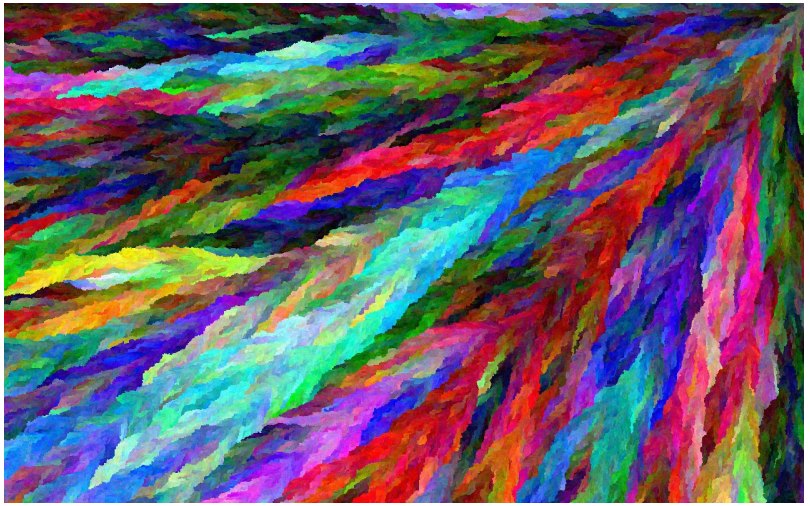
A Note for those Stumbling on these Slides (Hi!)

- These lecture slides are not intended as written reference materials.
  - Just reading them probably won't be very educational.

- Best used in combination with webcasts and source code references.
  - See (Video) and (Code) links under each lecture: http://datastructur.es

Announcement

If you want to chat after lecture, meet me in the lobby, don't climb on the stage, they will be mad. Also the Kronos Quartet's stuff could get destroyed and they have lawyers and deep seething anger for youth (I made that last part up).



CS61B: 2019

Lecture 1:

- Introduction
- Course Logistics
- Hello World

61B Overview

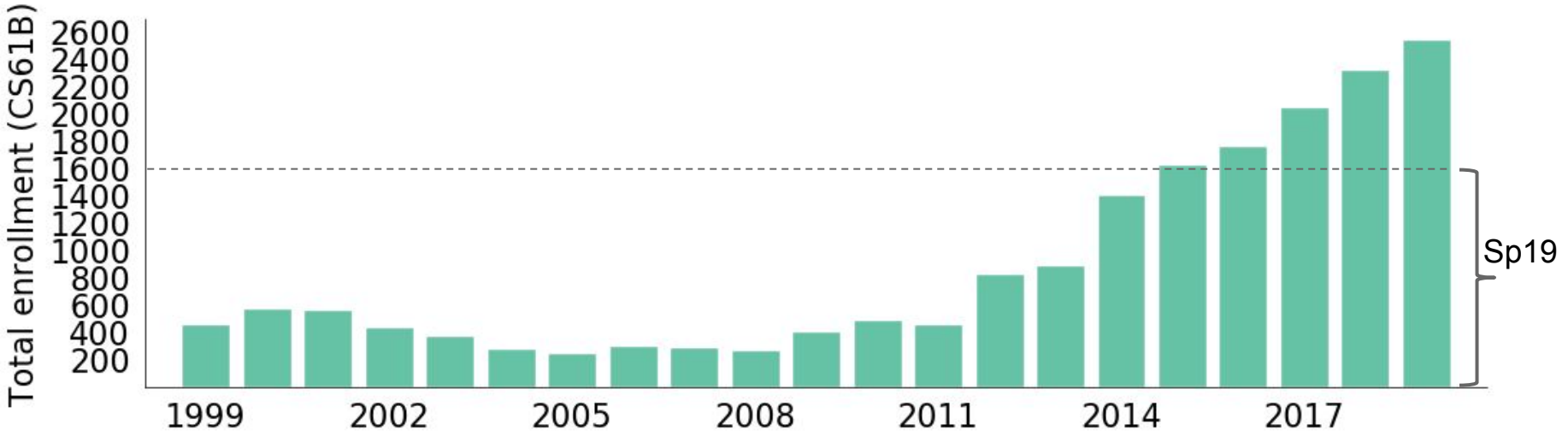
What is 61B about?

- Writing code that runs efficiently.
  - Good algorithms.
  - Good data structures.
- Writing code efficiently.
  - Designing, building, testing, and debugging large programs.
  - Use of programming tools.
    - git, IntelliJ, JUnit, and various command line tools.
  - Java (not the focus of the course!)

Assumes solid foundation in programming fundamentals, including:

