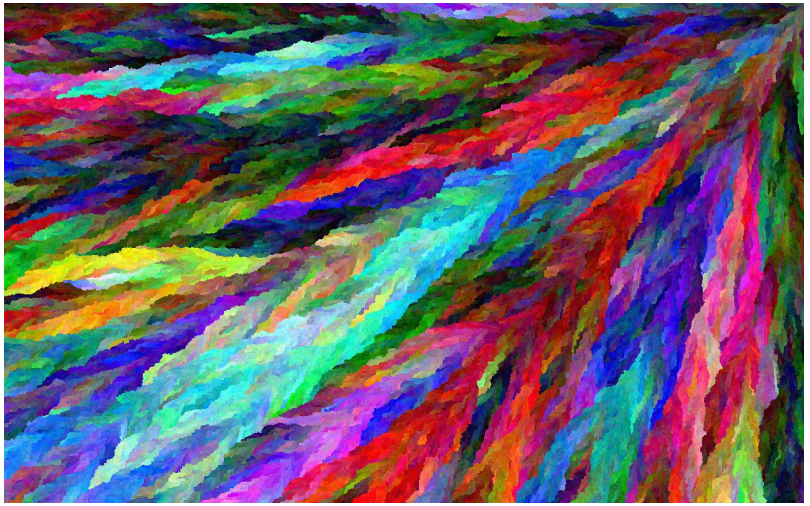
A Note for those Stumbling on these Slides (Hi!)

- 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



CS61B: 2018

Lecture 1:

- Introduction
- Course Logistics
- Hello World

61B Overview

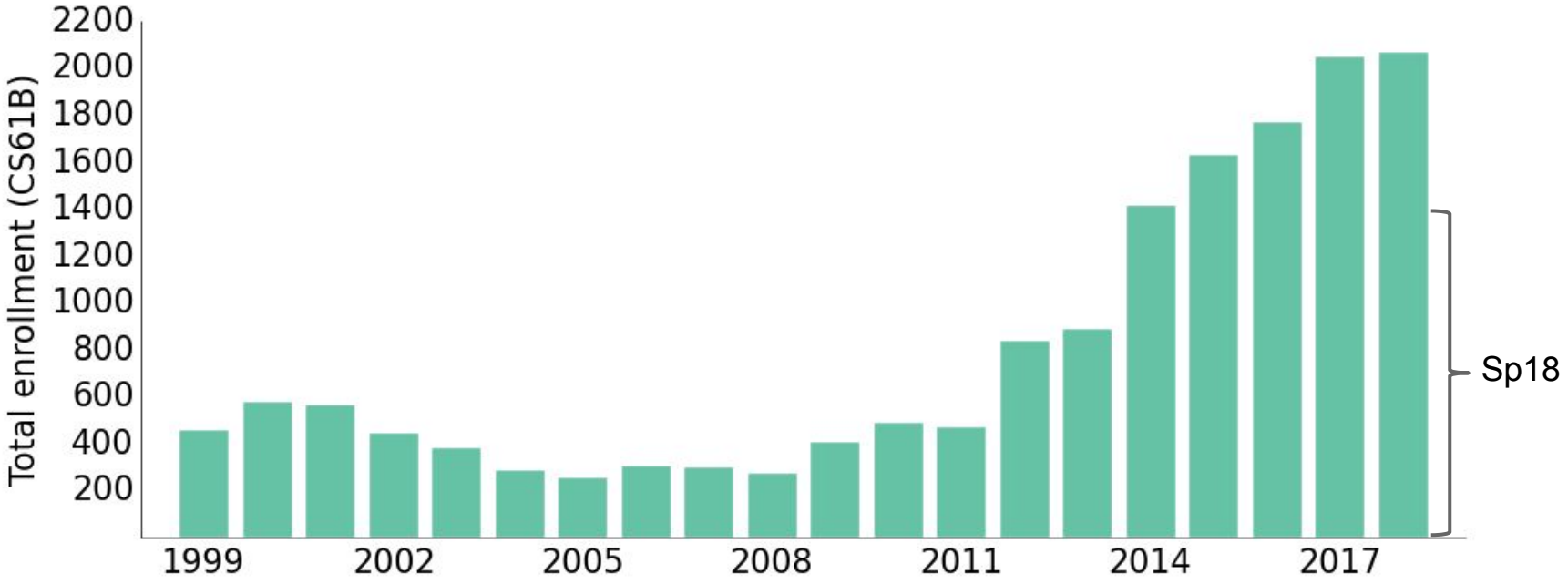
What is 61B about?

