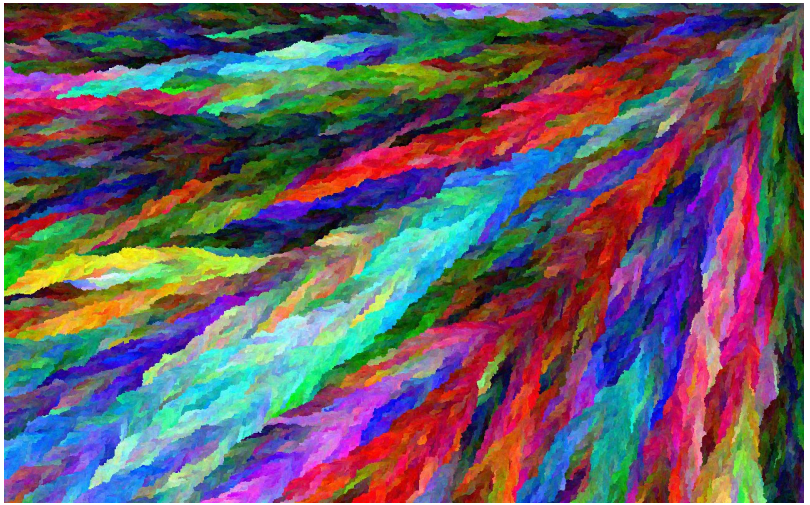
A Note for those Stumbling on these Slides (Hi!)

- 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: 2021

Lecture 1:

- Introduction
- Course Logistics
- Hello World

61B Overview

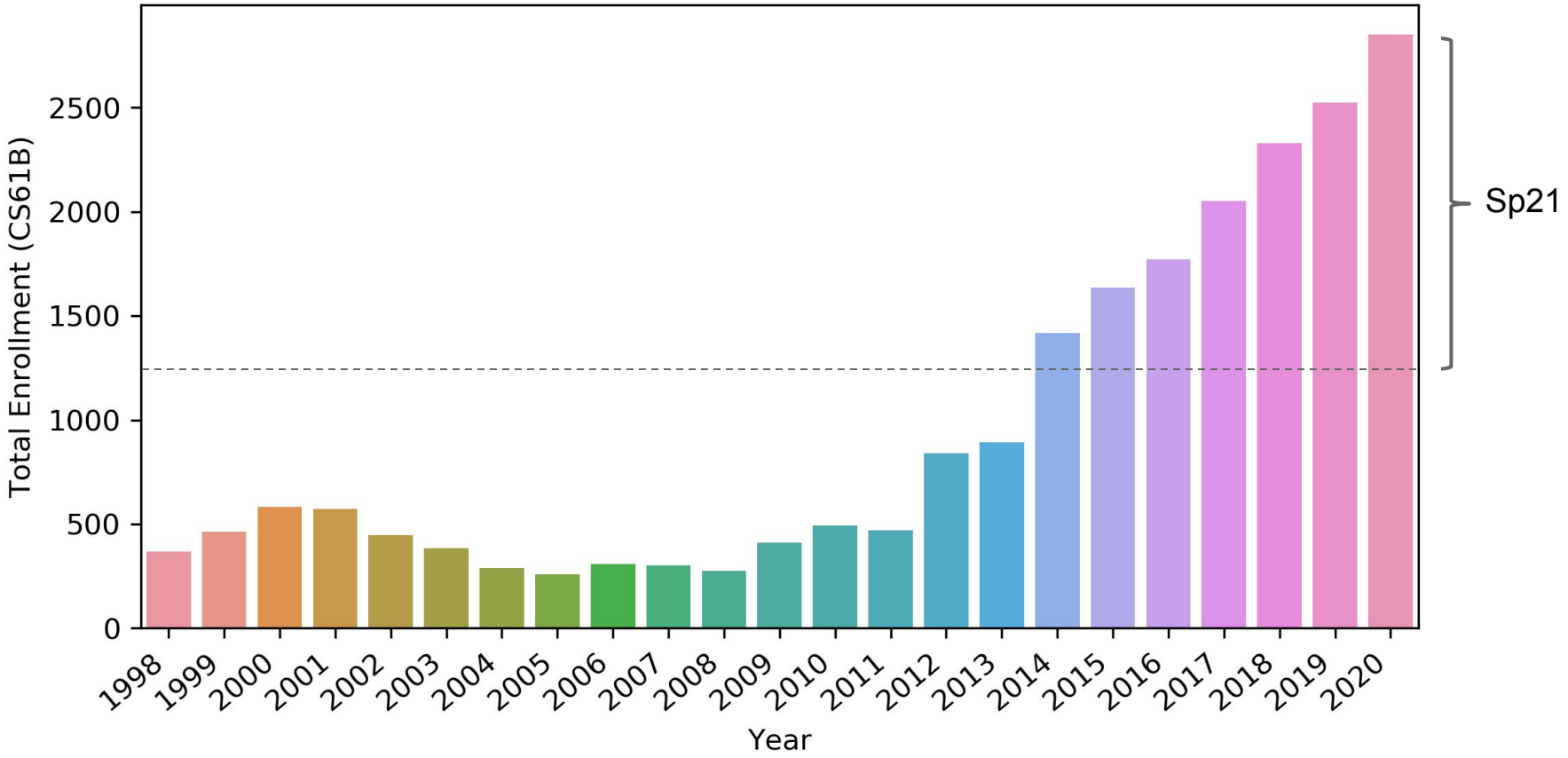
What is 61B about?