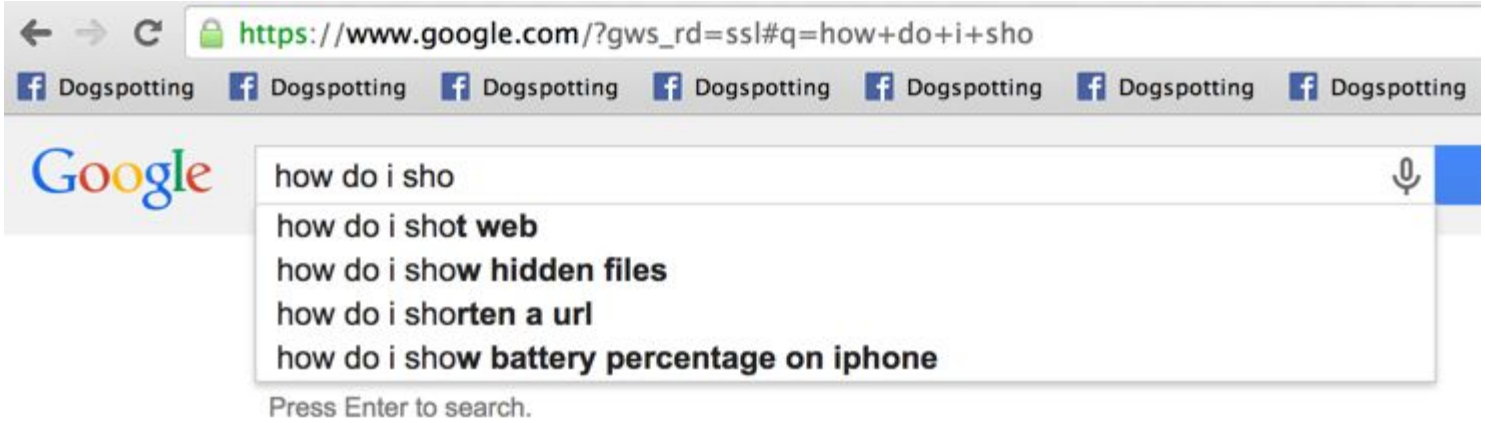
- Object oriented programming, recursion, lists, and trees.

Why Study Algorithms or Data Structures?



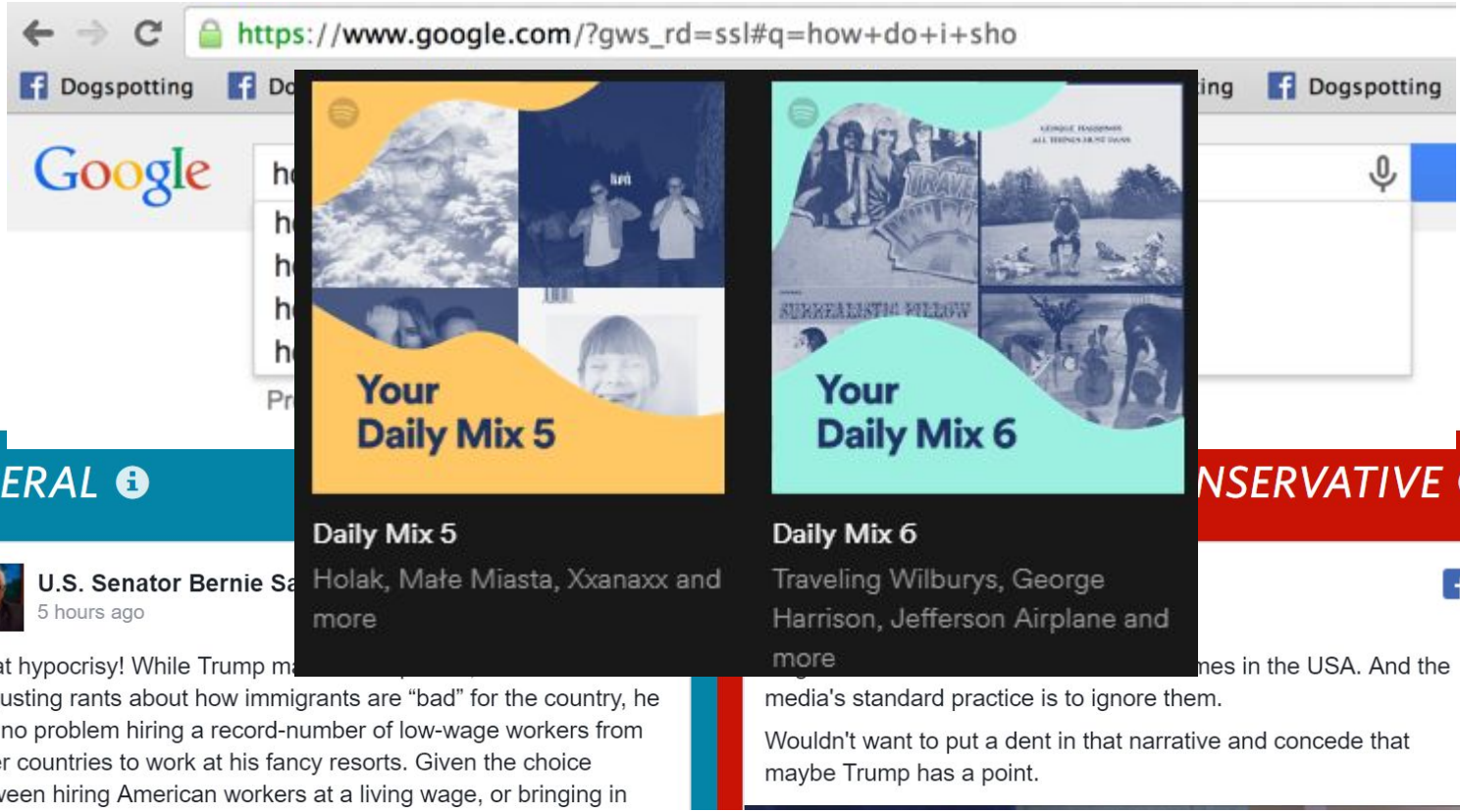
Why Study Algorithms or Data Structures?

Daily life is supported by them.



Why Study Algorithms or Data Structures?

Daily life is supported by them.



Why Study Algorithms or Data Structures?

