Announcements

- Lab sections on December 5, 6, and 7 will be organized as follows: Students will work on an exam-like set of exercises covering linked lists, stacks, queues, binary trees, binary search trees. Solutions will be thoroughly reviewed. 1 bonus point (out of 200) for completing the exercises.
- Please use git-bug for problems with submission, your code, the skeleton, or any of our software.
- **Tutors and lab assistants needed.** Consider volunteering to be a tutor or lab assistant for CS 10, self-paced courses, CS 61A, or CS 61B next semester.
- **Programming Contest:** Visit my web page for information about the annual programming contest, which we hold each fall. There are large collections of programming problems you can try your hand on.

Lecture #40: Course Summary

- Programming language: Java
- Program Analysis
- Categories of data structure: Java library structure
- Sequences
- Trees
- Searching
- Sorting
- Pseudo-random numbers
- Graphs
- Pragmatic implementation topics

Programming-Language Topics

- Object-based programming: organizing around data types
- Object-oriented programming:
  - Dynamic vs. static type
  - Inheritance
  - Idea of interface vs. implementation
- Generic programming (the <· · ·> stuff).
- Memory model: containers, pointers, arrays
- Numeric types
- Java syntax and semantics
- Scope and extent
- Standard idioms, patterns:
  - Objects used as functions (e.g., Comparator)
  - Partial implementations (e.g., AbstractList)
  - Iterators

– Views (e.g., sublists)

Analysis and Algorithmic Techniques

- Asymptotic analysis
- $O(\cdot)$, $o(\cdot)$, $\Omega(\cdot)$, $\Theta(\cdot)$ notations
- Worst case, average case.
- Amortized time
- Memoization and dynamic programming.

Major Categories of Data Structure

- Collection interface and its subtypes
- Map interface and its subtypes
- Generic skeleton implementations of collections, lists, maps (`AbstractList`, etc.)
- Complete concrete collection and map classes in Java library

Sequences

- Linking:
  - Single and double link manipulations
  - Sentinels
- Linking vs. arrays
- Stacks, queues, deques
- Circular buffering
- Trade-offs: costs of basic operations

Trees

- Uses of trees: search, representing hierarchical structures
- Basic operations: insertion, deletion
- Tree traversals
- Representing trees

- Game trees

Searching

- Search trees, range searching
- Multidimensional searches: quad trees.
- Hashing
- Priority queues and heaps
- Balanced trees
  - Rebalancing by rotation (red-black trees)
  - Balance by construction (B-trees)
  - Probabilistic balance (skip lists)
  - Tries
- Search times, trade-offs

Sorting

- Uses of sorting
- Insertion sort
- Selection sorting
- Merge sort
- Heap sort
- Quicksort and selection
- Distribution sort
- Radix sort
- Complexity of various algorithms, when to use them?

Random numbers

- Possible uses
- Idea of a pseudo-random sequence
- Linear congruential and additive generators
- Changing distributions:
  - Changing the range
  - Non-uniform distributions
- Shuffling, random selection

Graph structures

- Definition
- Uses: things represented by graphs
- Graph traversal: the generic traversal template
- Depth-first traversal, breadth-first traversal
- Topological sort
- Shortest paths
- Minimal spanning trees, union-find structures
- Memory management as a graph problem.

Debugging

- What debuggers can do
- How to use to pin down bugs
- Details of some debugger (Eclipse, gjdb, various Windows/Sun products).
- Unit testing: what it means, how to use it.
- JUnit mechanics.

Version Control

- What's it for?
- Basic concepts behind our particular system:
  - Working copy vs. repository copy
  - Committing changes
  - Updating and merging changes.
  - Tagging

A Case Study

- Presented Git version-control system as an example of a design using several ideas from this course.
- Graph (DAG) and tree structures represented with files as vertices and strings (file names), rather than machine addresses, as pointers.
- Use of hashing to create unique (or very, very likely to be unique) names: *probabilistic data structure*.
- Compression uses various kinds of map to facilitate conversion to and from compressed form, including arrays, tries, and hash tables
- Priority queue in Huffman coding.

Assorted Side Trips

- Compression.
- Parallel processing.
- Storage management and garbage collection.

What's After the Lower Division?

- CS160: User Interface Design (Hartmann)
- CS161: Computer Security (Popa)
- CS162: Operating Systems and System Programming (Joseph, Ragan-Kelley)
- CS164: Programming Languages and Compilers (Hilfinger)
- CS170: Efficient Algorithms and Intractable Problems (Chiesa, Vazirani)
- CS174: Combinatorics and Discrete Probability (Friedman)
- CS184: Graphics (Ng)
- CS186: Databases
- CS188: Artificial Intelligence (Dragan, Levine)
- CS189: Machine Learning
- CS194: Assorted Special Topics: Computational Design and Fabrication, Designing, Vi-

sualizing and Understanding Deep Neural Networks.

What's After the Lower Division? (II)

- CS152: Computer Architecture (Asanovic)

the Internet

Engineering

tional Biology

d PCB Design

- Numerous graduate courses: including advanced versions of 152, 160, 161 170, 184, 186, 189; plus Cryptography, VLSI design and many special topics.
- And, of course, EE courses!
- Various opportunities for participating in research and independent study (199)

What's After the Lower Division? (III)

- But EE and CS are just two of over 150 subjects!
- Internships offer more specific skills and exposure to real problems.
- Above all, I think that CS is a creative activity that (to the true artists) ought to fun!