- 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:

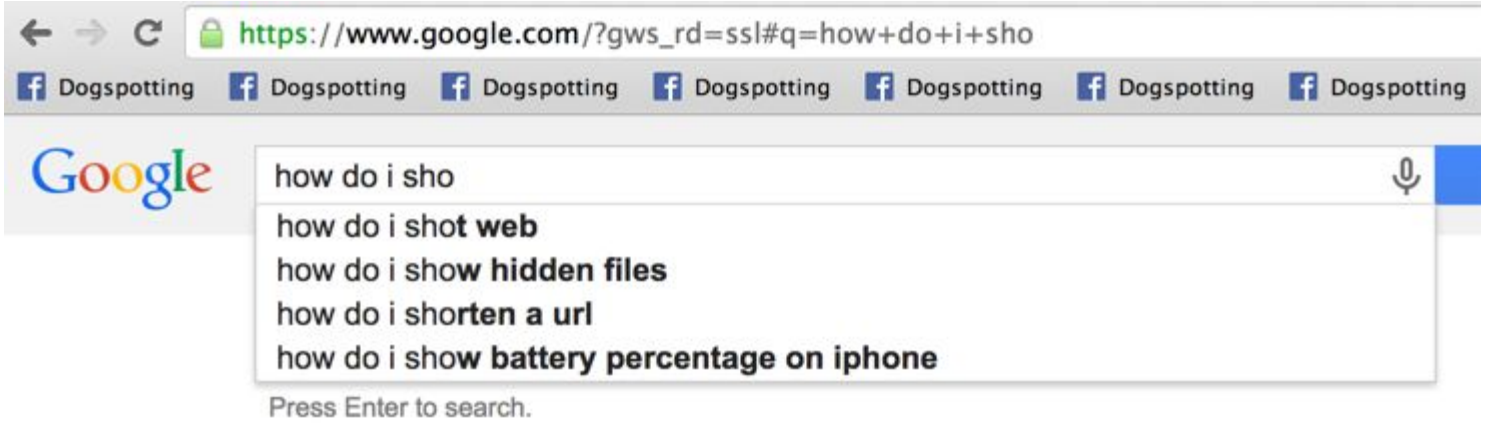
- Object oriented programming, recursion, lists, and trees.

Why Study Algorithms or Data Structures?



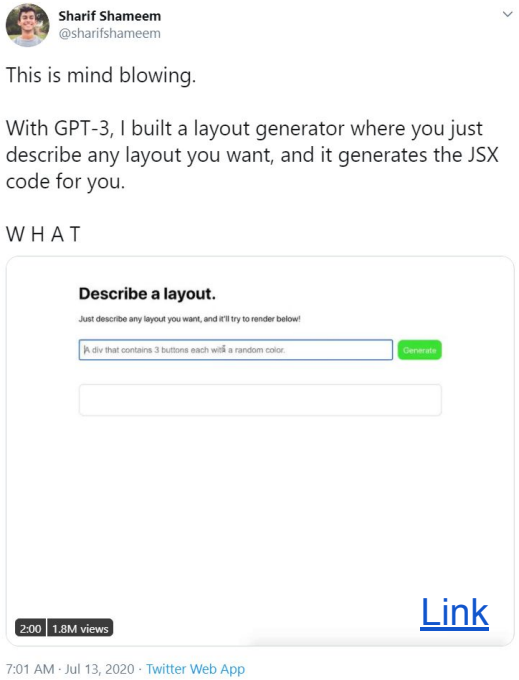
Why Study Algorithms or Data Structures?

