a web site (resume? blog? photos?)
 - Example: Rewrite 106A HW in C++; rewrite 106X HW in Java/Python
 - Example: grab cool data from reddit [/r/datasets](#) and process it



Teach yourself HTML/CSS

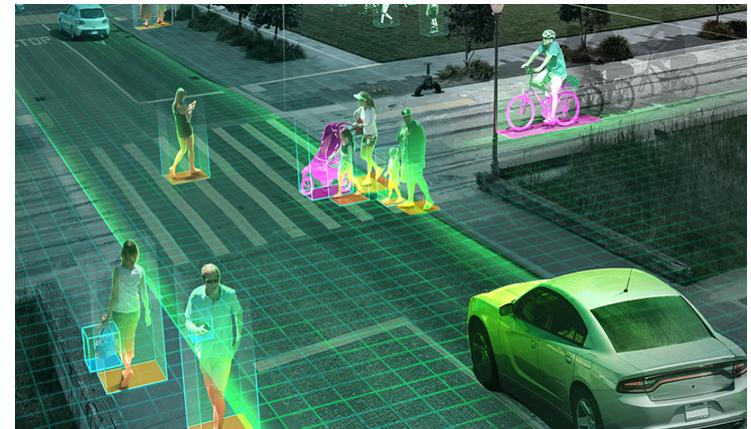
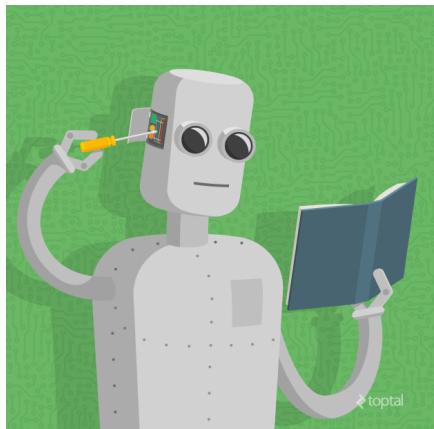
- **Challenge:** Write yourself a simple web page and resume in HTML.
 - Post it on Stanford web space! <http://stanford.edu/~yourUserName/>



Read about CS topics

- **Challenge:** Pick a CS topic or sub-field, and go read about the basics of it over the break.
 - *suggestion:* Pick a topic that aligns with a Stanford CS "track."

Machine Learning, Neural Networks, Computer Vision, ...



Apply to be an SL

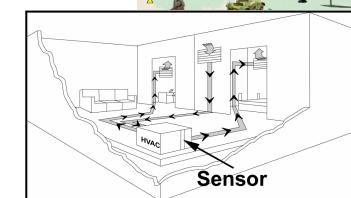
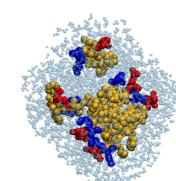
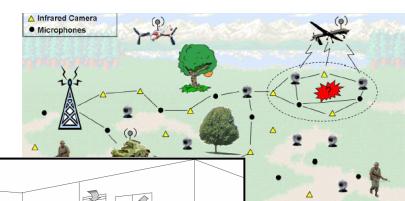
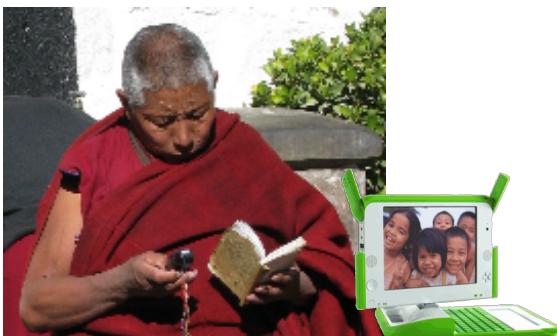


- Apply to be a **Section Leader!** cs198.stanford.edu
- Thanks to our awesome section leaders this quarter, and especially to our head TA, **Zach Birnholz!**



Computer Science Is Everywhere

- security
- quantum computing
- robotics
- graphics and vision
- networking
- data mining
- developing world
- neurobiology
- ... many more



Plan For Today

- Recap: CS 106X
- Next Steps
- **Parting Words**
- Ask-Anything
- Course Evaluations

Anyone can be a
computer scientist.

Computer science
is using computing
to solve problems.

Anyone is
welcome to be a
computer scientist.

We hope CS106X
excited you about
learning more
computer science!

Plan For Today

- Recap: CS 106X
- Next Steps
- Parting Words
- **Ask-Anything**
- Course Evaluations

Ask-Anything

“What are differences
between cs major tracks?”

Ask-Anything

“What are differences

Each track has the same fundamentals, but lets you branch off based on interest. For example, if you want to learn about large systems like compilers and operating systems, “Systems” is a good fit. If you’re interested in the role of computers in human life and how we interact with them, HCI is a good fit.

Ask-Anything

“What are some foolproof ways to come up with good tests?”

Ask-Anything

“What are some foolproof ways to come up with good tests?”

Pretend you’re an adversary; what ways could you try to provide invalid input, or use the functions in an unintended manner? Try to cause unintended behavior.

Ask-Anything

“How to get involved in
research/summer programs
at Stanford?”

Ask-Anything

“How to get involved in research/summer programs

CURIS (Undergraduate Research in CS) is a year-round program that matches students with departmental research projects. Visit <https://curis.stanford.edu>, or email/go talk to professors who are doing work you're interested in at their office hours! Be proactive and reach out to people you're interested in working with.

Ask-Anything

“How would you recommend
to prepare for technical
coding interviews?”

Ask-Anything

“How would you recommend to prepare for technical coding interviews?”

Don’t just practice coding problems; practice explaining your ideas as you come up with them, ideally at a whiteboard. Interviewers look at how you develop and explain your ideas, not just whether you get the right answer.

Ask-Anything

“What makes a good computer scientist?”

Ask-Anything

“What makes a good computer scientist?”

1. The knowledge of how computing fits in an interdisciplinary world.
2. An eagerness to learn new things.

Plan For Today

- Recap: CS 106X
- Next Steps
- Parting Words
- Ask-Anything
- **Course Evaluations**

Course Evaluations

We hope you can take the time to fill out the end-quarter CS 106X course evaluation. We sincerely appreciate any feedback you have about the course, and read every piece of feedback we receive. We are always looking for ways to improve!

Thank you!