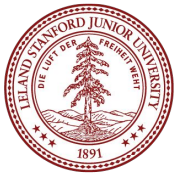
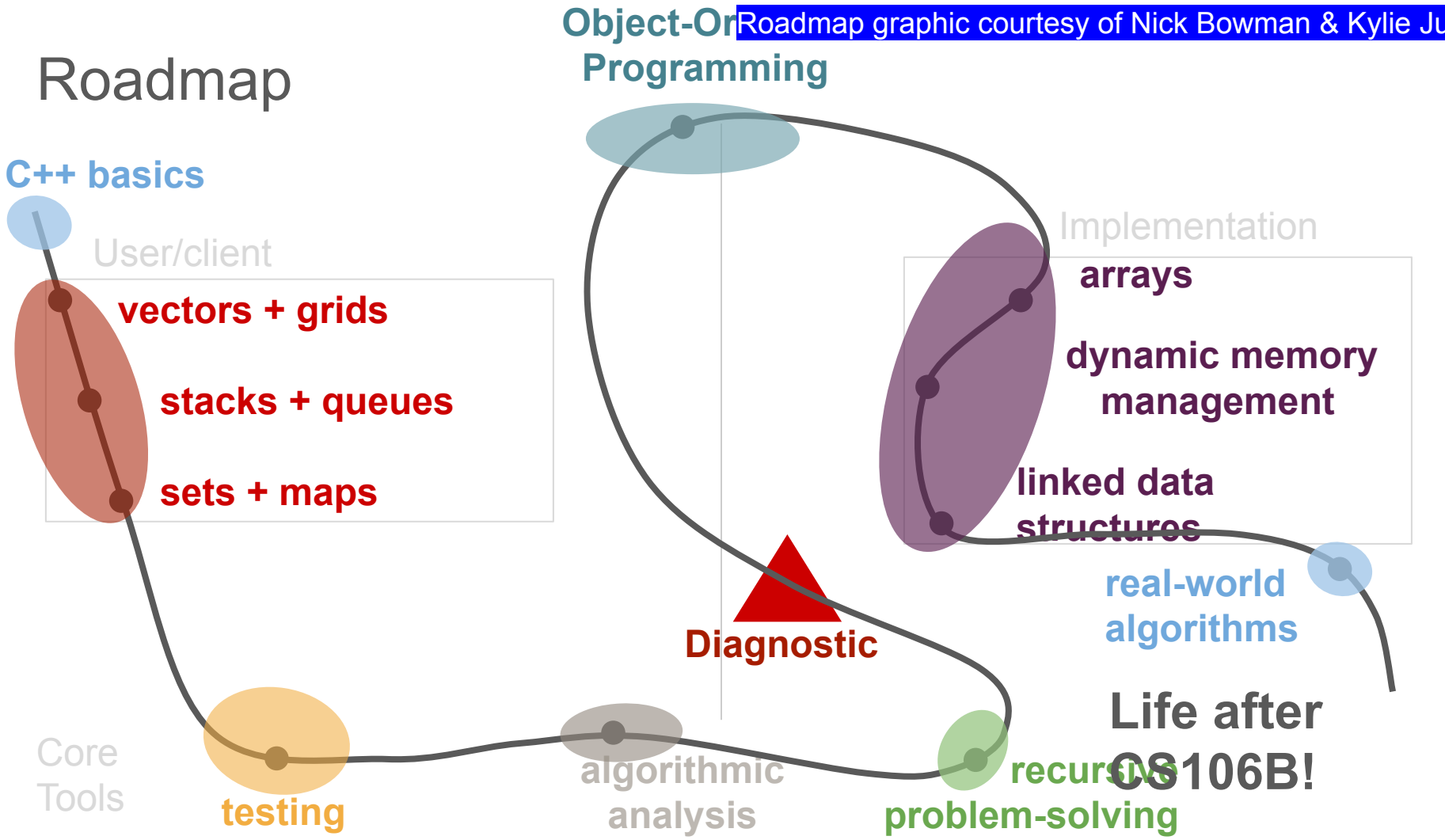
Life after CS106B

What are you up to next in life?