Daily life is supported by them.



Why Study Algorithms or Data Structures?

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



Why Study Algorithms or Data Structures?

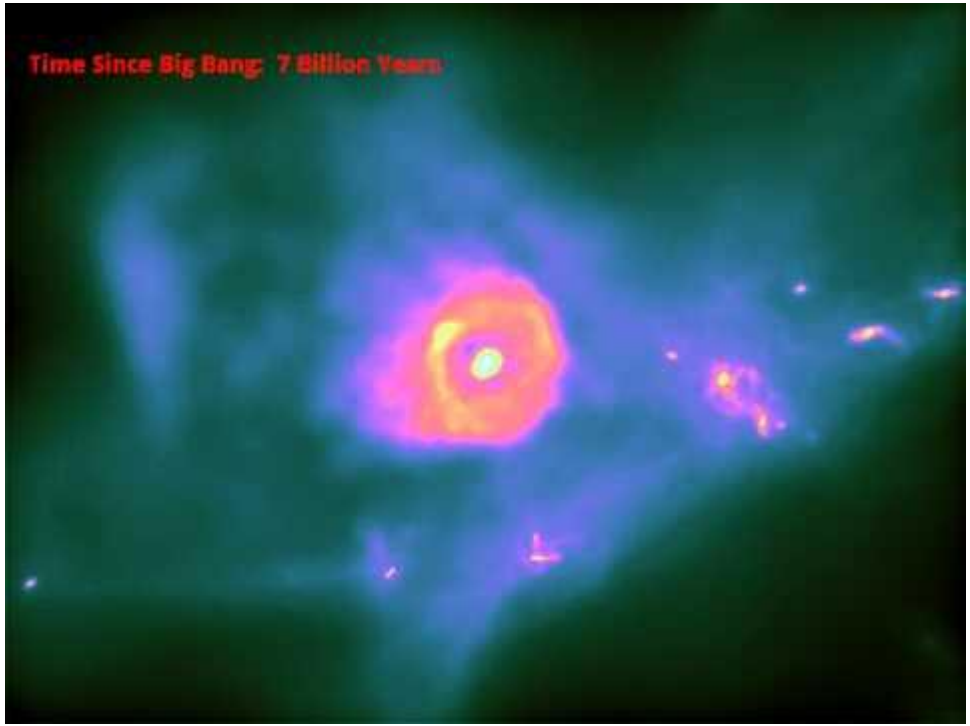
To become a better programmer.

> *"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:



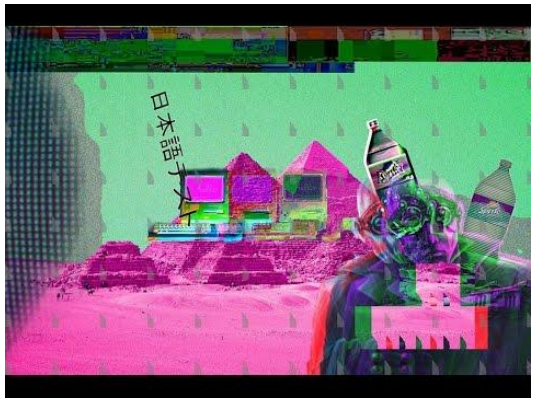
Why Study Algorithms or Data Structures?