Major driver of current progress (?) of our civilization.

Ancient Infant's DNA Reveals New Clues to How the Americas Were Peopled

Her 11,500-year-old remains suggest that all Native Americans can trace their ancestry to the same founding population.

AlphaGo

Why Study Algorithms or Data Structures?

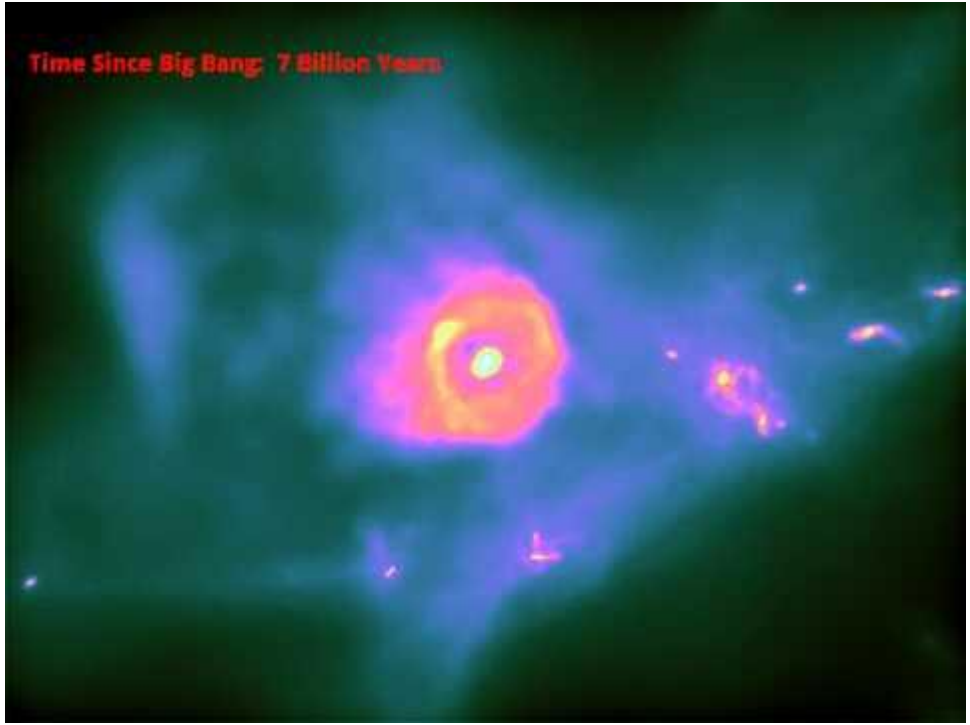
To become a better programmer.

> *"The difference between a bad programmer and a good one is whether [the programmer] considers code or data structures more important. Bad programmers worry about the code. Good programmers worry about data structures and their relationships."* - *Linus Torvalds (Creator of Linux)*

Being an efficient programmer means using the right data structures and algorithms for the job.

Why Study Algorithms or Data Structures?

To understand the universe. Science is increasingly about simulation and complex data analysis rather than simple observations and clean equations:



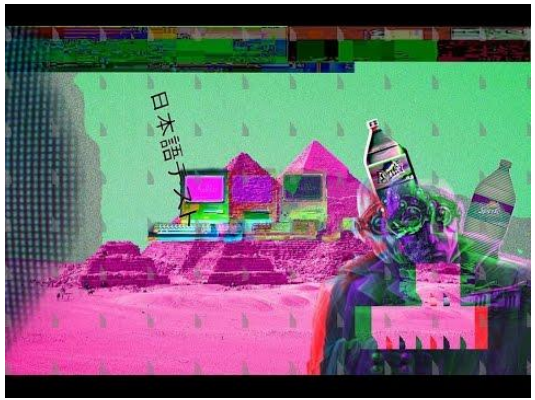
Why Study Algorithms or Data Structures?