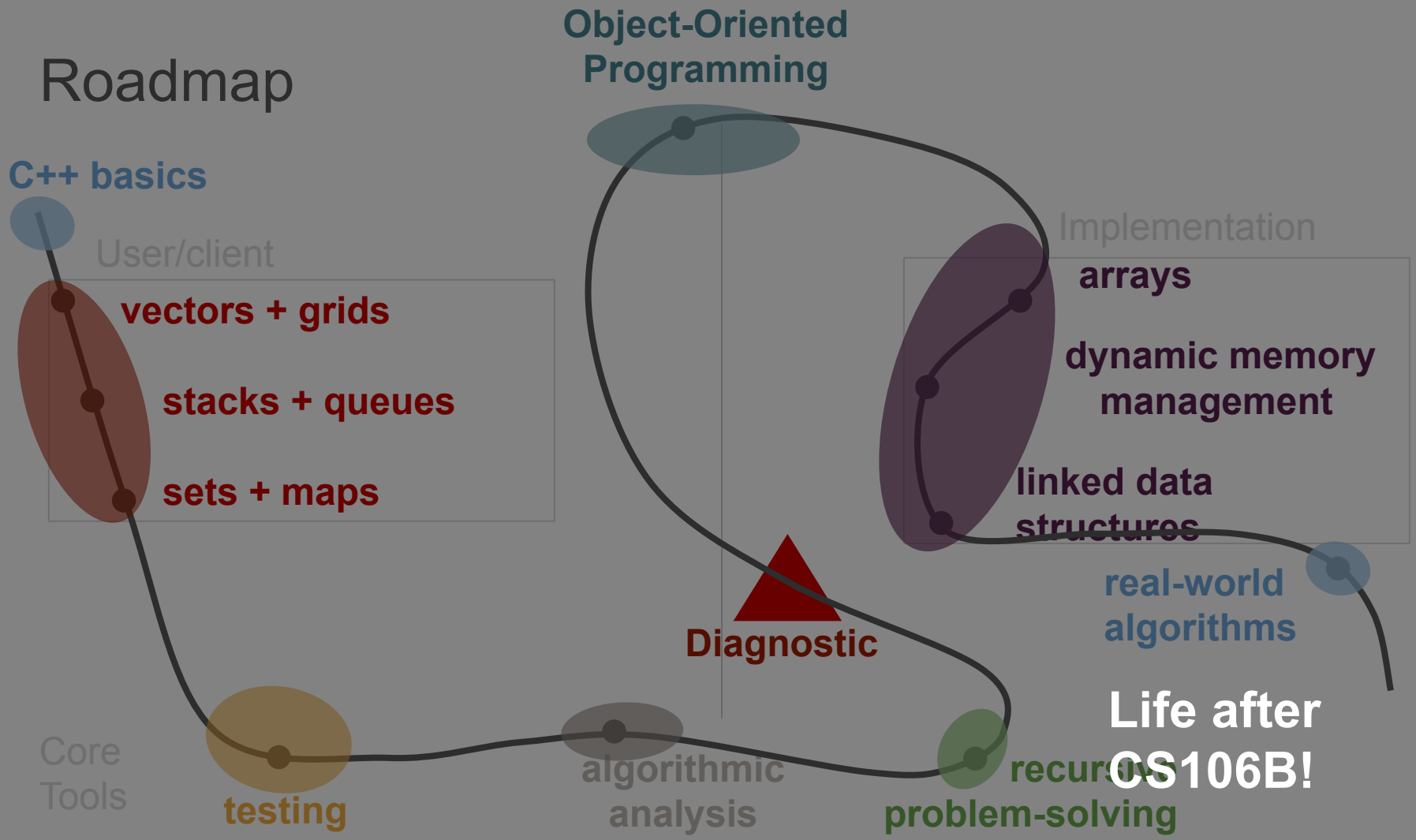
(put your answers in the chat)



Roadmap


Object-Or
Programming
Roadmap graphic courtesy of Nick Bowman & Kylie Jue
C++ basics
User/client
vectors + grids
stacks + queues
sets + maps
Implementation
arrays
dynamic memory management
linked data structures
real-world algorithms
Diagnostic
Life after CS106B!
Core Tools
testing
algorithmic analysis
recursive problem-solving

Roadmap


Object-Oriented Programming
C++ basics
User/client
vectors + grids
stacks + queues
sets + maps
Implementation
arrays
dynamic memory management
linked data structures
real-world algorithms
Diagnostic
Core Tools
testing
algorithmic analysis
recursive problem-solving
Life after CS106B!

Today's topics

1. Course Recap
2. Areas of Computer Science to Continue to Explore
3. Software Engineering

Goals for this Course

Learn how to model and solve complex problems with computers.

- Explore common abstractions for representing problems.
- Harness recursion and understand how to think about problems recursively.
- Analyze different approaches for solving problems: efficiency, optimization, and ethics.

Course Recap

[everything!]

What you've learned during this course

You've learned….

- C++ fundamentals (types, structs, pass by value/reference)
- Good testing practices and Debugging
- Console programs and C++ strings
- ADTs: Vectors, Queues, Stacks, Maps, Sets, Priority Queues, HashMaps, HashSets
- Breadth-First and Depth-First Search
- Big-O Notation and Algorithmic Analysis
- Social Impact Analysis of Algorithms and Ethics
- Recursion: Recursive problem-solving, fractals, backtracking

But wait, there's more!!

- Classes and Object-Oriented Programming
- Dynamic memory and pointers
- Arrays and linked data structures (trees, linked lists, graphs)
- Binary heaps
- Sorting algorithms (mergesort, quicksort, selection sort, insertion sort)
- Binary trees and Huffman encoding trees
- Hashing
- Parallel computing
- Abstractions!

What you've **learned** during this course

- C++ fundamentals (types, structs, pass by value/reference)
- Good testing practices & Debugging
- Console programs and C++ strings
- ADTs: Vectors, Queues, Stacks, Maps, Sets, Priority Queues, HashMaps, HashSets
- Breadth-First and Depth-First Search
- Big-O notation and algorithmic analysis
- Social Impact Analysis of Algorithms and Ethics
- Recursion: Recursive problem-solving, fractals, backtracking
- Classes and Object-Oriented Programming
- Dynamic memory and pointers
- Arrays and linked data structures (trees, linked lists, graphs)
- Binary heaps
- Sorting algorithms (mergesort, quicksort, selection sort, insertion sort)
- Binary trees and Huffman encoding trees
- Hashing
- Parallel computing
- Abstractions!