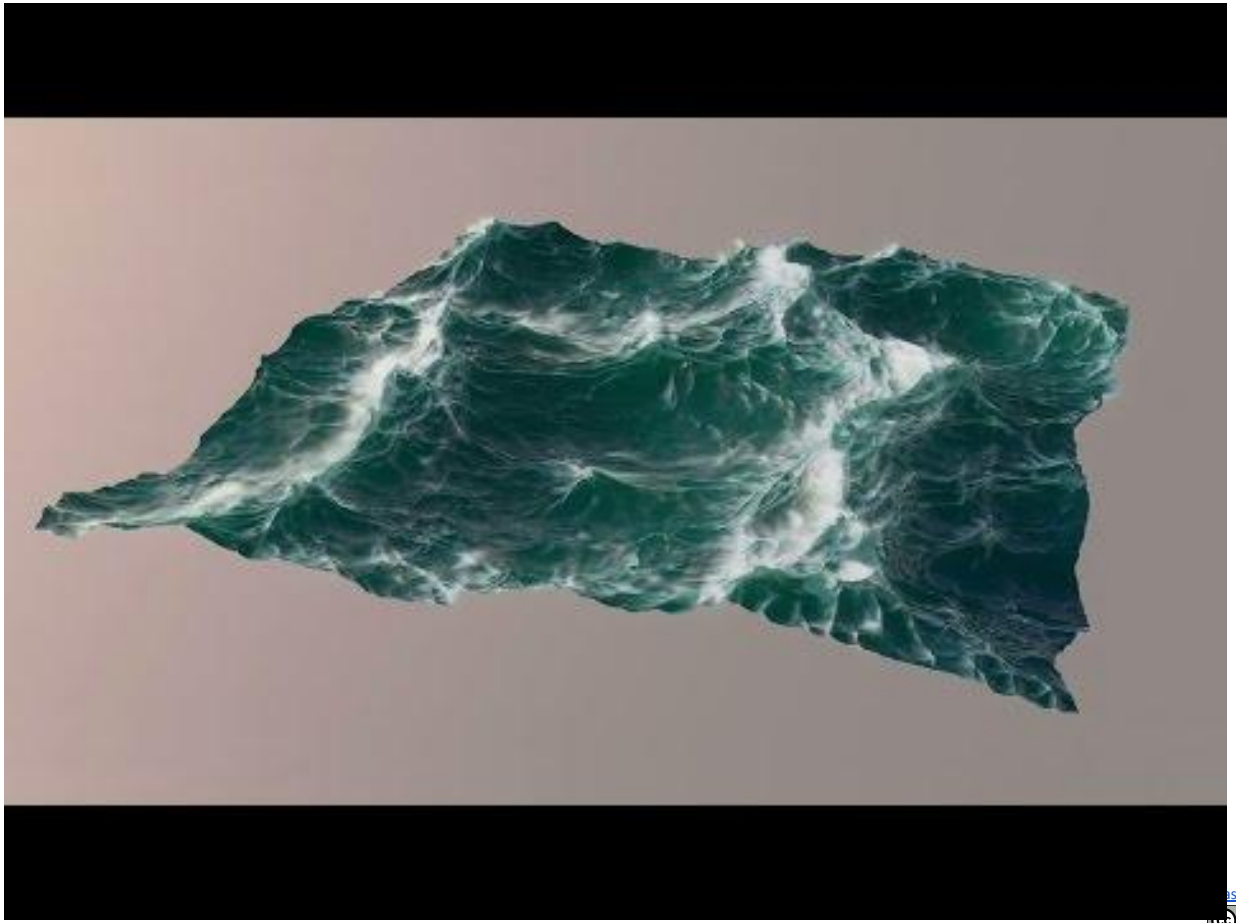
to

create

beautiful

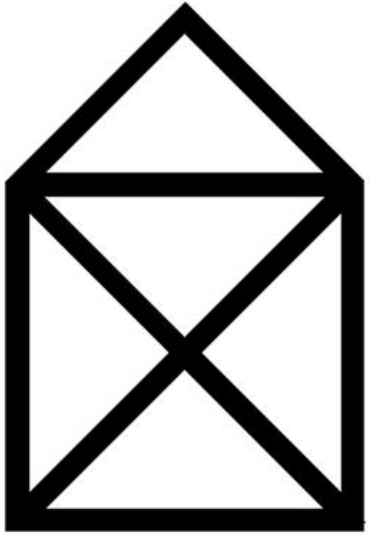
things



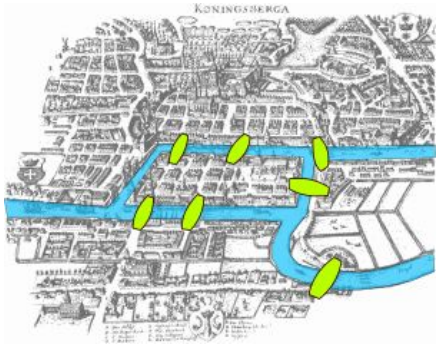


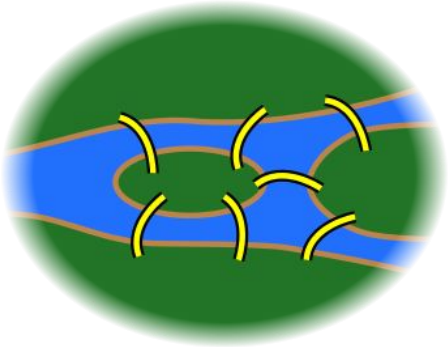
Why Study Algorithms or Data Structures?