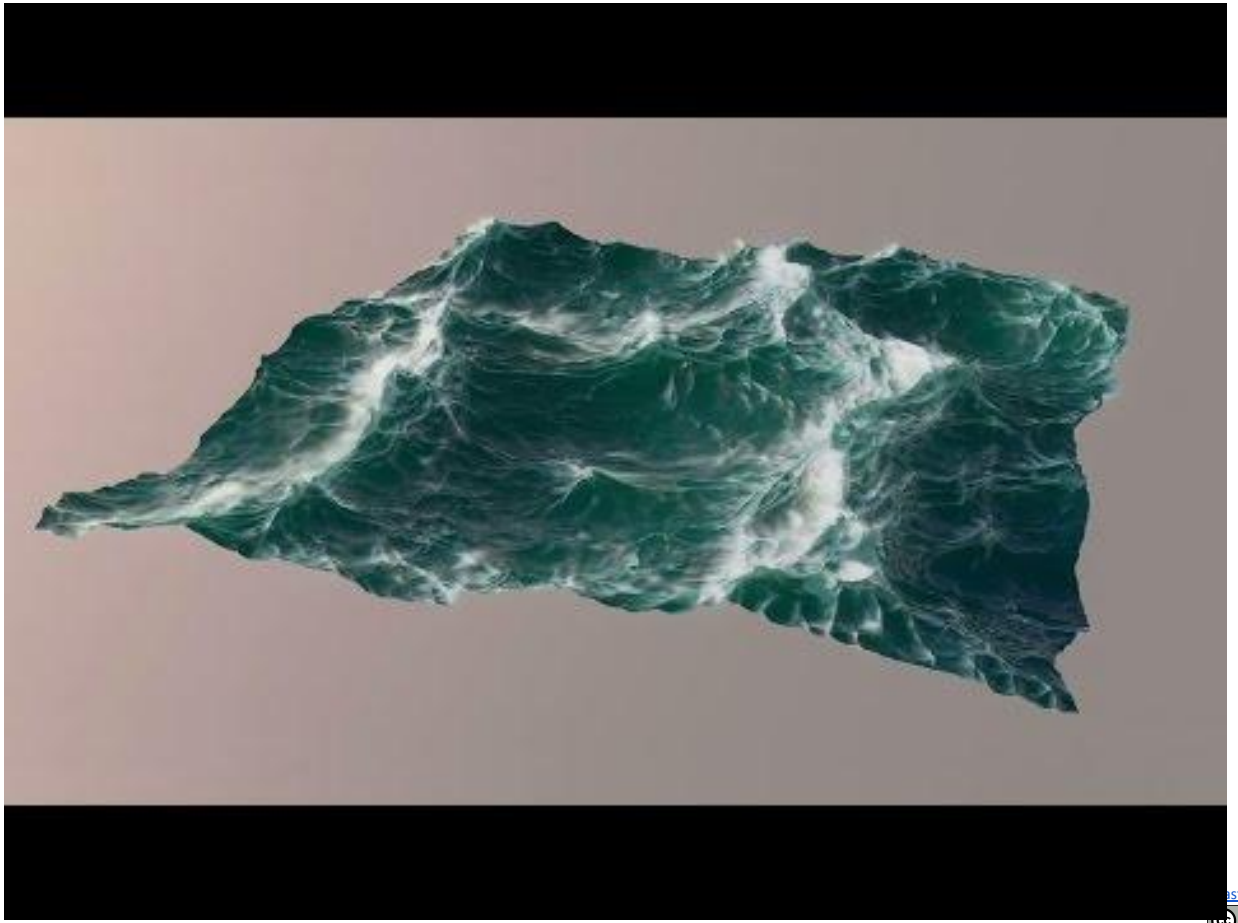
to

create

beautiful

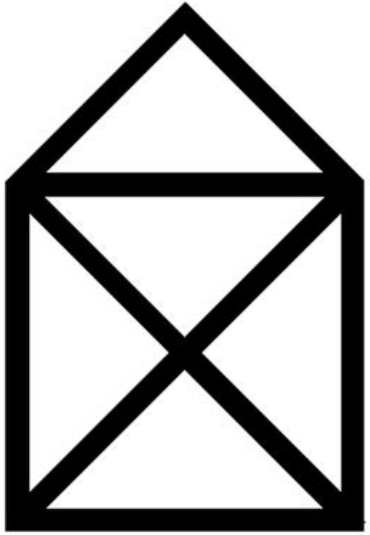
things



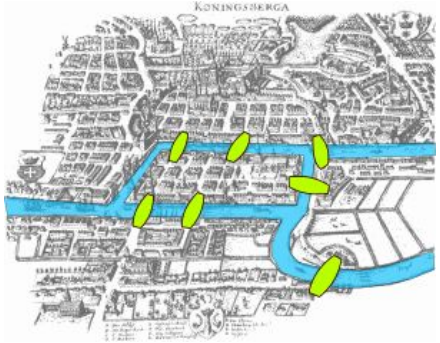


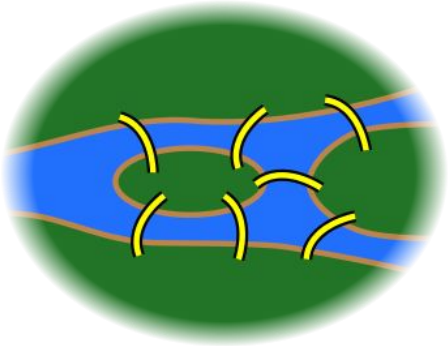
Why Study Algorithms or Data Structures?

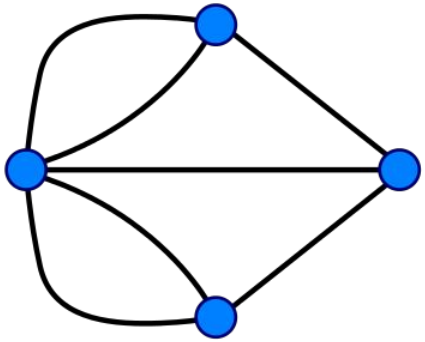
As an end unto itself.



Possible to draw without picking up pencil or going back over any lines.







Impossible.

Question for You

What do you hope / expect to learn from this class? Why are you taking it?

- Google/Tesla interview questions.
- Getting better at programming.
- What is “computer science”? Comptology.
- I have to - to declare my major: Data Science.
- Understand the memes on the meme page.

Who Are You?

Who Are We?