What you've **made** during this course

Assignment 1: C++ legs

NATIONAL ARCHIVES

Blogs · Bookmark/Share · Contact Us

Search Archives.gov Search

RESEARCH OUR RECORDS | VETERANS' SERVICE RECORDS | EDUCATOR RESOURCES | VISIT US | AMERICA'S FOUNDING DOCUMENTS

Research Our Records

Home > Research Our Records > Census Records > Soundex System

Census

About Census Records
Search Census Records Online
1940 Census
1930 Census FAQs
1850-1930 Census Clues
1790-1840 Census Clues
Nonpopulation Census
1935 Business Census
Indian Census Rolls
Presidents in the US Census Records
Census Links by Year

Resources

Soundex Coding System
Blank Census forms and charts
Order Census Copies Online

Soundex System

The Soundex Indexing System

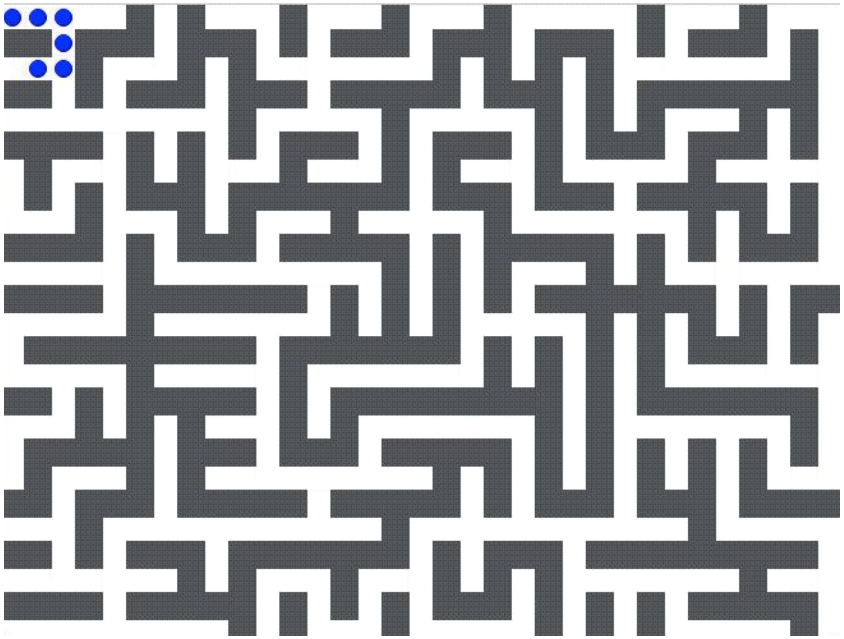
Updated May 30, 2007

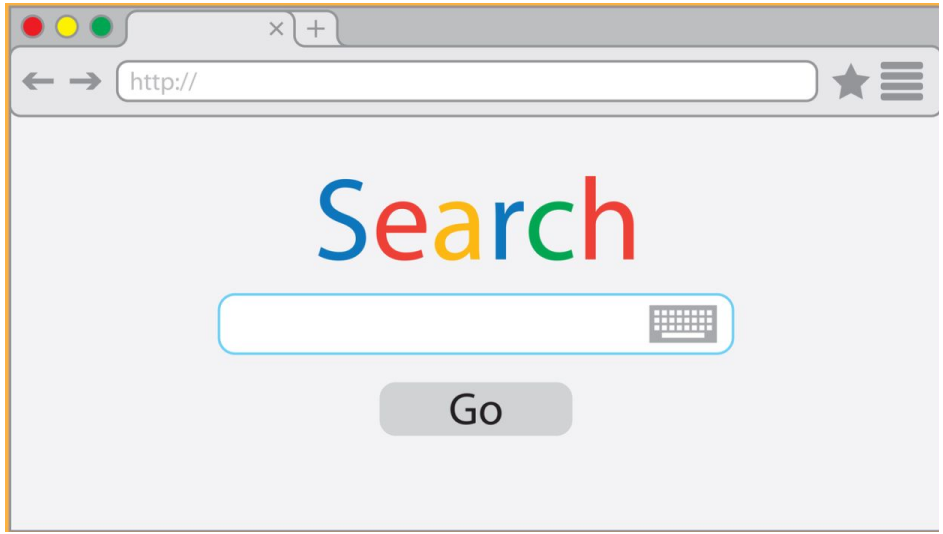
To use the census soundex to locate information about a person, you must know his or her full name and the state or territory in which he or she lived at the time of the census. It is also helpful to know the full name of the *head of the household* in which the person lived because census takers recorded information under that name.

The soundex is a coded surname (last name) index based on the way a surname sounds rather than the way it is spelled. Surnames that sound the same, but are spelled differently, like SMITH and SMYTH, have the same code and are filed together. The soundex coding system was developed so that you can find a surname even though it may have been recorded under various spellings.

To search for a particular surname, you must first work out its code.