- 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:

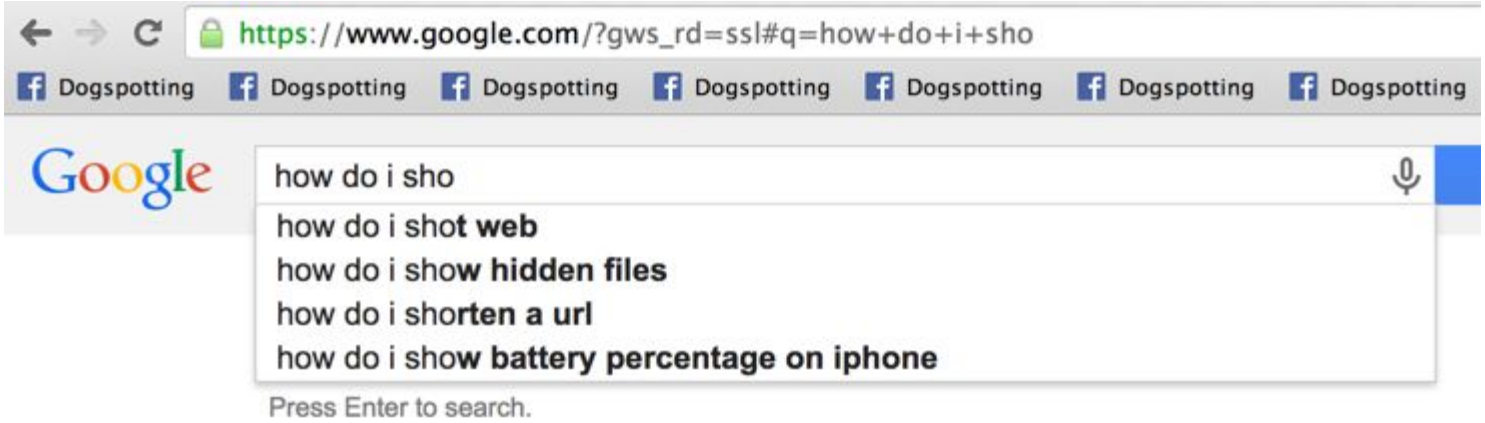
- Object oriented programming, recursion, lists, and trees.

Why Study Algorithms or Data Structures?



