Overview

- Preliminaries.
  - Admin.
  - Questions.
- The subject matter(s) of the course.
- Evaluations.
- Final comments from Sam (at 10:45)

Preliminaries

Admin

- Food!
- Review session tomorrow at 10 a.m. - A chance to talk about the finals.
- I was happy to see at yesterday’s budget session that our long-term model continues need blind admission and meeting full demonstrated need.

Upcoming Work

- Continue to work on the exam.
- Fill in the final preference sheet

Extra Credit

- Analytics and student success, today, noon JRC 101.
- CS table Friday: Casual conversation.
- Conference track meet Friday and Saturday. NBB runs at 4:05 and 5:10.
- Listen to EB’s radio show on KDIC Friday at 5pm.
- Listen to DNP guest star on some radio show Friday at 11pm.

Questions on the Exams

What, exactly, do you want for 2b?

Random tests are nice because they catch cases you wouldn’t think of. But "your code is wrong" is not helpful. Need a way to see how the tree was built so, which helps identify problems. Right now: Gives you the list of operations. You want Java code. Do it slowly by hand or write a program. Your
goal: Write the program.

*Can we change Randy’s randomized testing code?*

Yes. I assume you will change what you put into the ArrayList and how you print out the ArrayList.

*In that same question, you give a block of code that builds a tree and prints a trace. What’s the point?*

That’s a good model for you output.

**The subject matter(s) of the course**

**Java**

- Basic imperative stuff: Loops, Conditionals, Variables, etc.
- Running from the command line
- Javadoc
- Iterators
- Interfaces and Classes

**Algorithms**

- Big O / asymptotic analysis
- "The Literature"
  - Quicksort
  - Merge sort
  - Heap sort
  - Binary search
  - Recursive Parsing
  - Tree traversal
- General algorithm design techniques
  - Divide and conquer
  - Dynamic programming
- Practice / A technique for designing algorithms

**OOD and Software Construction**

- Big three: Encapsulation, Inheritance, Polymorphism
- Design patterns
  - Factories
  - Iterators
  - ...
- Testing - How and why
- The benefits and flaws of IDEs
- Make useful error messages!
- Documentation strategies
Loop invariants as ways analyzing correctness and designing loops
Version control (with Git); strategies for doing it well
Mechanisms for representing objects (e.g., Hash tables can be thought of as generalized objects)
Format to meet your company’s standards

Abstract Data Types

- An approach to designing ADTs.
- "The Literature"
  - Priority queues
  - Queue
  - Stacks
  - Lists
  - Arrays / Vectors
  - Dictionaries
  - ...

Data Structures

- An approach to designing and building data structures - implement ADTs
- "The Literature"
  - Hash tables
  - Skip lists
  - Heaps
  - Binary search trees
  - Array-based implementations
  - Node-based implements
  - Trees

General Skills

- Collaboration (and why it’s not always the right thing for you)
- It’s okay to say "I don’t know" once in a while
- It’s okay to challenge authority, but do so politely
- It’s okay to challenge your peers
- Pitching a product

Miscellaneous

- Eclipse
- Android
- Tarballs

And Beyond
- Care about the people around you
- Laugh
- Feed people
- There’s more to life than CS
- Computers and CS faculty are sentient and malicious

**Evaluations**

- Fill them out.
- Have someone bring them to the science division office.

Copyright (c) 2013-14 Samuel A. Rebelsky.

This work is licensed under a Creative Commons Attribution 3.0 Unported License. To view a copy of this license, visit http://creativecommons.org/licenses/by/3.0/ or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.