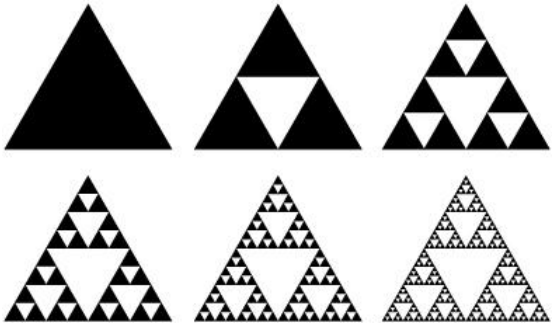
Assignment 2: Fun with Collections



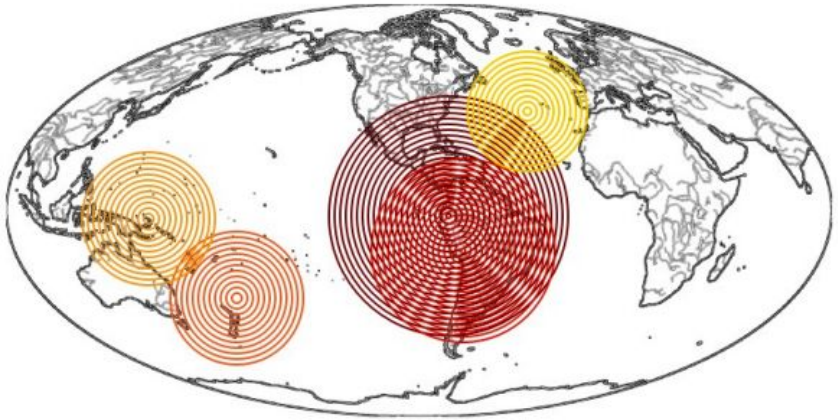


Assignment 3: Recursive Problem-Solving

{ }



Assignment 4: Priority Queue

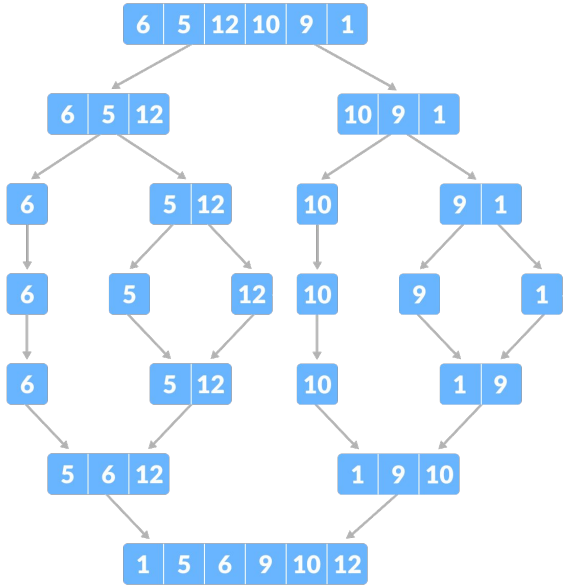


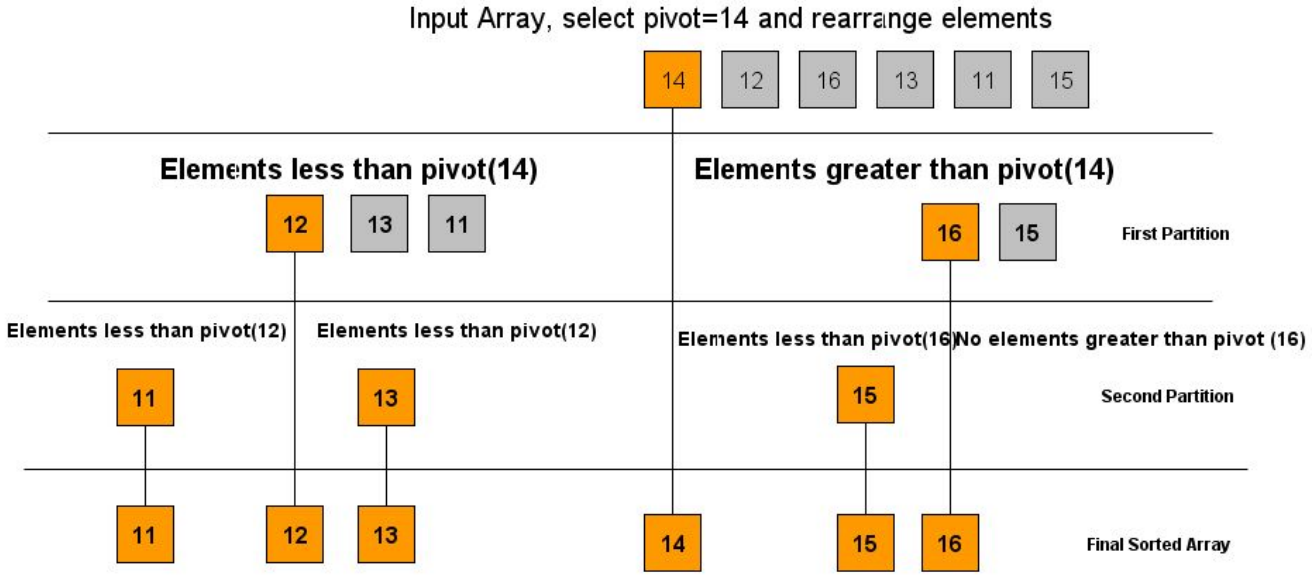
This tool displays the strongest recent earthquakes reported by the US Geological Survey
You can use the controls on the side of the window to select the time interval you're interested in. This visualizer will show the 5 strongest earthquakes within that interval.

Remember that the earthquake magnitude scale is logarithmic. An earthquake that is on magnitude in strength higher than another releases around 32 times as much energy.