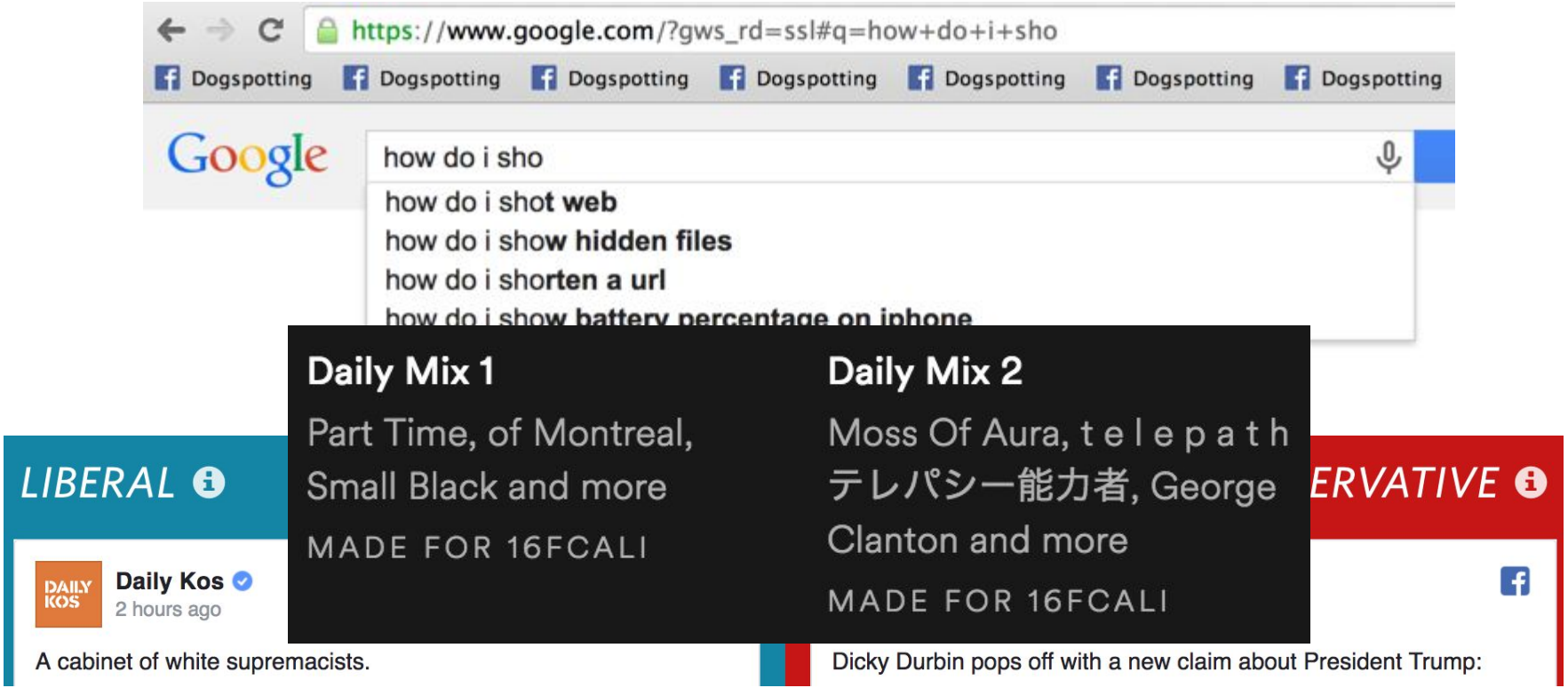
Why Study Algorithms or Data Structures?

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 (see CS195 for more).

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

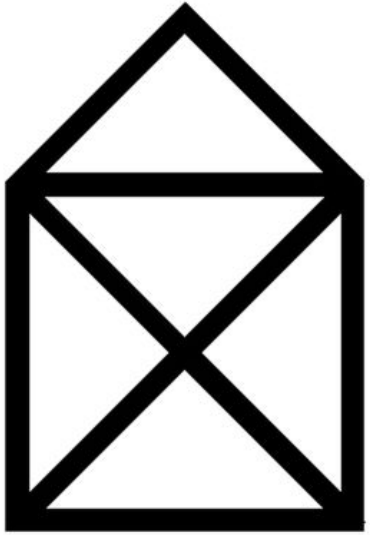
Why Study Algorithms or Data Structures?

To become a better programmer.

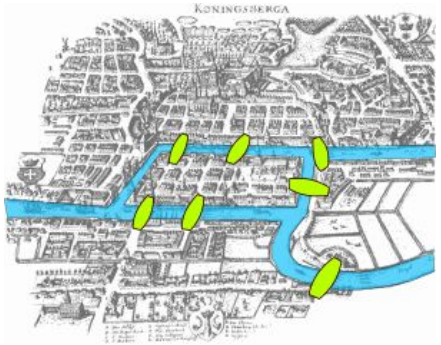
> *"The difference between a bad programmer and a good one is whether [the programmer] considers code or data structures more important. Bad programmers worry about the code. Good programmers worry about data structures and their relationships."* - *Linus Torvalds (Creator of Linux)*

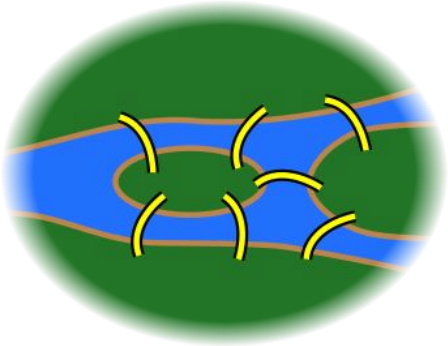
Why Study Algorithms or Data Structures?

