

# Announcements

---

Your job from now until the final: Study a little each day.

- Final exam is comprehensive.
- Struggling students: DO EVERY C LEVEL PROBLEM. All of them.
- Work with other people!
  - **Argue and discuss!**



# CS61B

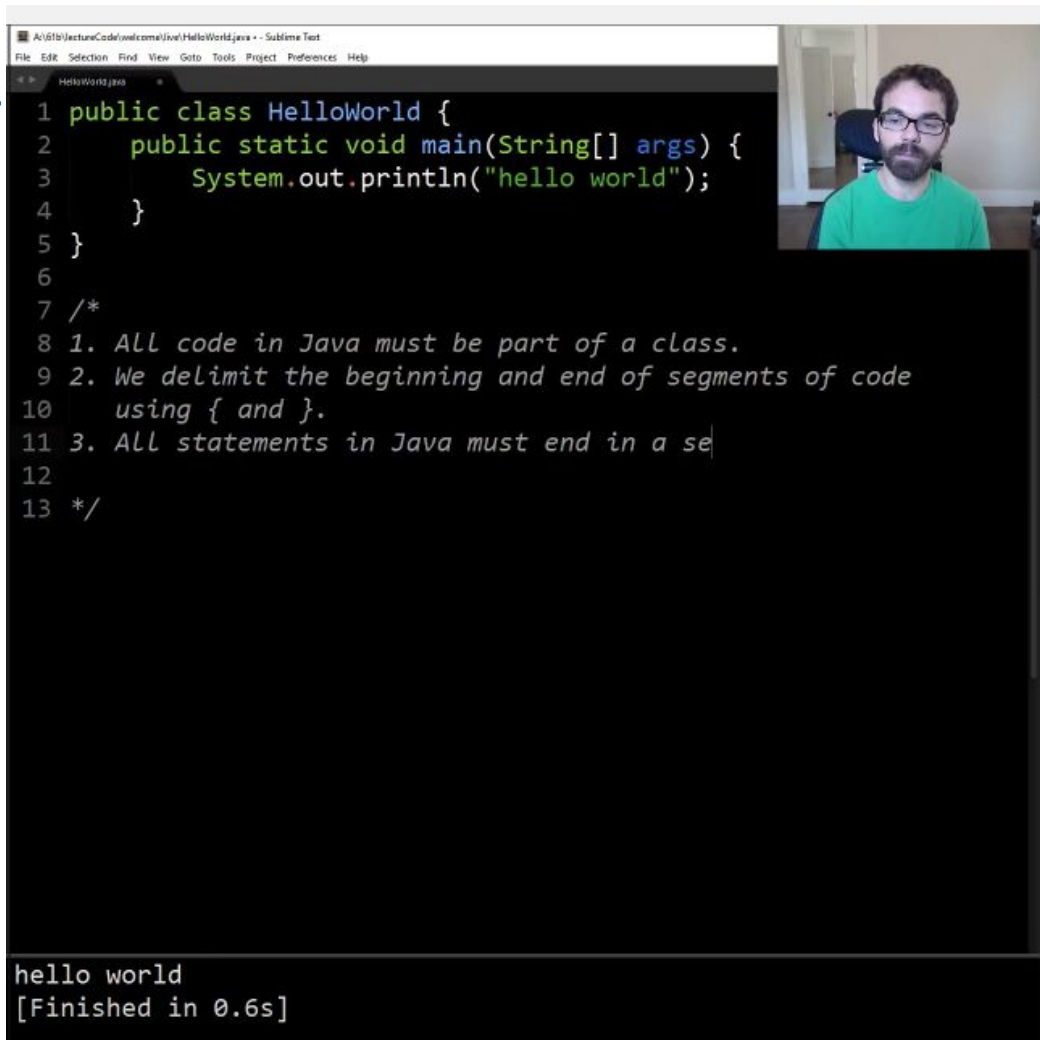
---

## Lecture 40: Wrapping Things Up

- What We've Done: 61B in 12 minutes or less.
- Moving Forwards.

# Where We Started

Just 14 weeks ago:



The screenshot shows a Sublime Text editor window with a file named 'HelloWorld.java'. The code is as follows:

```
1 public class HelloWorld {
2     public static void main(String[] args) {
3         System.out.println("hello world");
4     }
5 }
6
7 /*
8  1. ALL code in Java must be part of a class.
9  2. We delimit the beginning and end of segments of code
10     using { and }.
11  3. ALL statements in Java must end in a se
12
13 */
```



Below the code editor, the output of the program is displayed:

```
hello world
[Finished in 0.6s]
```

In the top right corner of the editor window, there is a small video feed of a man with a beard and glasses, wearing a green shirt, sitting in front of a computer.

# What We've Learned about Programming Languages

---

- Object based programming: Organize around objects.
  - Object oriented programming:
    - Interface Inheritance. 
    - Implementation inheritance.
  - Dynamic vs. static typing.
  - Generic Programming, e.g. `ArrayList<Integer>`, etc.
  - The model of memory as boxes containing bits. 
  - Bit representation of positive integers.
  - Java.
  - Some standard programming idioms/patterns:
    - Objects as function containers (e.g. Comparators, [IntUnaryFunctions](#)).
    - Default method specification in interfaces ([Link](#)).
    - Iterators.
- Example: Programmer only needs to know List API, doesn't have to know that ArrayList secretly does array resizing.
- Example: Array is a sequence of boxes. An array variable is a box containing address of sequences of boxes.