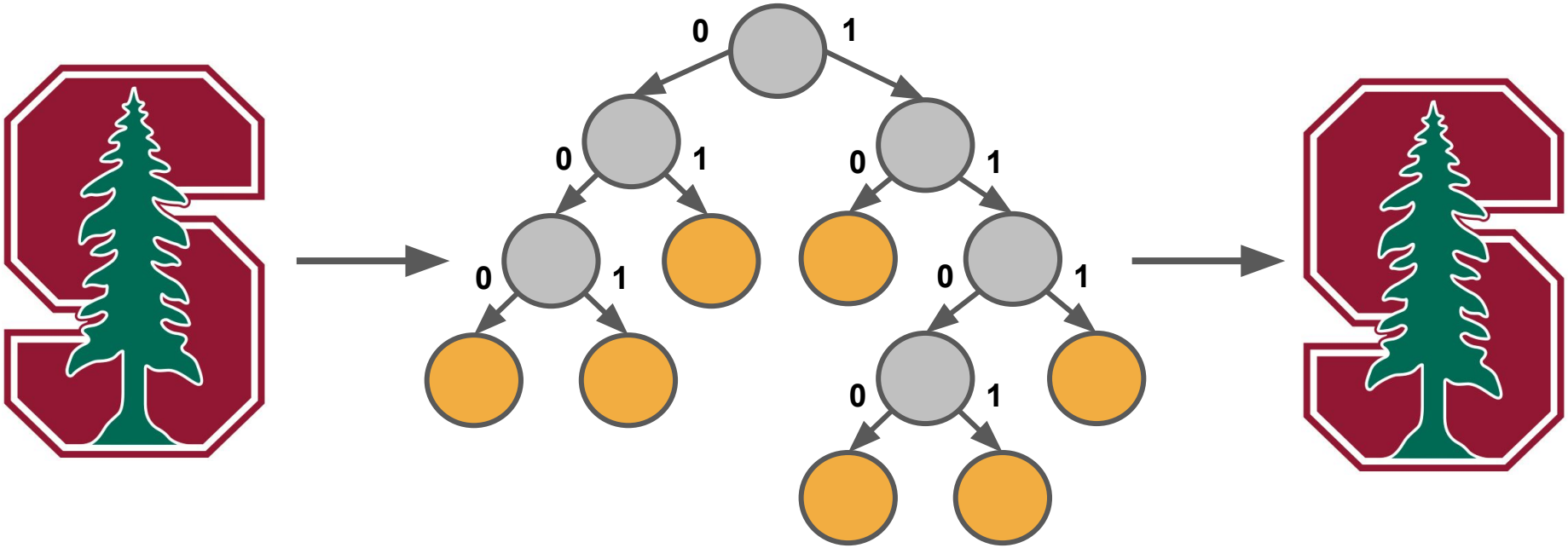
- Magnitude 7.5 115km ESE of Palora, Ecuador at 02:17:22 AM on Feb 22, 2019
- Magnitude 7 27km NNE of Azangaro, Peru at 12:50:41 AM on Mar 01, 2019
- Magnitude 6.4 116km SE of L'Esperance Rock, New Zealand at 07:46:14 AM on Mar 06, 2019
- Magnitude 6.4 49km NW of Namatanai, Papua New Guinea at 06:35:55 AM on Feb 17, 2019
- Magnitude 6.2 Northern Mid-Atlantic Ridge at 11:57:05 AM on Feb 14, 2019

Assignment 5: Linked Lists and Sorting





Assignment 6: Huffman Encoding



Computer science is more than just programming.

Questions you can start to answer

- What is possible with technology and code?
- What isn’t possible?
- How can I use programming to solve problems?
- What makes a “good” algorithm or data structure? Why?

Computer science is more than just programming!

These skills will make you better at whatever you choose to do in life!

So what comes next?

What exactly is computer science?