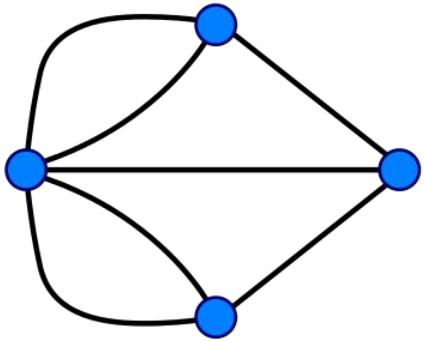
As an end unto itself.



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?

Who Are You?

Assuming the Zoom webinar polls are set up correctly, let's do a quick poll.

- Year?
- Prior Java experience?
- Which pre-req did you take?

Who Are We?

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

GSIs:

- 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
- Part time
  - 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
- Full time
  - 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

Who Are We? (continued)

Academic interns:

Tanya Jain

Justin Thein

Kyle Yu

Michael Huang

Xinling Yu

Rahul Mohankumar

Deepak Ragu

Tymon Thi

Aryan Agrawal

Devin Sze

YiTing Yan

David Chung

Viansa Schmulbach

Ritik Batra

Yash Gupta

Arda Demirci

Eric Pineda

Anik Gupta

Nitya Goyal

Sonja Johanson

Stephanie Jue

Dexter To

Sean Hayes

Olivia Huang

Genevieve Brooks

Kush Garg

Reza Sajadiany

Michelle Cheung

Maryam Azmandian

Arth Vidyarthi

Amanda Yao

Arvind Rajaraman

Andrew Lee

Anrui Gu

Haroun Khaleel

Ansh Nanda

Catherine Hwu

Stephen Yang

Jerry Sun

Aady Pillai

Emmett Dreyer

Kaci Gu

Richard Lee

Suhrid Saha

Sunay Dagli

Anne Nguyen

Fakhri Widodo

Shefali Goel

Alina Trinh

Amritansh Saraf

Siddhant Sharma

Phoebe

Troup-Galligan

Steven Chen

Who Are We? (continued (continued))

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

Elise Ong

Ola Alsaedi

Shivani Ahuja

Tiffany Luu

Isha Arora

Aditya Prasad

Daniel Detchev

Grace Chen

Austin Ho

Nicholas Cheng

Shannon Bonet

Jane Lee

Kaaviya Sasikumar

Ankita Janakiraman

Leo Kao

Kevin Jin

Kuhu Sharma

Vivian Liu

Xinyu Fu

Christina (Siting) Shao

Sofia Roz

Annie Wang

Learning Philosophy

The Manner in Which Learning Occurs (TMWLO)

TMWLO

Small minority of your learning:

- **Introduction to new material**: Lectures / reading.

The vast majority of your learning:

- **Theory**: Discussion sections, study guides, theory homework.
- **Programming, Tool Usage, Problem Decomposition:** Labs, coding HW, projects.
- **Design:** Projects 2 and 3.

Course Logistics

Places to Get Information

Official Course Resources

- Course website: http://datastructur.es
- Lectures. Discussion. Lab. Office Hours.
- Ed Discussions: https://edstem.org/us/courses/3735/discussion/
- Mini-Textbook: Obscurantism in Java https://joshhug.gitbooks.io/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, but no guarantees.
  - We should be able to accommodate everyone, but I can't 100% promise.
- To sign up for a discussion section or lab, see https://edstem.org/us/courses/3735/discussion/213629
- **Please post administrative issues to Ed or send an email to cs61b@berkeley.edu**
  - Please don't email me with such issues directly (sorry!).
  - 1600 students * 1 minute/student = 26 hours.

61B 4.0 - Course Structure

Phase 1: Programming Intensive Introduction to Java.