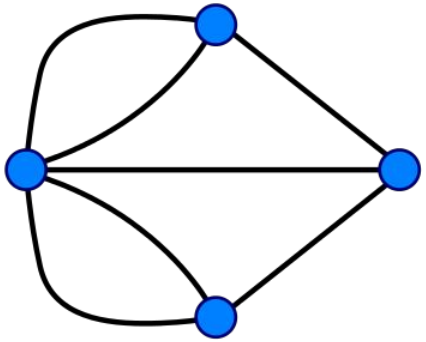
For intellectual stimulation:



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



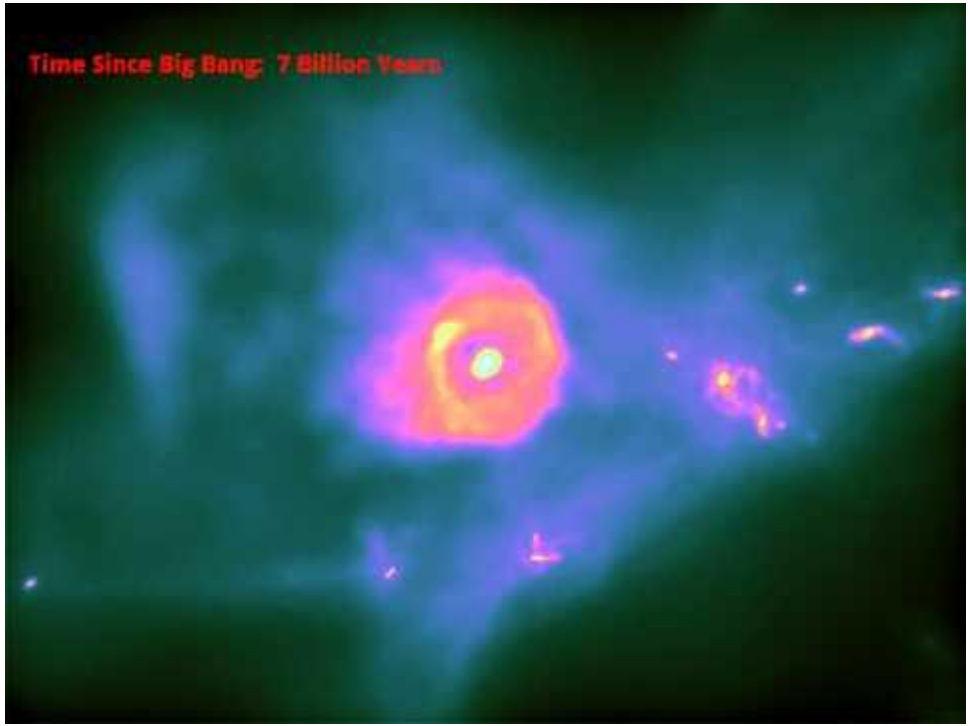




Impossible.