Fei-Fei Li
Artificial Intelligence

James Landay
Human/Computer Interaction

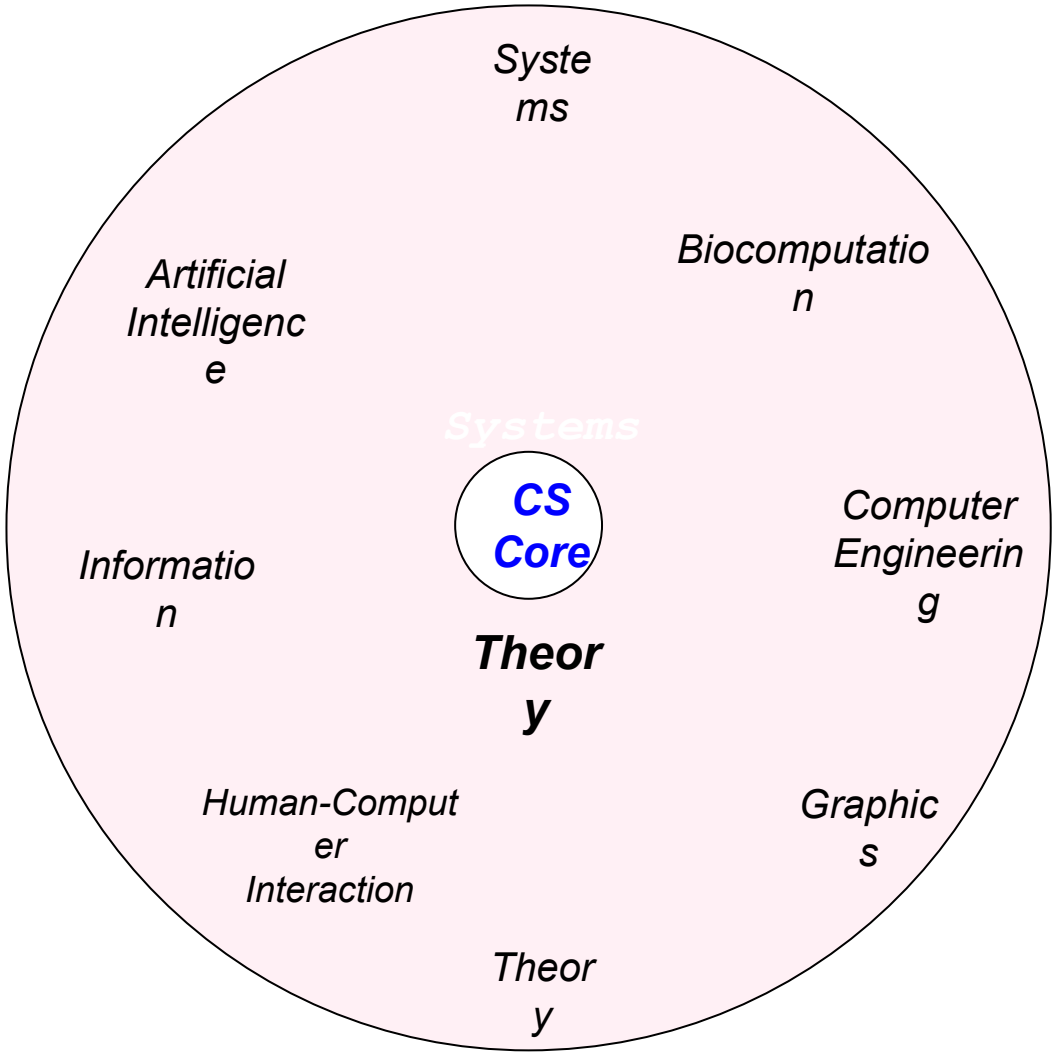
Chris Piech
Education, AI

Mary Wootters
Theoretical CS

Jeannette Bohg
Robotics

Pat Hanrahan
Graphics, Systems

How do I learn more about it?


Systems
Artificial Intelligence
Biocomputation
Systems
CS Core
Computer Engineering
Information
Theory
Human-Computer Interaction
Graphics
Theory

Computer Organization and Systems

What is the internal organization of memory in a computer?

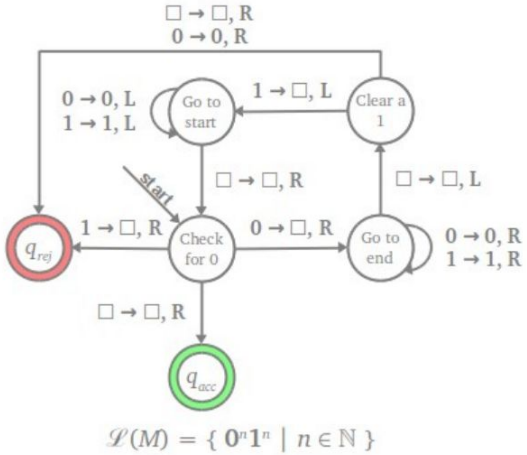
How do we bridge the dichotomy between high-level problem-solving and voltages in wires?

And why is this important to know?

Important Ideas in Computer Organization & Systems

- The nature of memory layout explains why computer security is so hard to get right.
- Computers are physical devices whose inner workings are visible even in higher-level languages.
- Compilers can sometimes rewrite recursive functions iteratively, giving you the best of both worlds.

Computational Theory and Mathematical Foundations of Computing



What are the fundamental limits of computing power?

How can we be certain about this?

Important Ideas in Discrete Math

- Some infinities are bigger than other infinities, and this has practical consequences.
- Tropes from Ancient Greek mythology can be made mathematically rigorous to prove limits on computing power.
- Abstract models of computation have applications in network drivers, user interfaces, compiler design, and text processing.

Probability for Computer Scientists

- Why are hash tables fast? Why are random binary search trees probably good?
- How do we encode data so that if bits get flipped in transit, the message still arrives?
- How do I explore big data sets and make sense of them?
- What is this whole machine learning thing, how does it work, and how do I do it?

What can I study within computer science?

Other CS Courses

Practical Programming Technologies

- Android Programming
- Client-Side Web Technologies
- iOS Programming
- iPhone and iPad Programming

Human-Computer Interaction

- How do you design software to be usable?
- What are the elements of a good design?
- How do you prototype and test out systems?

Learning Beyond College

- Some online resources (mostly in the form of free courses)
  - Codecademy
  - Coursera, edX, Udemy, and other MOOCs
  - Khan Academy
  - MIT Open Courseware

- Strategies for programming self-improvement
  - Write lots of code!
  - Work on a project you find inspiring
  - Find other people to collaborate with
  - Join an open-source project!

What should my academic path look like?

Thinking about studying more CS?

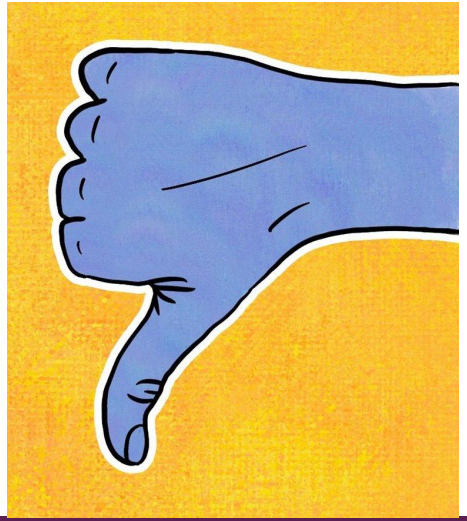
Good reasons to think about doing CS:

- I like the courses and what I'm doing in them.
- I like the people I'm working with.
- I like the impact of what I'm doing.
- I like the community.

Thinking about studying more CS?

Bad reasons to think about not doing CS:

- I'm good at this, but other people are even better.
- The material is fun, but there's nothing philosophically deep about it.
- I heard you have to pick a track,
  and I don't know what I want to do yet.
- What if 20 years later I'm just
  working in a cubicle all day and it's not fun
  and I have an Existential Crisis?



How do the skills we've learned translate to how CS is used in the technology industry?

How do the skills we've learned translate to how CS is used in the technology industry?

