

UC Berkeley EECS
Sr Lecturer SOE
Dan Garcia

The Beauty and Joy of Computing

Lecture #1
Welcome; Abstraction



UC Berkeley EECS
Lecturer
Gerald Friedland

BJC: YOU'LL LOVE IT!

Watch the student testimonials about the course, what it means to them, and how it has changed their lives.
Inspiring!



inst.eecs.berkeley.edu/~cs10/



BJC in one slide

Big Ideas of Programming

- Abstraction
- Algorithms (2)
- Recursion (2)
- Functions-as-data, λ (2)
- Programming Paradigms
- Concurrency
- Distributed Computing

Beauty and Joy

- "CS Unplugged" activities
- All lab work in pairs
- Two 3-week projects in pairs
 - Of their own choice!! (data + prog)
- One writeup
 - Of students' own choice!!

Big Ideas of Computing

- HowStuffWorks
 - 3D Graphics + Video Games
 - Internet
- Research Summaries
 - Al
 - HCI
- The Power of Data (big, small, etc)
- Apps that Changed the World
- Social Implications of Computing
- Saving the World with Computing
- Cloud Computing
- Limits of Computing
- Future of Computing







Format & Textbooks

Format (7 hrs/wk * 14 wks)

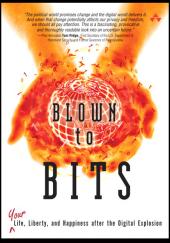
Mon	Tue	Wed	Thu	Fri
Lecture	Lab	Lecture	Lab	Discussion
	Lab		Lab	

Selected Reading

- Taken from great book ("Blown to Bits" by Abelson, Ledeen & Lewis) + articles + videos
- Current events EVERY LECTURE
 (e.g., IBM's Watson vs Jeopardy)

All resources FREE

- Even clickers!
- Pair Programming!



HAL ABELSON . KEN LEDEEN . HARRY LEWIS



IS ABSTRACTION THE KEY TO COMPUTING?

Why is it that some software engineers and computer scientists are able to produce clear, elegant designs and programs, while others cannot. Is it possible to improve these skills through chacation and training? Critical to these meetion is the main of sharters in

JEFF KRAMER

For ever 50 years. I have been involved in teaching and reason fit to computer exister and offerware registering, the reason fit to experience transpire from courses in programming, of the properties of the properties of the properties of the standard of the properties of the properties of the standards are also perform problems assoling, conceptualization, modeling, students are clearly able to handle complexity and to produce depart models and origins. The same sendents are also able to cope with the complexities of distributed algorithms, the applicability of vitams modeling noteston, and other

contributed articles

al fluency" should mean design

creating, and remixing, not just browsing chatting, and interacting.

IT MITCHEL RESIDER, JOHN MALENEY, ANDRÉS HONEY

Scratch: Programming for All

WHEN MORRHY V. VARIN, Editor-in-Chief of Communications, invited us to submit an article, be recalled how be first learned about Screatic. "A colleague of mine (CS faculty," he suid, "told me how she tried to get her 10 year-old daughter interested in programming, and the only thing that appealed to her was Secusion." That's what we were hoping for when we set out the develop Screation superare and superare such superared was well as the superared when we not out the develop Screation superared was superared.

That is what we were boping for when we set out to develop Sersich silvy arm age. We wented to develop an approach to programming that would appeal to people who had fair T peofusoly imaging the timesters as programmens. We wanted to make it easy for everyone, edial ages, hadegounds, and internet, to program their own interactive stories, gurnes, animations, and shimulations, and share their restrictions with one another. Since the public launch in May 2007, (the Sentath Web site (https://ceartch.mit.edu.) has becream a

E (1900)

opini om den John den opinijen semidale behanged entempt ble semidale entempt ble semidale

The state of the s







Week at a glance

Weekly Schedule											
Hour	Monday	Tuesday	Wednesday	Thursday	Friday						
7:00am		Lab Section J	- Hamasa,	Lab Section J		- 11144					
8:00am		(Zach) 200 Sutardja Dai		(Zach) 200 Sutardja Dai			Discussion Section A (Zach) 310 Soda Hall				
9:00am		Lab Section A	Lab Section B (Max) 200 Sutardja Dai	Lab Section A (Zach) 200 Sutardja Dai	Lab Section B (Max) 200 Sutardja Dai						
10:00am		(Zach) 200 Sutardja Dai					Discussion Section B (lan) 310 Soda Hall				
11:00am		Lab Section C (lan)	Lab Section D (Michael)	Lab Section C	Lab Section D (Jon)		Discussion Section G (Michael) 310 Soda Hall				
12:00pm		200 Sutardja Dai	(Michael) 200 Sutardja Dai	(lan) 200 Sutardja Dai	200 Sutardja Dai	Discussion Section C (Max) 310 Soda Hall	Discussion Section D (Zach) 320 Soda Hall				
1:00pm	Lecture <u>Li Ka Shing 245</u>	Lab Section E (Peter)	Lecture Li Ka Shing 245	Lab Section E (Peter)			Discussion Section E (Jon) 310 Soda Hall				
2:00pm		200 Sutardja Dai		(Peter) 200 Sutardja Dai		Discussion Section H (Jon) 310 Soda Hall	Discussion Section I (Peter) 320 Soda Hall				
3:00pm	Lab Section K (Michael)	Lab Section G (Max) 200 Sutardja Dai	Lab Section K (Jon) 200 Sutardja Dai	Lab Section G (Max) 200 Sutardja Dai		Discussion Section J (Max) 310 Soda Hall	Discussion Section K (Peter) 320 Soda Hall				
4:00pm	200 Sutardja Dai						Discussion Section L (Samir) 310 Soda Hall				
5:00pm	Lab Section I (Jon)	Lab Section H (Samir) 200 Sutardja Dai	Lab Section I (Jon) <u>200 Sutardja Dai</u>	Lab Section H (Samir) 200 Sutardja Dai			Discussion Section M (Michael) 310 Soda Hall				
6:00pm	200 Sutardja Dai										
7:00pm	Lab Section L (Michael)		Lab Section L (Michael)								
8:00pm	330 Soda Lab Section M	1	271 Soda Lab Section M (Peter)								
9:00pm	273 Soda		273 Soda								