# Important Data Structures in Java

---

Important data structure interfaces:

- `java.util.Collection` (and its subtypes).
  - With a special emphasis on `Map` (and its subtypes).
- Our own Collections (e.g. `Map61B`, `Deque`): Didn't actually extend `Collection`.

Concrete implementations of these abstract implementations:

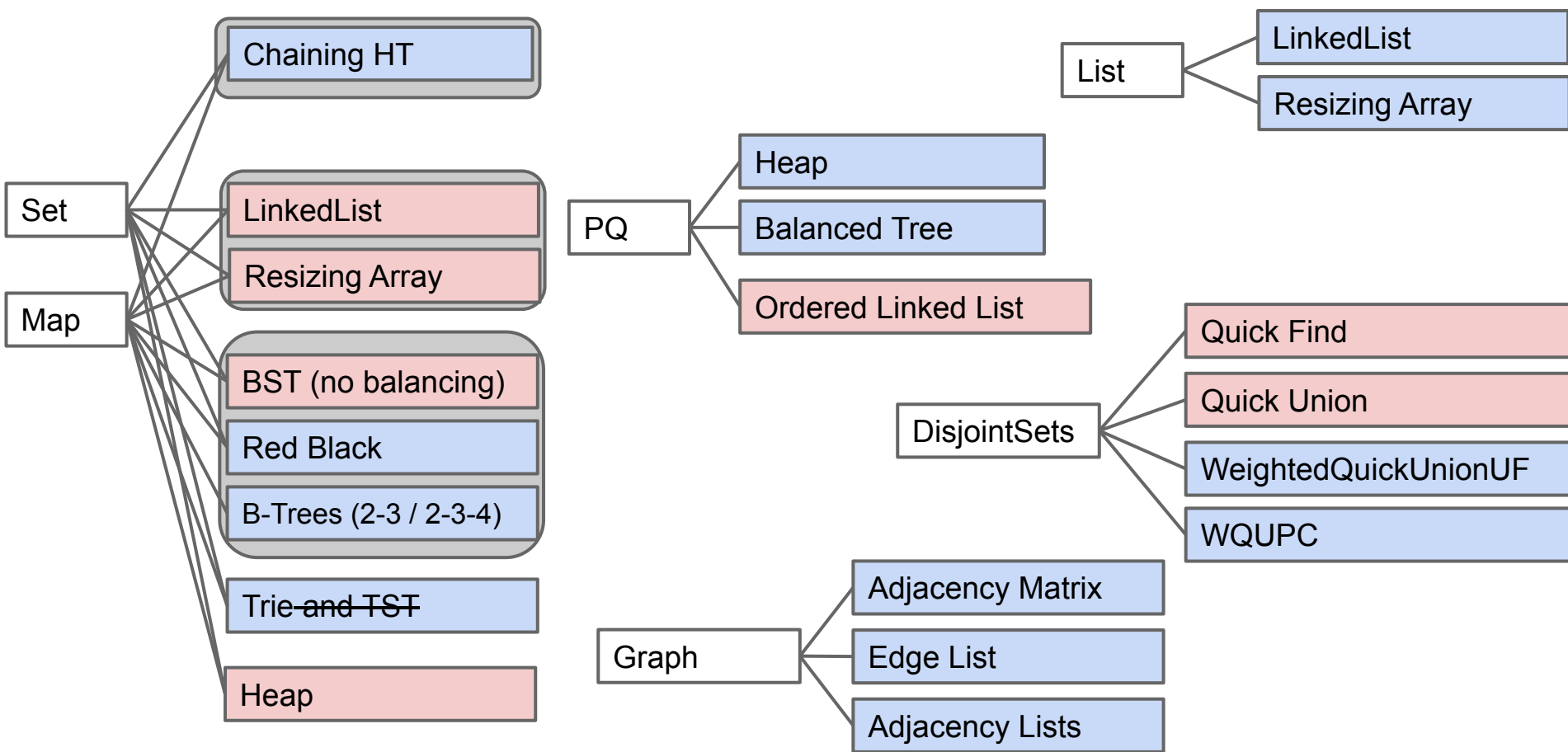
- Examples: `ArrayDeque` implements `Deque`.

# Mathematical Analysis of Data Structure/Algorithm Performance

---

- Asymptotic analysis.
- $O(\cdot)$ ,  $\Omega(\cdot)$ ,  $\Theta(\cdot)$ , and tilde notation.
- Worst case vs. average case vs. best case.
  - Exemplar of usefulness of average case: Quicksort
- Determining the runtime of code through inspection (often requires deep thought).
- Multiplicative resizing is much better than additive resizing (see midterm 2).
  - Multiplicative resizing results in  $\Theta(1)$  add runtime for ArrayLists.
  - Additive resizing results in  $\Theta(N)$  add runtime for ArrayLists.