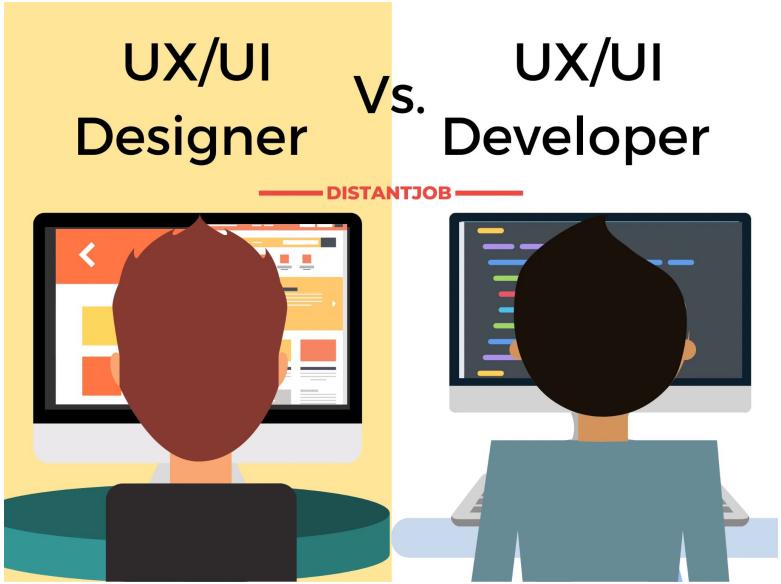
Like we said, there are *many different* jobs in CS.

- Product Managers

How do the skills we've learned translate to how CS is used in the technology industry?

Like we said, there are *many different* jobs in CS.

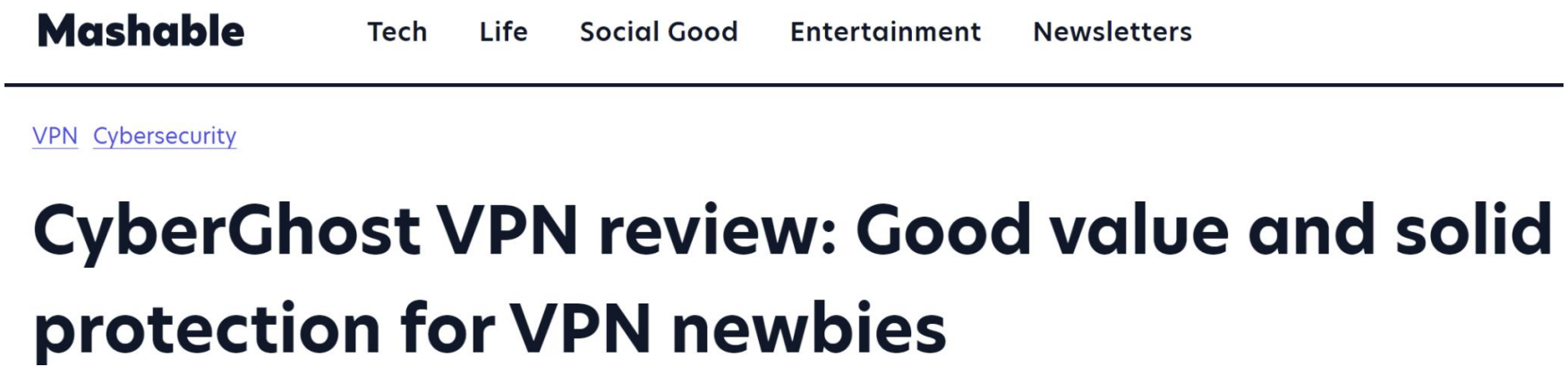
- UI/UX Designers



How do the skills we've learned translate to how CS is used in the technology industry?

Like we said, there are *many different* jobs in CS.

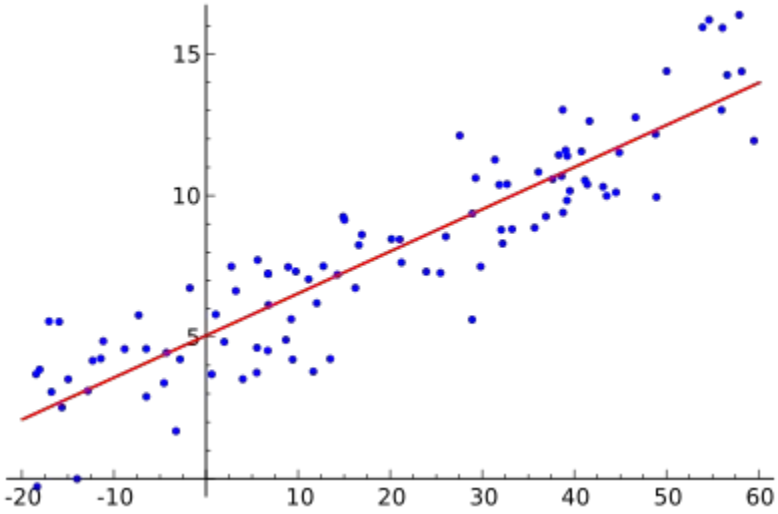
- Technical writers



How do the skills we've learned translate to how CS is used in the technology industry?

Like we said, there are *many different* jobs in CS.

- Data Scientists



How can I translate my CS106B skills into Python?

- While some of the ADTs we learned in C++ also exist in Python (dicts as maps, lists as vectors, sets as sets), many would require you to implement them yourself (queues, stacks, etc.)!

- Python doesn't enable the low-level memory management and system access that C++ does – **there are no pointers**, and you don't get the choice between allocating memory on the stack vs. the heap.
  - As a result, creating linked data structures like trees or linked lists would require creating an entirely new classes for the Nodes class and the linked data structure itself (turning it into more of an ADT in Python).