- Weeks 1-4.
- One browser-based programming HW (this HW0 is optional).
- Four labs to introduce you to various tools (starting this week).
- Two projects (proj0 and proj1).
- Midterm 2/10 at time TBD.

61B 4.0 - 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.
- One programming HWs (HW1) and one exam-prep theory HW (HW2).
  - Applications and deeper insight into data structures.
- One very large solo project (Proj 2), due 4/2. Checkpoint due 3/12.
- Labs:
  - Lab 5: Peer review on project 1.
  - Two labs that implement data structures (hash table and BST).
  - Remaining labs are focused on project 2.
- Midterm 3/17 at time TBD.

61B 4.0 - Course Structure

Phase 3: Algorithms and Software Engineering.

- Weeks 10-14
- Project:
  - Proj 3: Build Your Own World: An open ended project where you and a partner build a 2D world with physics according to your own design. Due during lab in the last week of the class.
- Labs devoted to project.
- One theory homework due 5/3.

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

Labs and Discussion

Attendance for lab/discussion is not required.

- Cameras must be on to attend.
- If you don’t cameras on, offline versions will be provided.

Labs:

- Lab always due by Friday at 11:59 PM.
- Full credit for ‘reasonable effort’.
  - Only have to pass some of the autograder tests for full credit.

Collaboration Policy

Collaboration:

- You will submit your own solution to all HWs and projects 0, 1, and 2.
- For labs, you may optionally have a partner.
- For project 3, you must have a partner.

All code on solo projects, HWs, labs must be your own work.

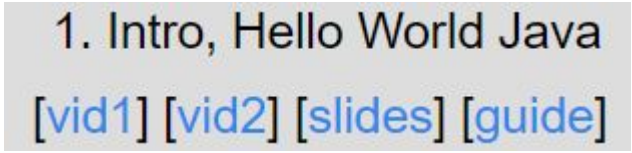
- Ok to discuss with others and help debug.

There are more details on the course website. You are responsible for knowing these policies!

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.

- 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!
- Study guide includes a check-in exercise. No credit, but useful!

Exams

- Closed note except you can bring cheat sheets.
- Will be pretty hard (60-65% medians).
- Showing improvement on final can boost overall exam score.
  - If your final is statistically better, it can replace both of your midterms. See “clobbering” under course website for details.

Exam dates:

- Midterm 1: **February 10th**, time TBD. Will accommodate timezones.
- Midterm 2: **Match 17th**, time TBD. Will accommodate timezones.
- Final Exam Times: **May 11th,** 8 AM Pacific time. Will accommodate timezones.
- There will be **no alternate exams** (see exam replacement policy).
  - One exception: Direct conflict with a final in another class.

Course Grade

Breakdown: 12,800 points total. Letter grade will be determined by your total.

- Midterms: 3200 points total.
- Final: 3200 points.
- Projects: 4480 regular points.
- HW: 960 points (320 points each)
- Lab: 640 points (64 points each)
- Weekly Surveys: 320 points (32 points each)

Plus extra credit for filling out pre-, mid-, and post- semester course surveys.

Plus extra credit for project checkpoints.

Plus extra credit for project stretch goals.

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://sp21.datastructur.es/about.html for full details including grading bin cutoffs (values not chosen yet, but will be by week 3).

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!
- If needed, use automated extension system to request extensions.
  - Up to 6 total slip days.
  - Maximum of 3 per assignment.
  - Don't expect similar extensions in other classes. 61B is not as strongly cumulative as classes like EE16A or CS70.

During week 2, you will 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.
- Reach out to mentor GSI if you are falling behind.

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 Setup: How to set up your home computer (do before lab1).
  - Lab1: How to use various tools.
- Project 0 released Friday. Due Friday Jan 29th (9 days from start of semester).
  - Start by Saturday if possible, especially if you're new to Java.
  - Exercises basic Java features.
  - Full credit requires solving a very challenging algorithmic problem.

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.

Reflections on Static Typing

The Good:

- Catches certain types of errors, making it easier on the programmer to debug their code.
- Type errors can (almost) never occur on end user's computer.
- Makes it easier to read and reason about code.
- Code can run more efficiently, e.g. no need to do expensive runtime type checks.

The Bad:

- Code is more verbose.
- Code is less general, e.g. had to have two different larger functions earlier.

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. 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 (released Friday). Must be done solo.
- Next lecture: What all that public static blah blah stuff actually means.