Instructor: Josh Hug (me) hug@cs.berkeley.edu 779 Soda

GSIs:

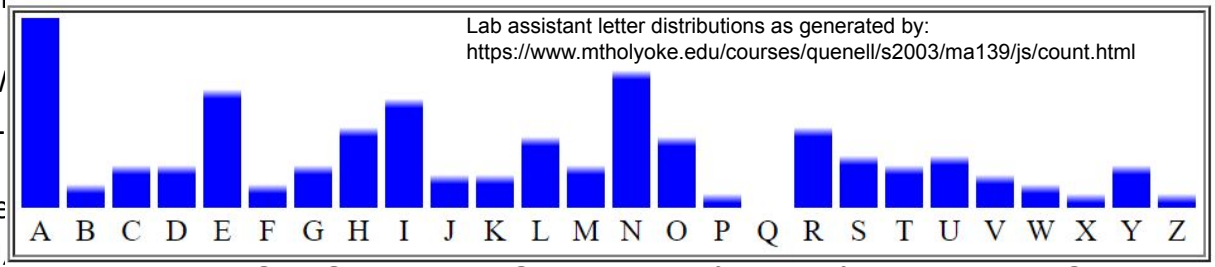
- Part time: Angela Kwon, Annie Huang, Brandon Fong, Carlo Cruz-Albrecht, Catherine Cang, Charles Hager, Ellin Zhao, Erica Kong, Itai Smith, Jennifer Liu, Jeremy Dong, Jihan Yin, Julian Kung, Julianna White, Kartik Kapur, Kelly Lin, Kevin Chang, Lauren Hong, Lucas Pan, Michael Lum, Michael Meng, Neil Kulkarni, Nicholas Titterton, Omar Khan, Riley Woo, Shreya Sahoo, Shubham Gupta, Tina Zhao, Vibha Seshadri
- Full time: Albert Hu, Brandon Lee, Catherine Han, **Christine Zhou**, Danny Chu, **Eli Lipsitz**, Gigi Lu, **Jackson Leisure**, Jenny Huang, Joe Deatrick, Josh Zeitsoff, **Matt Owen**, Matthew Sit, **Michelle Hwang**, **Mudit Gupta**, Rahul Nangia, Sam Zhou, Sandra Hui, Sandy Zhang, Ting Ding

Who Are We (continued)?

Tutors: Aakash Shukla, Ajuney Hora, Akshit Annadi, Alan Ton, Alex Kassil, Anirvin Sikha, Anson Tsai, Arjun Sahai, Darren Huang, Emily Zhong, George Zhou, Hannah Yan, Joshua Yang, Mahesh Murag, Megan Kawakami, Mounica Putrevu, Nathan Miller, Noah Kuo, Oscar Ortega, Ramsey Mardini, Rene Lee, Ryan Nuqui, Ryan Tran, Ryan Tseng, Shriya Vohra, Shubha Jagannatha, Uma Krishnan, Yiling Kao

Who Are We (continued)?

Academic interns: Kevin Ko, Jeffrey Hong, Andrew Wang, Amol Pant, Ju Sheen Kim, Amanuel Yitbarek, Dylan Tran, Richard Huang, Apoorva Prakash, Ethan Yeh, Ziqi Jin, Ikram Magzoub, Mher Torjyan, Ruhi Pudipeddi, Kelvin Ngo, Edmund Wang, John Siano, Alejandro A. Stefan Zavala, Prom Putthisri, Cliff Liu, Prateek Kher, Vlad Korsunenko, Stefano Colmignoli, Youqi Chen, Adya Putra Indera, Jie Chen, Ryan Lee, Emily Wang, John Markham, Alex Schedel, Arjun R Bharat, Harim Lee, Harry Song, Jun Jeon, Frederick Fan, Sean Lam, Jacqueline Chu, Connor Lafferty, Howard Ho, Sanchit Kapoor, Kunal Agarwal, Alice Lyu, Michael Chang, Haiwei Zhen, Monica Tang, Cindy Liu, Jiekun Liang, Yuanzhe Ying (Cody), Kevin Li, Ahir Datta, Salman Ahmad, Shivani Prabala, Caleb Kahookele, Amy Wang, Edward Sun, Emanuel Rew, Tarsus Lam, Ashwin Shanker, Soumya Avva, Tammy Truong, Eric Chen, Vatsal Bajaj, Jenny Song , Daniel Fan, Jason Kim, Benson Yuan, Jenna Dorse, Manula Dombagahawatta, Cheenar Gupte, Richard Padilla, David Ruan, Yash Vanvari, Winnie Zhang, Alfredo Andere V., Henry Leou, Asad Abbasi, Tianhao Chen, Yibin Li, Guoxin Yu, Karan Shah, Fei Cong, Brian Fu, Vikram Chandran, Juan Ramon Cai, Eric Ortiz, Eric Gan, Aleksander Petuskey, Yuan Xie, Devesh Rai, Felix Li, Alan Zhou, Theo Luan, Alex Cui, Kathleen Chang, Amitav Baruah, Kevin [illegible] Jonathan Pan, Yafei Liang, Carolyn W[illegible]ua Samuel, Jerry Gong, Timothy L[illegible]o, Calvin Chen, Shein Lin Phyo, Jaslyn [illegible]eld, Jacky Chow, Charlie Tian, Re[illegible]ia Lee, Alexia Colmenero, Jimmy Xu[illegible]mad Maarouf, Soung Bae Kim, Calvin Kwon, Zijin Cui, Lam Pham, Shahzar Rizvi, Ann Yoo Abbott, Vidhi Chander, Yiming Yang, Dylan Buer, Annie Wong, Shahroze Ranjha, Katniss Lee, Dave Kwon, William Tong, Eugene Lee, Beatriz Israde



Learning Philosophy

The Manner in Which Learning Occurs (TMWLO)

TMWLO: A Small Minority

Lectures

- Introduction to new material.

Reading

- More thorough introduction.

TMWLO: The Vast Majority

Discussion Section and Study Guides:

- Practice with concepts. Often more theoretical.