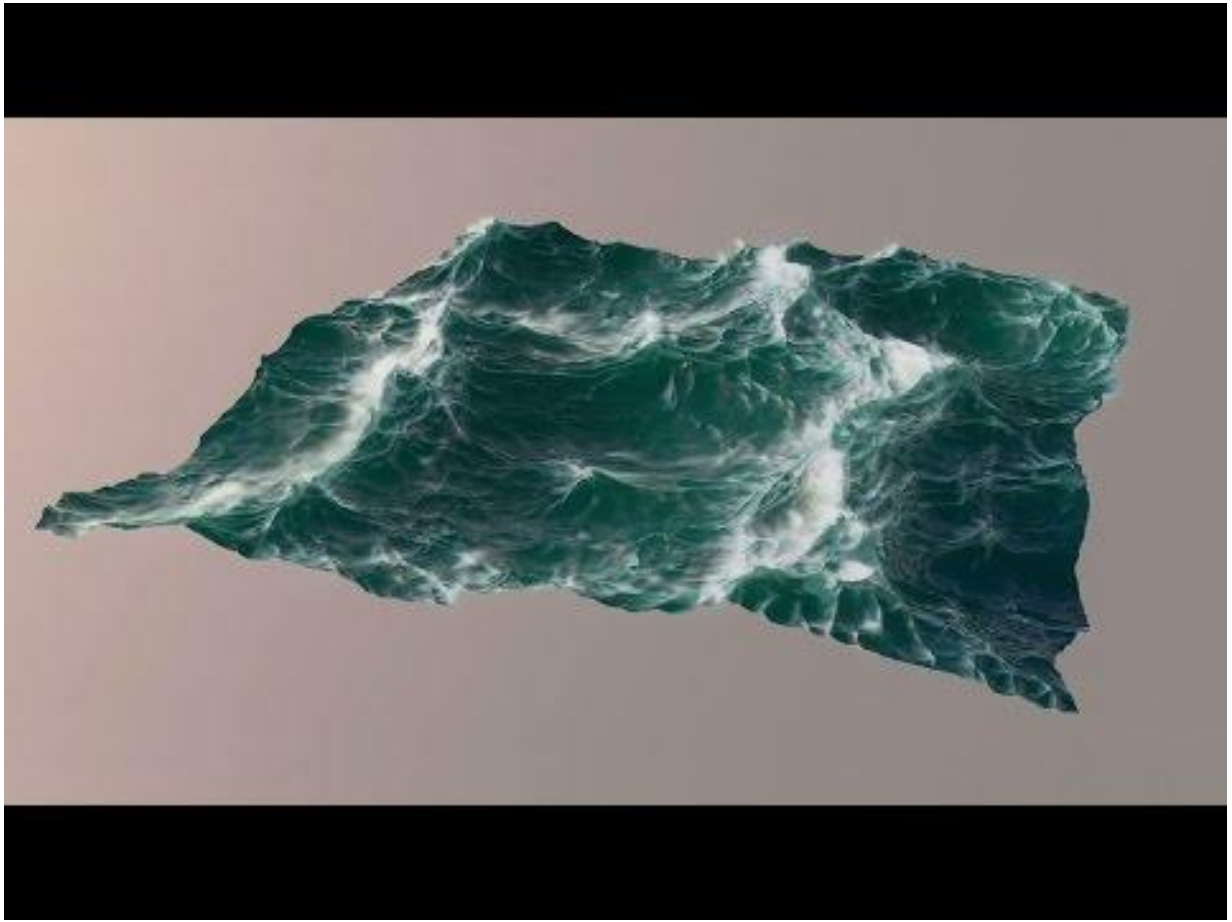
Why Study Algorithms or Data Structures?

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.



Questions for You

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

- Internship: Seems useful for getting internships.
- Gain some sort of abilities that will allow us to make modifications to the world that are good for it.
- Data structures
- Entertainment: homeworks and projects.
- Be a better programming: See gaining abilities.
- Code well enough to change your gradescope grades.
    - Mysterious: through getting it right? Or scammmz
-

Who Are You?

Who Are We?

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

GSIs (and a GSR):

- Part time: Alex Kazorian, Alex Krentsel, Allen Guo, Annie Tang, Ashley Chien, Betty Chang, Catherine Han, Danny Chu, Dorian Chan, Gi-Gi Lu, Jackson Leisure, Jenny Huang, Jeremy Dong, Jiana Huang, Justin Mi, Karuna Wadhera, Kelly Lin, Matthew Owen, Matthew Sit, Michael Ju, Michelle Hwang, Sandy Zhang, Shubham Gupta, **Vivian Fang**
- Full time: Albert Hu, Alex Hwang, Andy Zhang, Ashley Chen, Brandon Lee, **Christine Zhou**, Eli Lipsitz, JC Dy, Josh Zeitsoff, Kevin Arifin, Kevin Chang, **Kevin Lin**, Kevin Lowe, Nicole Rasquinha, Sam Zhou, Ting Ding, Wayne Li, Sarah Sterman, Jim Ren

Who Are We (continued)?