Some Examples of Implementations for ADTs



# Arrays vs. Linked Data Structures

---

## Array-Based Data Structures:

- ArrayLists and ArrayDeque
- HashSets, HashMaps, MyHashMap: Arrays of 'buckets'
- ArrayHeap (tree represented as an array)

## Linked Data Structures

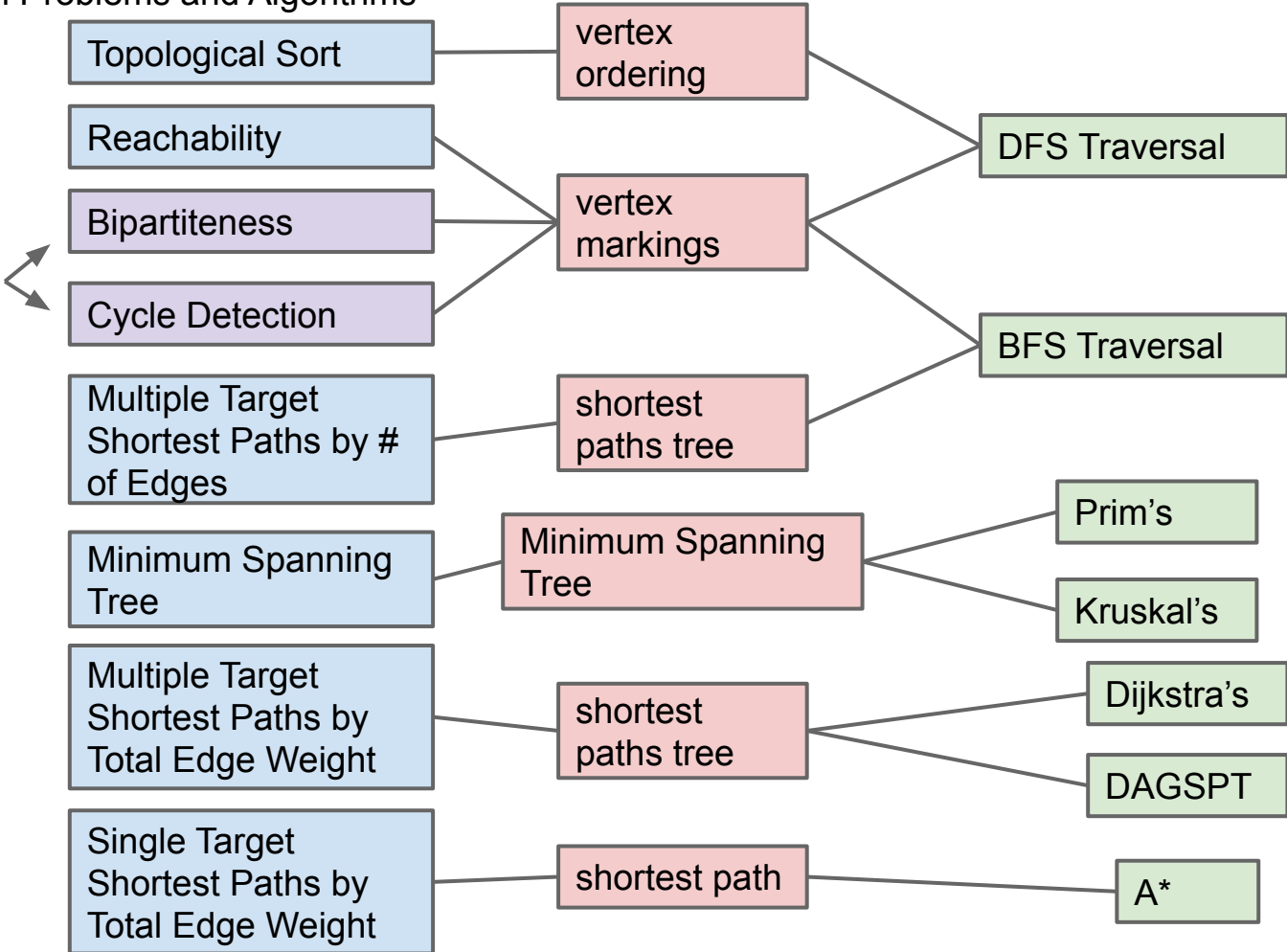
- Linked Lists
  - LinkedList, IntList, LinkedListDeque, SLList, DLList
- Trees: Hierarchical generalization of a linked list. Aim for bushiness.
  - TreeSet, TreeMap, BSTMap, Tries (trie links often stored as arrays)
- Graphs: Generalization of a tree (including many algorithms).

Tradeoffs of arrays vs. linked data structures.

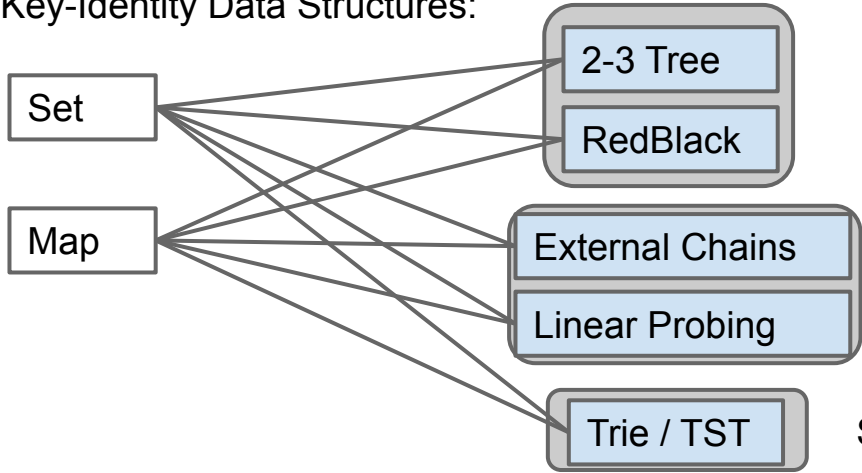


Tractable Graph Problems and Algorithms

Discussed in  
section



Search-By-Key-Identity Data Structures:

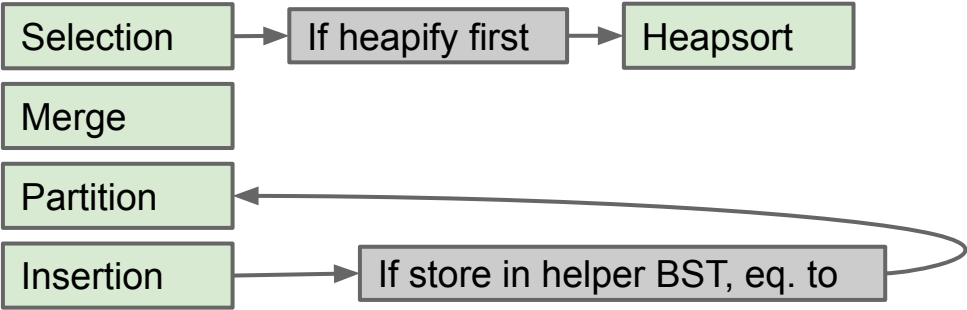


Searches using compareTo()  
Analogous to **Comparison-Based**

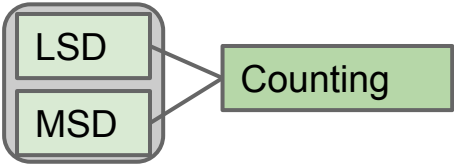
Searches using hashCode() and equals()  
Roughly Analogous to **Integer Sorting**

Searches by digit. Analogous to **Radix Sorting**

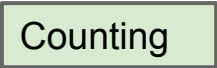
**Comparison Based** Sorting Algorithms:  $\Omega(N \log N)$  worst case



**Radix Sorting Algorithms:**  $\Theta(NL)$  worst case:  
(require a sorting subroutine)



**Small-Alphabet (Integer) Sorting Algorithms:**  $\Theta(N)$  worst case



# Fun/Weird Topics (This Week)

---

## Compression:

- Huffman Coding, and selection of data structures for Huffman Coding.
- Other approaches: LZW and Run Length Encoding (extra slides).

## Compression, Complexity, and P=NP:

- Optimal compression is impossible.
- Bounded space/time compression is possible if  $P=NP$ .
  - Allows you to find arbitrarily short programs that produce a given output in a given runtime.
  - Compression of a stream of bits might provide a useful model of that stream of bits (e.g. hugPlant.bmp -> hugPlant.java).
- If  $P = NP$ , we could also solve all kinds of other interesting problems.

# The Practice of Programming

---

- Java syntax and idioms.
- JUnit testing (and its more extreme form: Test-driven development).
- Mining the web for code.
- Debugging:
  - Identify the simplest case affected by the bug.
  - Hunt it down, giving it no place to hide.
  - With the right methodology, can find bugs even when finding bug through manual code inspection is impossible.
- Tactical vs. Strategic Programming.
  - Avoid information leakage. Build deep modules. Minimize dependencies. Minimize obscurity.
- Real tools: IntelliJ, git, command line
- Data structure selection (and API Design)
  - Drive the performance and implementation of your entire program.

# The Future

# Your Life

---

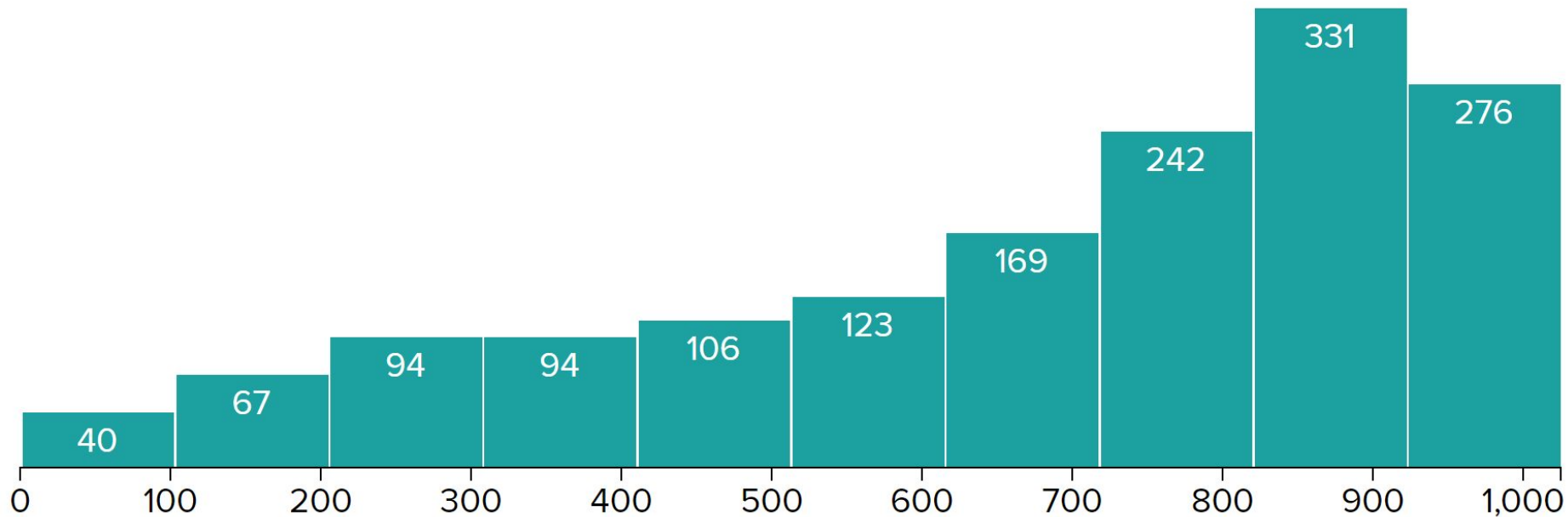
Two interesting questions:

- What am I capable of achieving?
- What should I do with my life?