Labs, Homework, and Your Own Experimentation

- Practice with tools, programming techniques, Java syntax, and algorithms and data structures.

Projects

- Similar to labs and HW, but larger and include a design component.

Course Logistics

Places to Get Information

Official Course Resources

- Course website: http://datastructur.es
- Lectures (or webcasts). Discussion. Lab. Office Hours (starting next week, times and locations TBA).
- Piazza: http://piazza.com/class/61b
- Mini-Textbook: Obscurantism in Java https://joshhug.gitbooks.io/obscurantism-in-java-first-edition
- 5 student Group Tutoring Sections (starting week 3?).

Unofficial: Google, Stack Overflow, other programming courses on the web, various online documentation, etc.

Logistical Details

- For waitlisted folks: If you do project 0, I'll do what I can to get you in by week 4, but no guarantees.
  - Unlike past semesters, there is a chance we might not be able to accommodate everyone! Demand is much higher than expected.
  - Sp17: 1379 students, Sp18: 1420, Sp19: 1646
- Week 1 sections and labs are first-come, first-served. See piazza post here: https://piazza.com/class/jqr7hfmf4v74e?cid=73
- Once attendance settles, doesn't matter which lab/section/lecture you're registered for.
- **Please post administrative issues to Piazza or send an email to cs61b@berkeley.edu**
  - Please don't email me with such issues directly (sorry!).
  - 1650 students * 1 minute/student = 27.5 hours.

61B 3.0 - Course Structure

Phase 1: Programming Intensive Introduction to Java.

- Weeks 1-4.
- One browser-based programming HW (this HW0 is optional).
- Three labs to introduce you to various tools (starting this week).
- Two projects (proj0 and proj1).
- Midterm 2/20 at 8:00 PM.

61B 3.0 - Course Structure

Phase 2: Data Structures.

- Weeks 5-10.
- Incredibly important and foundational material: Expect an CS job interview to lean heavily on this part of the course.
- Two medium sized HWs (HW1, HW2) and one small HW (HW3).
  - Applications and deeper insight into data structures.
- One large project (Proj2A/2B), due ~3/18 and ~3/22.
  - No autograder. You're responsible for your code's correctness.
- Standard labs implement data structures (e.g. hash table).
  - Challenge labs are more like hard puzzles using data structures.
- Midterm 4/5 at 8:00 PM.

61B 3.0 - Course Structure

Phase 3: Algorithms and Software Engineering

- Weeks 10-14
- Labs: Implement a basic algorithm (e.g. quicksort).
  - Challenge labs: Hard data structures / algorithmic puzzles.
- One medium sized HW: Implement an AI that solves problems using A* algorithm.
- Two projects:
  - Proj2C: BearMaps. Uses data structures from Proj2A/2B and algorithm from HW4 to implement a browser-based mapping application.
  - Proj 3: Build Your Own World: An open ended project where you build a 2D world with physics according to your own design.

See calendar at http://datastructur.es for more.

Labs and Discussion

Attendance for lab/discussion is not required, but can earn “pacing points” by attending discussion or lab.

- See course website for details on pacing points.
  - tl;dr is that they can help you get up to a B-, but don’t help beyond that.

Labs:

- Lab always due by Friday at 11:59 PM. Full credit for ‘reasonable effort’.

HWs and Projects

4 homeworks.

- Due dates vary widely, see calendar.

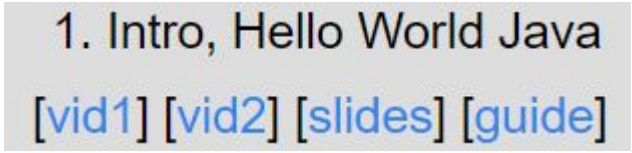
Projects:

- Projects 1 and 2 must be done solo. 0 and 3 can be done as a pair.
- Projects 2 and 3 will be really time consuming and difficult.
- **All code on solo projects must be your own work.**
- Ok to discuss with others and help debug.
- Extra credit opportunities on projects 2 and 3 for finishing early.
- Can earn gold points on projects for going above and beyond.
    - Gold points are special points whose value is worth more if your exam scores are lower. See course website for full details.

Weekly Surveys and Study Guides

Weekly survey due every Sunday.

- Check in on your progress and report attendance.
- Free points, but no late submissions allowed!
- Lowest four dropped.



Study guides for each lecture.

- HWs will not prepare you directly for many types of exam problems.
  - Use these **study guides** instead!
- Provides a brief summary of the lecture.
- Provides (usually) C level, B level, and A level problems for exam studying.
  - A level problems are usually hard enough that I anticipate TAs will have a hard time with them, so be nice!

Exams

- Closed note except you can bring cheat sheets.
- Will be pretty hard (60% medians).
- Showing improvement on final can boost overall exam score.
  - If your final is statistically better, it can replace one of your two midterms. See “supersession” under course website for details.

Exam dates:

- Midterm 1: **February 20th**, 8:00 -10:00 PM.
- Midterm 2: **April 5th**, 8:00 - 10:00 PM.
- Final Exam: **May 15th**, 7:00 - 10:00 PM.
- There will be **no alternate exams** (see exam replacement policy).
  - One exception: Direct conflict with a final from another class.

Course Grade

Breakdown: 3,152 points total. Letter grade will be determined by your total.