And 91 (at current count) academic interns: Kevin Ko, Boris Yue, Zoe Plaxco, Jingjing Jia, Alan Nguyen, Diana Tu, Samuel Shen, Angela Xiao, Nicolas Raga, Joshua Lin, Tanvee Desai, Stephen Zhou, Nico Camerino, Max Yuan, Moses Kim, Daniel Nguyen, Jasper Gan, Jennifer Xie, Jackie Li, Kevin Deguzman, Shayan Askarian Namaghi, Yu Chen, Janani Vijaykumar, Yiling Kao, Yowsean Li, Justin Lin, Jim Hollingworth, Zipeng Qin, Doreene Kang, Andre Andreassa, Jim Won, Katherine Liu, Min Jae Lee, Elizabeth Avelar Mercado, Natasha Wong, Harsha Nandiwada, Jack Ji, Seung Jin Yang, Yifan Ning, Adel Setoodehnia, Adi Zimmerman, Gabrielle Delforge, Anastasia Vela, Amy Mendelsohn, Shawn Yang, Yuan Gao, Emily Hill, Emily Lan, Rishab Kedia, Daniel Lin, Candace Chiang, Yihui Zhu, Saurav Chhatrapati, Jason Ma, Vibha Seshadri, Robert Chen, Kaan Dogusoy, Rena Chen, Emma D'Esopo, Daniel Li, Nasim Binnur , Brian DeLeonardis, Vera Wang, Camille Harris, Suren Gunturu, Benny, Julianna White, Duy Nguyen, Xu-Bin Kuang, Qian Pan, Lawrence Chen, Aura Barrera, Shruthi Chockkalingam, Auni Bagchi, Tina Zhao, Kevin Chien, Austin Cheng, Kyle Schweizer, Tony Liu, Fabiola Lopez, Kieran Davis, Brian Liao, Michelle Lee, Alex Yi, Jessica Gao, Andrew Raguse, Sai Mandava, Stella Wang, Sydney Yang, Robert Iancu, George Zeng

As well as 16 tutors, TBD.

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, Vitamins, and Study Guides:

- Practice with concepts and Java syntax.

Labs, Weekly(ish) 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 (Condensed)

See Slides at End / Online Video for the 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 (or got removed from the class) 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.

Labs and Discussion

Attendance for lab/discussion is not required, but can earn “gold points” (extra credit) by attending discussion.

Labs:

- Lab always due by Friday at 11:59 PM. Full credit for ‘reasonable effort’.
- Lowest two of 14 labs dropped. **No extensions or grace hours** except emergencies that make you miss > 2 labs.

HWs and Projects

5 standard homeworks and 2 makeup homeworks.

- Lowest 2 of 7 dropped.
- Due dates vary widely, see calendar.
- **No extensions or grace hours** except emergencies that make you miss > 2 labs.

Projects:

- Projects 1 and 3 must be done solo. 0 and 2 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 1, 2, and 3.
- Point deduction per hour late.

Vitamins and Study Guides (for conceptual understanding)

Short lecture exercises, a.k.a vitamins.

- Due on Sunday at 11:59 PM.
- Lowest 2 of 14 are dropped.

1. Intro, Hello World Java

[video] [slides] [guide]

Study guides for each lecture.

- 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.

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. Letter grade will be determined by your 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).

1. Intro, Hello World Java

[video] [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.

Reflections on Static Typing

The Good:

- Debugging is a lot easier, type errors are avoided.
- Produciton code has no type errors, so that means people's phones won't crash because of type errors.
- Programs run more efficiently in time and memory.
- Self-documenting: YOU KNOW WHAT YOU'VE GOT.

The Bad:

- Code is more verbose.
- Code is less general.

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 coming soon (out by Friday, maybe sooner). 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.