Let's check enrollments (in real time)

- We have NEVER turned anyone away ... if more students sign up, we'll open up more sections!
- We don't intend to turn anyone away now







Peer Instruction

- Increase real-time learning in lecture, test understanding of concepts vs. details
- As complete a "segment" ask multiple choice question
 - 1-2 minutes to decide yourself
 - 2 minutes in pairs/triples to reach consensus. Teach others!
 - 2 minute discussion of answers, questions, clarifications



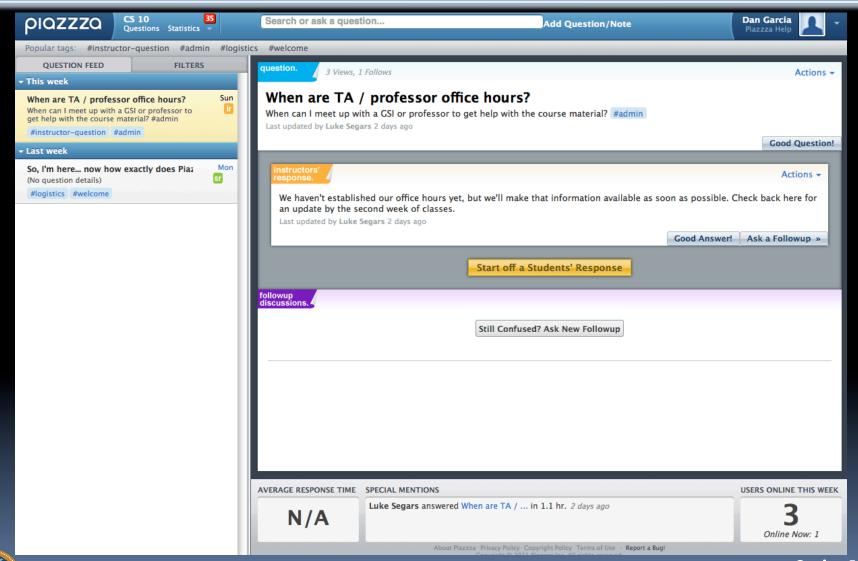








Piazza for {ask,answer}ing questions









Pro-student Grading Policies

EPA

- Rewards good behavior
- <u>Effort</u>
 - E.g., Office hours, doing every single lab, hw, reading Piazza pages
- Participation
 - E.g., Raising hand in lec or discussion, asking questions on Piazza
- Altruism
 - E.g., helping other students in lab, answering questions on Piazza

You have 3 "Slip Days"

- You use them to extend due date, 1 slip day for 1 day extension
- You can use them one at a time or all at once or in any combination
- They follow you around when you pair up (you are counted individually)
 - E.g., A has 2, B has 0.Project is late by 1 day.A uses 1, B is 1 day late
- Late is 1/3 off/day







Abstraction

Detail removal

"The act or process of leaving out of consideration one or more properties of a complex object so as to attend to others."

Generalization

 "The process of formulating general concepts by abstracting common properties of instances"



Henri Matisse *"Naked Blue IV*"







Detail Removal



Automatic Generation of Detail Maps Maneesh Agrawala (UCB EECS), among others







Detail Removal (in BJC)

- You'll want to write a project to simulate a realworld situation, or play a game, or ...
- Abstraction is the idea that you focus on the essence, the cleanest way to map the messy real world to one you can build
- Experts are often brought in to know what to remove and what to keep!



The London Underground 1928 Map & the 1933 map by Harry Beck.







Generalization Example

- You have a farm with many animal kinds.
- Different food for each
- You have directions that say
 - To feed dog, put dog food in dog dish
 - To feed chicken, put chicken food in chicken dish
 - To feed rabbit, put rabbit food in rabbit dish
 - □ Etc...
- How could you do better?
 - To feed <animal>, put <animal>
 food in <animal> dish







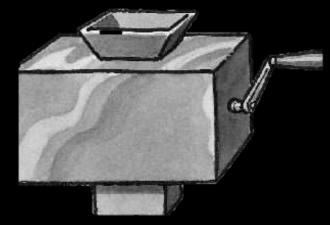


Generalization (in BJC)

You are going to learn to write functions, like in math class:

$$y = \sin(x)$$

 You should think about what inputs make sense to use so you don't have to duplicate code



"Function machine" from *Simply Scheme* (Harvey)







The Power of Abstraction, everywhere!

Examples:

- Functions (e.g., sin x)
- Hiring contractors
- ApplicationProgramming Interfaces(APIs)
- Technology (e.g., cars)
- Amazing things are built when these layer
 - And the abstraction layers are getting deeper by the day!

We only need to worry about the interface, or specification, or contract NOT how (or by whom) it's built

Above the abstraction line

Abstraction Barrier (Interface)(the interface, or specification, or contract)

Below the abstraction line

This is where / how / when / by whom it is actually built, which is done according to the interface, specification, or contract.



Garcia + Friedland

(C) (S) (9)



Summary

- Abstraction is one of the big ideas of computing and computational thinking
- Think about driving. How many of you know how a car works? How many can drive a car? Abstraction!



Someone who drove in 1930 could still drive a car today because they've kept the same Abstraction!

(right pedal faster, left pedal slow)