- Algorithms like searching/sorting and recursive problem-solving translate well!

CS106B has been taught at Stanford for more than 30 years.

Today's lecture blends materials created by Keith Schwartz, Chris Piech, Kylie Jue, and Nick Bowman. Many of the course assignments were created by Julie Zelenski and Chris Gregg.

Gates Computer Science Building, Stanford

Trip Master
Head TA

Jin-Hee

Grant

Jose

Ryan

Ayelet

Section Leaders

Where are you now?

- You now have a wide array of tools you can use to solve a variety of problems.

Where are you now?

- You now have a wide array of tools you can use to solve a variety of problems.

- You have the skills to compare and contrast those solutions.

Where are you now?

- You now have a wide array of tools you can use to solve a variety of problems.
- You have the skills to compare and contrast those solutions.
- You have expressive mental models for teasing apart those problems. (abstractions!)

Closing thoughts

- How is the technology we use made and who makes it?

Closing thoughts

- How is the technology we use made and who makes it?
- Who benefits from the technology?

Closing thoughts

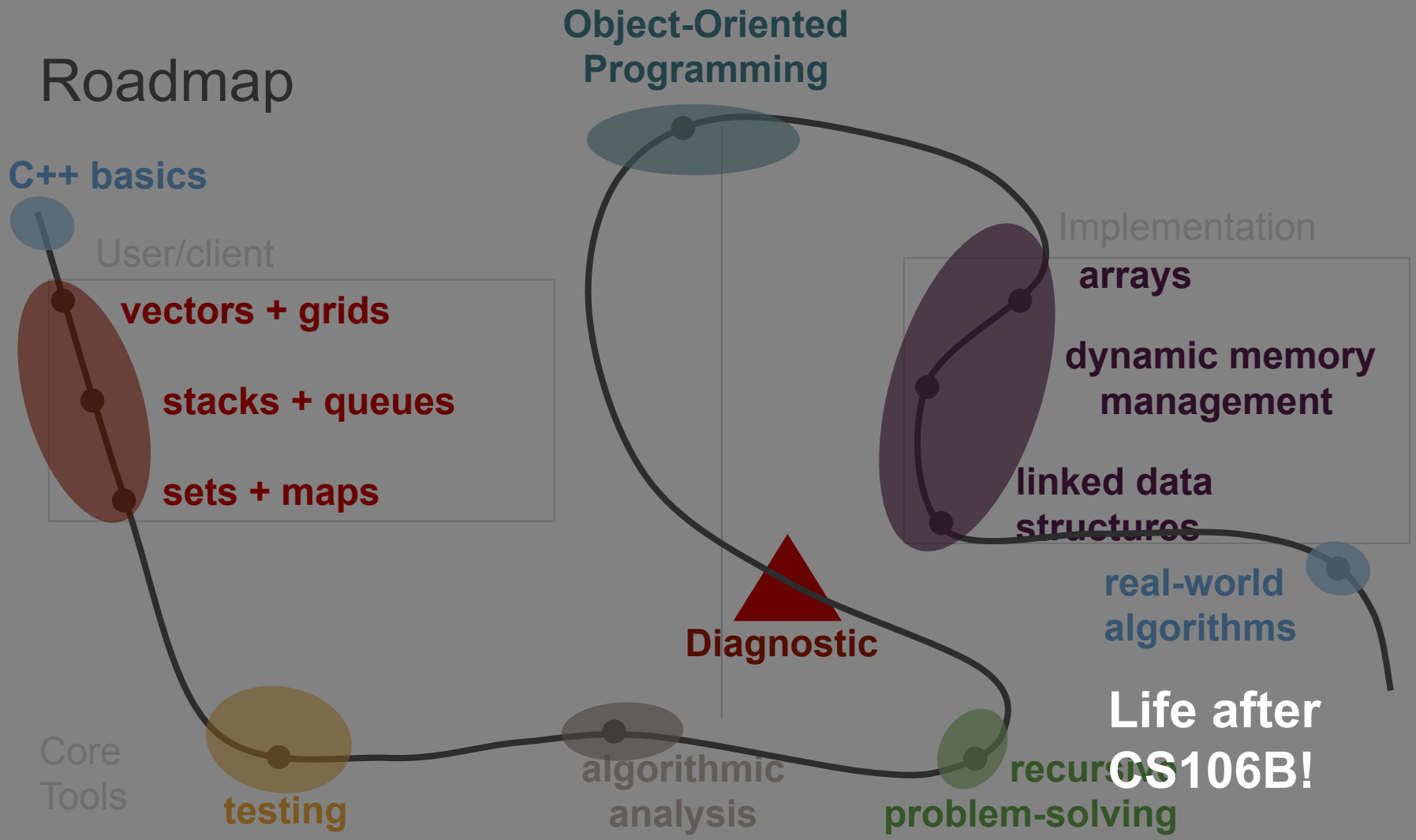
- How is the technology we use made and who makes it?
- Who benefits from the technology?
- Who might not benefit from the technology?

Closing thoughts

- How is the technology we use made and who makes it?
- Who benefits from the technology?
- Who might not benefit from the technology?
- What problems will ***you*** choose to solve with technology?

What’s next?

Roadmap


Object-Oriented Programming
C++ basics
User/client
vectors + grids
stacks + queues
sets + maps
Implementation
arrays
dynamic memory management
linked data structures
real-world algorithms
Diagnostic
Core Tools
testing
algorithmic analysis
recursive problem-solving
Life after CS106B!

Thank you!!!