- Midterms: 800 points total.
- Final: 800 points.
- Projects: 960 regular points.
- HW: 320 points (80 points each)
- Lab: 192 points (16 points each)
- Weekly Surveys: 80 points (8 points each)

Plus extra credit for filling out pre-, mid-, and post- semester course surveys, also for turning in projects 2 and 3 early.

Plus gold points for going above and beyond on projects.

Grades are not curved, i.e. they are not based on your relative performance. In past semesters, my grade bin cutoffs have not moved much at all.

- See http://sp19.datastructur.es/about.html for full details including grading bin cutoffs.

Lateness and Mentor GSIs

No late work will be accepted in the course. No hws/labs/projects are dropped.

- You should treat the deadlines seriously!
- Use automated extension system to request extensions **in advance**.
  - It will give a small number of extensions (details will not be divulged).
  - Once you exceed these, you'll need to talk to your Mentor GSI.
  - "In advance" means any time before deadline.

During week 2, you will be pick a **mentor GSI**.

- If you do not pick one, one will be assigned for you.
- Mentor GSI will keep track of your progress and reach out (or have someone else reach out) if they feel like there are potential issues.
- If automated system rejects extensions, you must ask your mentor GSI.

Course Pacing

We will start off very fast.

- Optional HW0 is out.
  - Intro to Java syntax.
  - Will take 1-4 hours.
  - Work with friends!
  - Recommended that you complete before your lab.
  - Strongly recommended that you complete by lecture Friday.
- Lab1 and Lab1 Setup are both available.
  - Lab1: How to use various tools.
  - Lab1 Setup: How to set up your home computer (maybe do before lab1).
- Project 0 released 1/23. Due Friday Feb 1st (10 days from start of semester).
  - Start by Saturday if possible, especially if you're new to Java.
  - Exercises all the basic Java features.
  - Allowed to work in pairs (more in lecture next time or see proj0 spec).

1. Intro, Hello World Java

[vid1] [vid2] [slides] [guide]

Hello World

(See guide for link to the code I write today)

(Might be a little boring if you know Java already)

Java and Object Orientation

Java is an object oriented language with strict requirements:

- Every Java file must contain a class declaration*.
- **All code** lives inside a class*, even helper functions, global constants, etc.
- To run a Java program, you typically define a main method using `public static void main(String[] args)`

*: This is not completely true, e.g. we can also declare “interfaces” in .java files that may contain code. We’ll cover these later.

Java and Static Typing

Java is statically typed!

- All variables, parameters, and methods must have a declared type.
- That type can never change.
- Expressions also have a type, e.g. "`larger(5, 10) + 3`" has type int.
- The compiler checks that all the types in your program are compatible **before the program ever runs**!
  - e.g. `String x = larger(5, 10) + 3` will fail to compile.
  - This is unlike a language like Python, where type checks are performed DURING execution.

Your Reflections on Static Typing

The Good:

- Easier to follow what the code does!!
- It lets you catch one type of mistake beforehand.
  - Customers never see type erorrs.
  - You the programmer immediately catch one type of error.
- More efficient under the hood.

The Bad:

- Tedious!
  - tHE CODE IS MOST verbose
  - Code is less general and making it general takes lots more code.
-

What's Next

This week:

- HW0: Out now. Will give you a chance to explore Java basics on your own.
- Lab1: How to compile and run code on the lab machines. How to check out homework starter files and submit them. **If possible, do HW0 before lab!**
- Lab1 Setup (optional): How to compile and run code on your own machine.
- Project 0 (just released). Only two types of partnerships allowed:
  - Both partners have taken a Java class.
  - Neither partner has taken a Java class.
  - No mixed groups.
- Next lecture: What all that public static blah blah stuff actually means.

Course Logistics (Full Version)

Places to Get Information

Official Course Resources

- Course website: http://datastructur.es
- Lectures (or webcasts).
- Piazza: http://piazza.com/class/61b
- Office hours (locations and times TBA).
- Lab (ok to discuss anything, even topics unrelated to that day's lab).
- Discussion.
- Homework Parties.
- 5 student Group Tutoring Sections.
- Mini-Textbook: Working title is just Hug61B for now: http://gitbook.com/book/joshhug/hug61b

Unofficial: Google, Stack Overflow, other programming courses on the web, various online documentation, etc.

Logistical Details

- For waitlisted folks: If you do project 0, I'll do what I can to get you in by week 4.
- If possible, go to your assigned section and lab this week.
  - If you don't have one yet, go to any section or lab.
  - If room is too full, priority goes to those officially registered.
    - Not registered? Please wait outside until 2 minutes before start (XX:08).
- Once attendance settles, doesn't matter which lab/section/lecture you're registered for.
- **Please post administrative issues to Piazza or send an email to cs61b@berkeley.edu**
  - Please don't email me directly (sorry!).
  - 1400 students * 1 minute/student = 24.2 hours.

61B 2.5 - Course Structure

Phase 1: Programming Intensive Introduction to Java.

- Weeks 1-4.
- One browser-based programming HW (this HW0 is optional).
- Three labs to introduce you to various tools (starting this week).
- Two projects (proj0 and proj1).

Phase 2: Advanced Programming

- Weeks 5-7.
- One small HW (HW1).
- One large project, due ~3/5.
  - New: You design your own explorable world (within some constraints).
- Labs to support large project.