[illegible]

# Midterm 1

---



MINIMUM

**0.0**

MEDIAN

**754.5**

MAXIMUM

**1025.0**

MEAN

**677.44**

STD DEV

**261.63**



# Poll

---

To what degree do you believe the following statement: Nearly anybody enrolled at Berkeley could achieve an A in CS61B.

- Assume they can spend the entire summer preparing beforehand, hire a tutor during the semester, etc.
- a. Strongly disagree
- b. Disagree
- c. Neutral
- d. Agree
- e. Strongly agree

# Growth vs. Fixed Mindset

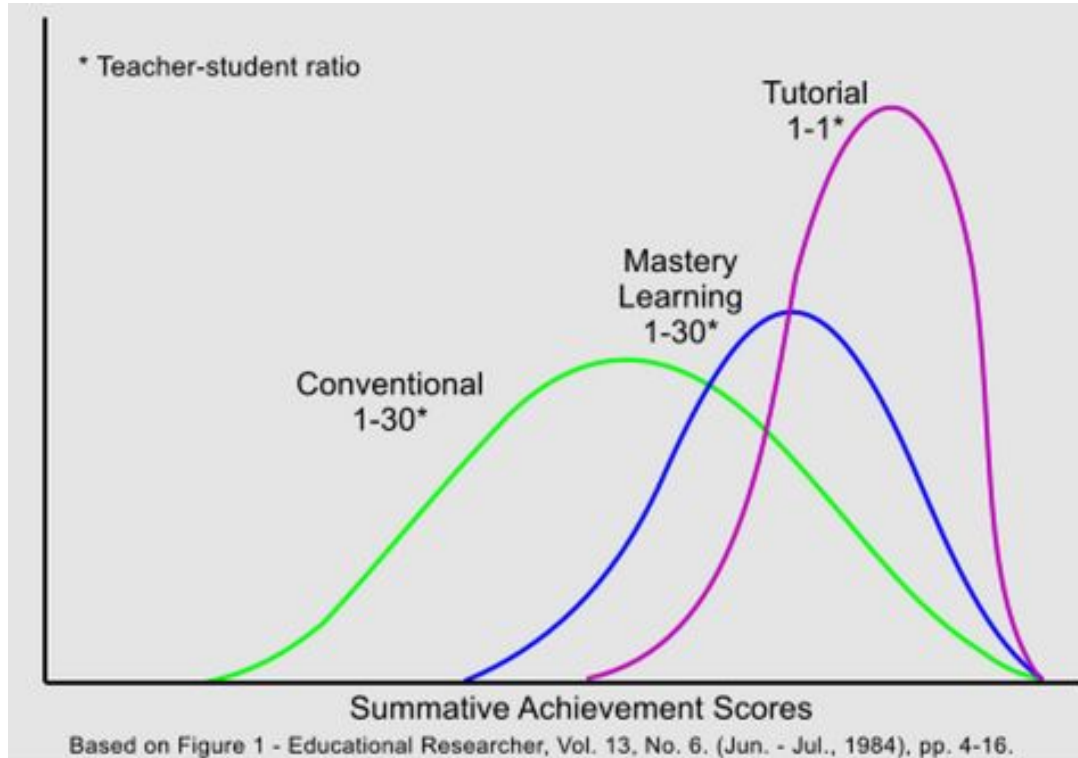
---

Students can be thought of as having either a “growth” mindset or a “fixed” mindset (based on research by Carol Dweck).

- “In a **fixed mindset** students believe their basic abilities, their intelligence, their talents, are just fixed traits. They have a certain amount and that's that, and then their goal becomes to look smart all the time and never look dumb.”
  - Perhaps most damningly, having to put in effort is a sign of weakness!
- “In a **growth mindset** students understand that their talents and abilities can be developed through effort, good teaching and persistence. They don't necessarily think everyone's the same or anyone can be Einstein, but they believe everyone can get smarter if they work at it.”

# Bloom's Two Sigma Problem

Bloom's Two Sigma Problem: In one experiment, student randomly picked for 1-on-1 teaching performed similarly to the top 2% of a simultaneous lecture course.



# A Supporting Experiment

---

In Sp16, I gave students the option to fail intentionally.

- On average, students were 2.5 letter grades higher the second time, e.g. someone in the middle of the B range got an A the second time.

Possible interpretation: There exists some sort of preparation for 61B that helps students do much better.

Bottom line: You're capable of achieving more than you might think possible.

# Your Life

---

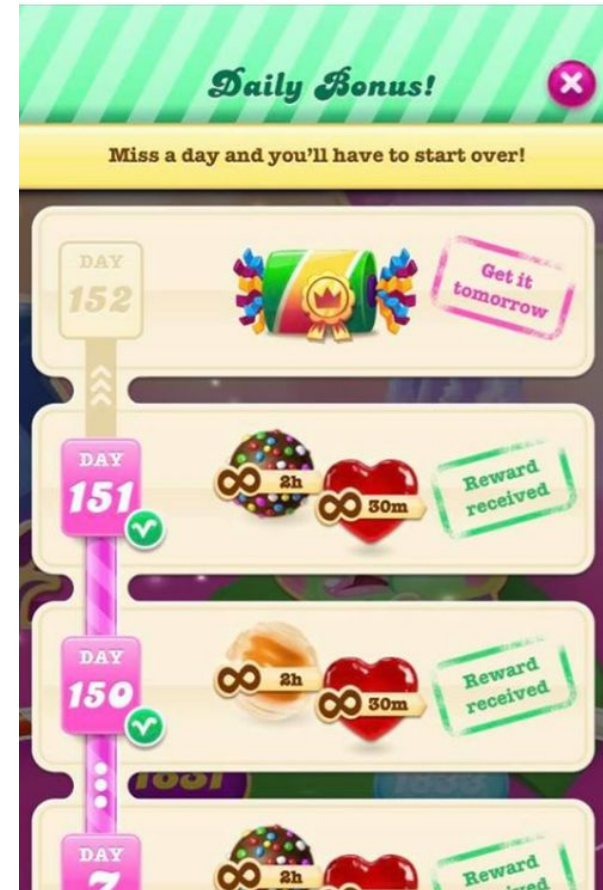
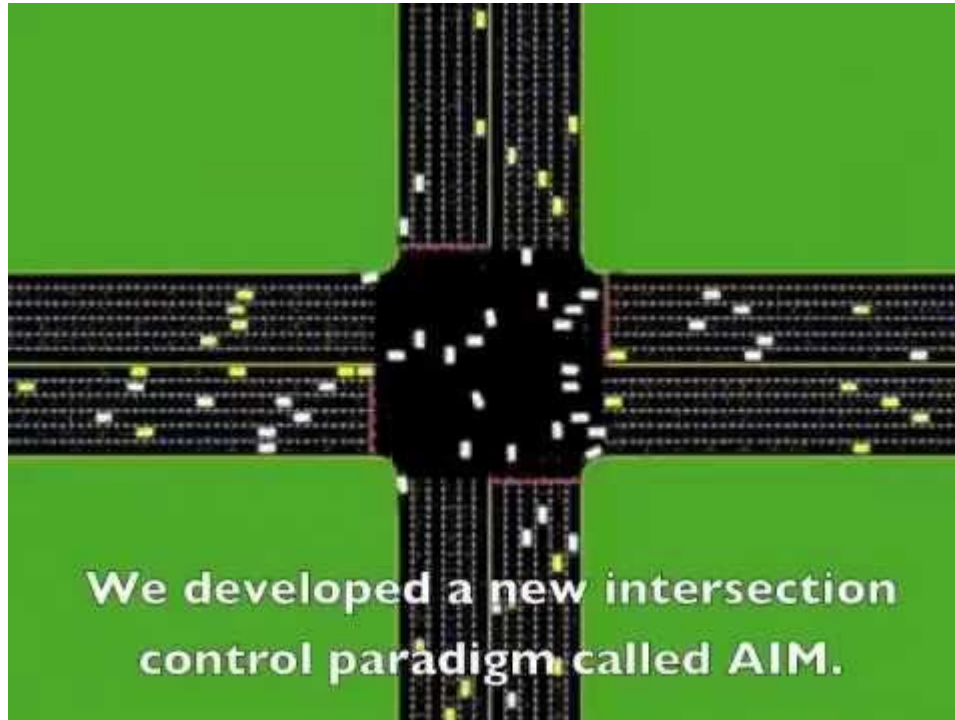
Two interesting questions:

- What am I capable of achieving?
- What should I do with my life?
  - You will be in high demand by everyone.
  - Your skills will enable you to affect the world, possibly profoundly.

Technology will continually to radically reshape the way we live our lives.

# Choices (Yup)

How you use your skills is up to you.



# Using Your Powers for Good

---

My request: Use your superpowers in a way that is a net positive for the lives of your fellow humans.

- I'm not saying that you must work for relatively low wages to uplift the downtrodden (though that'd be great!).
- .... but keep in mind that your work will profoundly affect the world.

(Wanna talk about this stuff more? Take CS195!)