61B 2.5 - Course Structure

Phase 3: Data Structures and Algorithms

- Weeks 8-14
- Incredibly important and foundational material: Expect an CS job interview to lean heavily on this part of the course.
- Labs: Implement a data structure or algorithm.
  - Each lab ends with a TA led discussion of best implementation.
- Six HWs: Apply a data structure or algorithm toward a real world problem.
  - Two released during RRR week. Can be used to makeup missed homeworks earlier, or for practice.
- One very challenging data structure/algorithms project (but not as big as project 2).

See calendar at http://datastructur.es for more.

Lab Logistics

- OK to work on labs ahead of time.
- Attendance not required, except for special project labs (more later).
- Lab always due by Friday at 11:59 PM.
- Full credit for ‘reasonable effort’.
  - Some labs are freebies (automatic credit, even if you don’t show up).

14 total labs, worth 8 points each [96 points total].

- Lowest two are dropped. Intended to cover life difficulties.
- **No extensions or grace hours** except emergencies that make you miss > 2 labs.

Discussion Logistics

- Attendance not required, but 2 gold points per discussion you attend (up to a maximum of 20 gold points).
- Attendance not officially taken the first two weeks.

What's a gold point?

- Helps boost your score if you don't do as well as on exams.
- The lower your exam score, the more gold points help.
  - Up to a maximum of counting double if you get a zero on all exams.
  - Not a good strategy to intentionally get zero points on all exams.
- See course info for the full details.

HWs

Homework breakdown:

- HW0: Optional browser based Java exercises.
- HW1: Practice with advanced Java features.
- HW2-7: Applications of various data structures and algorithms.

Due dates vary. See calendar.

7 total required homeworks, worth 32 points each [160 points total].

- Lowest two are dropped. Intended to cover life difficulties.
- **No extensions or grace hours** except emergencies that make you miss > 2 labs.

Vitamins and Study Guides (for conceptual understanding)

Each week, there will be a series of exercises (vitamins) for that week's lectures.

- Due on Sunday at 11:59 PM.
- Relatively short.
- Primarily intended to keep you on track with lectures.
- 4 points each for 48 points (lowest two of 14 are dropped).

For each lecture, there is also a "study guide".

- Provides a brief summary of the lecture.
- Provides (usually) C level, B level, and A level problems for exam studying.
  - A level problems are usually hard enough that I anticipate TAs will have a hard time with them, so be nice!

Projects

Four projects

- One lightweight project, two medium projects, one large project.
  - Project 0 (solo or pair): 50 points
  - Project 1 (solo): 80 points
  - Project 2 (pair): 200 points
  - Project 3 (solo): 150 points
- Projects 2 and 3 will be very time consuming. Plan ahead.
- **All code on solo projects must be your own work.**
- Ok to discuss with others and help debug.

Projects 1, 2, and 3 will have extra credit opportunities.

- Early submission deadline: Bonus points.
- Stretch goals: “Gold” points.

Exams

Exams will be “hard”

- Median scores will be lower than you might be used to (ideally ~60%).
- Two midterms in evenings, one final exam.
- One sheet of paper (front and back) per exam.
- If your midterm grades are statistically much worse than your final, we’ll replace your midterm grade. See “supersession” on course info page for the full details.

Exam dates (midterms tentative until room deadlines confirmed):

- Midterm 1: **February 12th**, 8:00 -10:00 PM (drop deadline is February 16th)
- Midterm 2: **March 20th**, 8:00 - 10:00 PM.
- **May 9th** (final exam) at 7 PM.
- There will be **no alternate exams** (see exam replacement policy).

Course Grade

Breakdown: 1,584 points total.

- Midterms: 400 points total.
- Final: 400 points.
- Projects: 480 regular points.
- HW: 160 points (32 points each)
- Lab: 96 points (8 points each)
- Vitamins: 48 points (4 points each)

Plus occasional opportunities for extra credit for filling out course feedback surveys and projects 1, 2, and 3.

Grades are not curved, i.e. they are not based on your relative performance. In past semesters, grade bin cutoffs have not budged (or if they did, just barely).

- See http://sp18.datastructur.es/about.html for full details including grading bin cutoffs.

Course Pacing

We will start off very fast.

- Optional HW0 is out.
  - Intro to Java syntax.
  - Will take 1-4 hours.
  - Work with friends!
  - Recommended that you complete before your lab.
  - Strongly recommended that you complete by lecture Friday.
- Lab1 and Lab1 Setup are both available.
  - Lab1: How to use various tools.
  - Lab1 Setup: How to set up your home computer (maybe do before lab1).
- Project 0 released Friday. Due next Friday Jan 26th (10 days from start of semester).
  - Exercises all the basic Java features.
  - Allowed to work in pairs (more next time).

Citations

Real-time MRI by the New Scientist: https://www.youtube.com/watch?v=8XQlIvlWqpo

Self-driving car image by The Guardian:
http://www.theguardian.com/technology/2014/may/28/google-self-driving-car-how-does-it-work

Dance Dance Revolution videos from:

https://www.youtube.com/watch?v=OVtnnIifaU8

https://www.youtube.com/watch?v=12lSScKSx20

Questions for You [first lecture]

What do you hope / expect to learn from this class? Why are you taking it?

- Learn how to organize code to make my life easier.
- Get a job with it.
  - Civlization seems to have gone weird in the west, and if I want to bein hte middle class or higher, being able to program certainly seems useful. Let's do that. Required for degree.
- Need it for major: EECS.
- Better understanding of efficiency.
- Learn how to build large projects.
- Subscribe to my spotify list. To undersatnd the algorithm behind it… this class might be marginally helpful.
  - Because you can get a job, and then work at Spotify and then learn how it works.