# Questions About Anything?

---



# Questions About Anything?

---

# Course Reflections

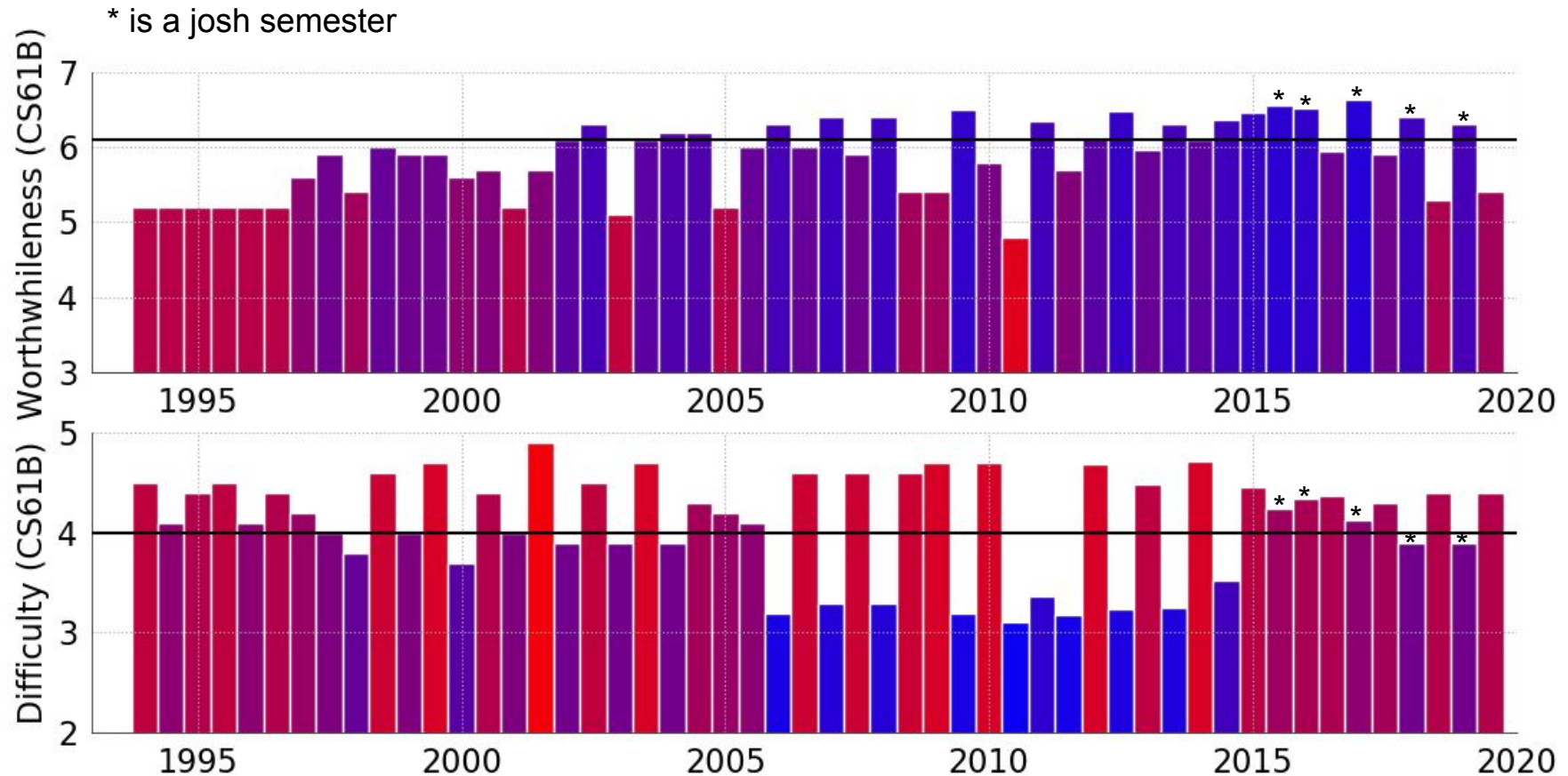
# Reflection on the Course: New for Spring 2021

---

New 61B features since last time I taught 61B (Fall 2020):

- 6 slip days plus flexible policy for people who came to use with life crises.
- 2048 instead of NBody.
- Gitlet instead of Bearmaps.
- More widespread use of Snaps plugin including data analysis (see lec34).
- Two conceptual homeworks instead of 1.
- Asynchronous checkoff for project 3.

# On the Subject of Difficulty: 61B History (black line is Fa18 61A)



# Fall 2022: Things For Next Time

---

Major possibilities:

- Replace Gitlet with something that has significant data structures content but still features a significant design component (e.g. open-ended BearMaps, Sp15 Ngordnet project).
  - Could also maybe revise Gitlet, but I think it's too big.
- More proactively help students find community / study groups.
- Make sharing project 3 even easier online.
- Midterm 1 that allows students to execute code?
- Weekly theory homework that synchronizes with discussion?
- Make project 3 multiplayer?
- Lectures, discussions, and labs in person.

Let me know what else you might want to see on post-final exam survey.

# Any Questions About Course Structure?

---

## Closing Thoughts

---

It is a terrifying and awesome responsibility to decide how you should spend hundreds of your precious hours here at Berkeley.

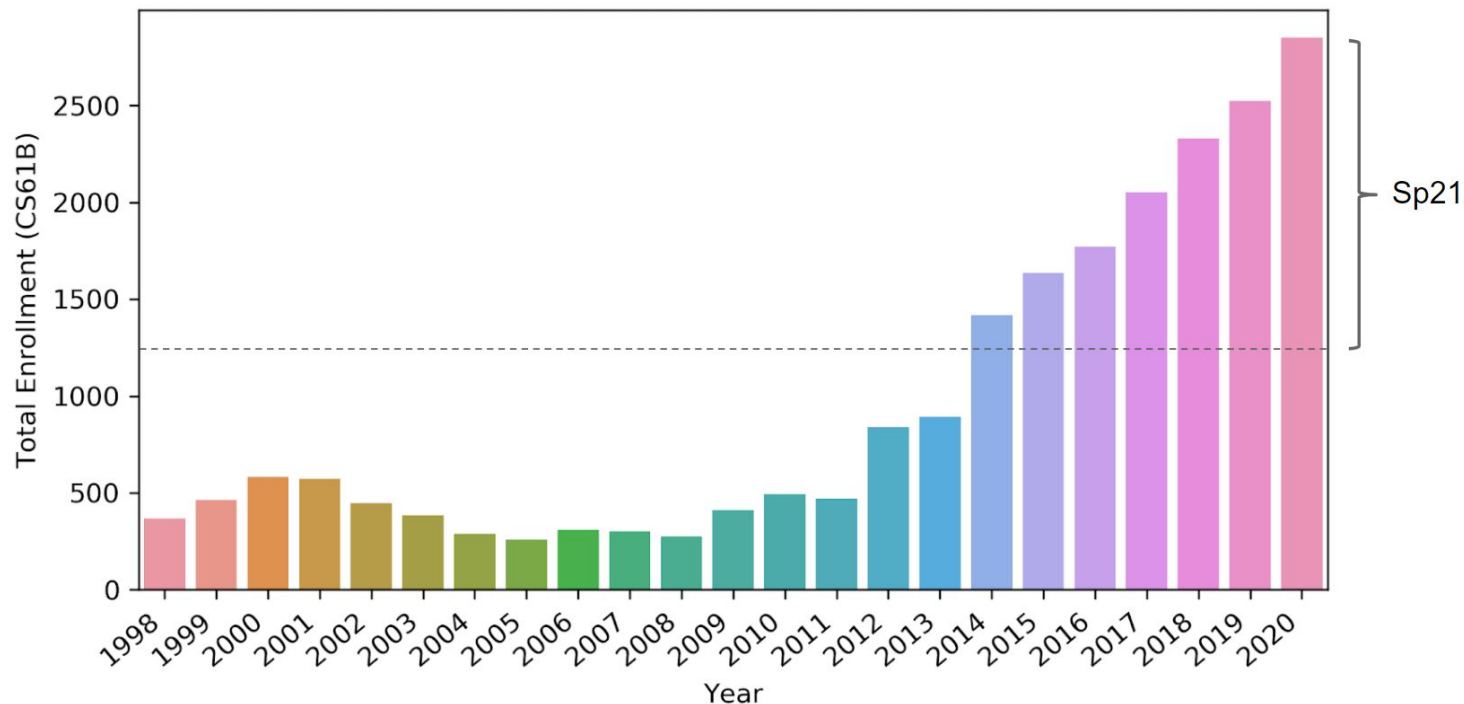
I do this job because I want you to live more fulfilling lives.

I hope I've done a good job. I know it wasn't perfect (and sadly never will be).

... and thanks for all the kind words, feedback, and understanding when things are not as good as they could be.

# 61B Needs You Next Fall

- Academic interning: Learn more, help others, get units, maybe become a GSI.
- Watch out for an announcement on the Spring 2021 61B Ed.





# Special Thanks to the Staff (who keep the wheels on the bus)

---

**Full time GSIs:** Ada Hu, Alex Schedel, Allyson Park, Anjali Kantharuban, Arjun Sahai, Boren Tsai, Claire Ko, **Connor Lafferty**, Eric Tang, Eric Zhu, **Henry Maier**, **Itai Smith**, Linda Deng, Neil Kulkarni, **Omar Khan**, Sohum Hulyalkar, Sumer Kohli

**Part time GSIs:** Ajay Singh, Anton Zabreyko, Aram Kazorian, Cindy Zhang, Crystal Wang, Ethan Mehta, Fatema Yasini, Grace Altree, Hannah Yan, Isha Srinivasan, Jack Wang, Joshua Blanchard, Joshua Yang, Luke Liu, Naama Bareket, Richa Kotni, Robin Qiu, Romain Priour, Sara Reynolds, Sarah Liu, Sarina Sabouri, Sherry Fan, Shreyas Kompalli, SreeVidya Ganga, Tony Kam

**Tutors:** Abhishek Kumar, Ahmed Baqai, Angela Chen, Arushi Somani, Avyakth Challa, David Lee, Jesse Dai, Michael Sparre, Nandini Singh, Nikhil Mandava, Nishant Patwardhan, Saad Jamal, Saikumar Gantla, Sean Kim, Shreyans Sethi, Shriya Nandwani, Smruthi Balajee, Srinidhi Sankar, Todd Yu

**Academic Interns:** Justin Thein, Kyle Yu, De Zheng Zhao, Michael Huang, Rahul Mohankumar, Deepak Ragu, Tymon Thi, Devin Sze, YiTing Yan, David Chung, Viansa Schmulbach, Ritik Batra, Yash Gupta, Arda Demirci, Anik Gupta, Nitya Goyal, Paris Zhang, Stephanie Jue, Dexter To, Sean Hayes, Olivia Huang, Genevieve Brooks, Noah Schwartz, Kush Garg, Reza Sajadiany, Michelle Cheung, Maryam Azmandian, Amanda Yao, Andrew Lee, Haroun Khaleel, Catherine Hwu, Stephen Yang, Jerry Sun, Aady Pillai, Emmett Dreyer, Kaci Gu, Richard Lee, Suhrid Saha, Sunay Dagli, Fakhri Widodo, Shefali Goel, Alina Trinh, Amritansh Saraf, Siddhant Sharma, Phoebe Troup-Galligan, Steven Chen, Abrar Rahman, Xinqi Yu, Sum Ying Celeste Wu, Brenda Huang, Zoe Parcells, Nathan Tran, Aditi Bamba, Jasper Emhoff, Amudha Sairam, Ananya Gupta, Samuel Stulman, Carolina Rios-Martinez, Ola Alsaedi, Shivani Ahuja, Tiffany Luu, Isha Arora, Lawrence Gan, Aditya Prasad, Daniel Detchev, Grace Chen, Austin Ho, Nicholas Cheng, Shannon Bonet, Jane Lee, Kaaviya Sasikumar, Ankita Janakiraman, Kevin Jin, Kuhu Sharma, Vivian Liu, Xinyu Fu, Sofia Roz, Walker Browning, Efsane Soyer, Christopher Seo, Grace Cen, Maggie Yi, Rohan Khandelwal

That's 17 full time GSIs, 25 part time GSIs, 19 tutors, 83 academic interns, 1 me, 1437 you.