CSE 373: Data Structures and Algorithms
Course Information and Syllabus
Winter 2007

Logistics and Contact Information: See the course homepage for information about the course schedule, staff, office hours, mailing lists, discussion boards, etc.: `www.cs.washington.edu/373`.

Staff: Instructor: Hal Perkins, CSE 548, perkins at cs.washington.edu.
Teaching assistant: Tian Sang, sang at cs.washington.edu.

Overview and Goals: Achieve an understanding of fundamental data structures and algorithms and the tradeoffs between different implementations of these abstractions. Theoretical analysis, implementation, and application. Lists, stacks, queues, heaps, dictionaries, maps, hashing, trees and balanced trees, sets, and graphs. Searching and sorting algorithms. Java's collections framework as an example implementation of fundamental data structures and algorithms.

Prerequisite: CSE 143

Text: Weiss, *Data Structures and Algorithm Analysis in Java*, 2nd ed, Addison-Wesley, 2007.

Assignments: There will be a mix of shorter, written assignments and longer programming problems. Assignments will normally be due Thursday evenings and will be submitted electronically via the web. Exact dates and deadlines will be specified on each assignment.

Exams: Two midterm exams and a final, which is scheduled for Tuesday, March 13 from 2:30-4:20 p.m. No makeup exams will be offered; you should plan to attend the exams when the are given. Exams will normally be closed-book, closed-notes, and no calculator will be needed.

Late Policy: Deadlines for homework and scheduled times for exams are strict — no late assignments or exams will be accepted. However, if unusual circumstances that are truly beyond your control prevent you from submitting an assignment or attending an exam on time, you should discuss this with the instructor, preferably in advance. (Even if you're sick in bed at home, you should still be able to make a phonecall or send an email message.)

Grading:

Midterm 1	15%
Midterm 2	15%
Final	20%
Homework	50%

These percentages may be adjusted if necessary to ensure fair assignment of course grades. Assignments will be weighted individually depending on their degree of difficulty.

Academic Integrity: You are to complete assignments individually. You may discuss the assignment in general terms, but the code you write must be your own. You are encouraged to discuss ideas, approaches, concepts, bugs, etc., in English, but you may not show or give your code to anyone except this course's TAs and instructor. You are not allowed to write code with another student on an assignment or to show another student your solution to an assignment.

Communications: The course message board is a good medium for discussing the course, getting help on assignments, and staying in touch outside of class hours. You can also email the instructor or TAs or go to office hours.

Computing Resources: We will use Java 5 (aka Java 1.5) for programming assignments; Java 6 also works fine. There are many good text editors and development environemnts available for Java, including Eclipse, DrJava, and Textpad (windows only). The Math Sciences Computing Center is the designated lab for this course; they have the above software installed, but the software should also be available in public campus labs.

All of the software, as well as many other Java tools and environments, is freely available on the web and we generally don't care what you use for your programming projects. Exception: you *may not* rely on or use "wizards" or other program generation tools in your programming environment to create code for your assignments. The code in your assignments should be written by you, and you should understand and be able to explain anything in it. However you create it, your code should compile and run properly using the standard Sun Java 5 software.

Preliminary topic list

1. Course administration
2. Review: data structure concepts, arrays, simple linked lists, different implementations of lists, stacks and queues, binary trees
3. Introduction to complexity: $O()$-notation
4. Sets, including union/find algorithms
5. Recursion and backtracking
6. Sorting and searching
7. Balanced trees
8. Heaps, priority queues, heapsort
9. Dictionaries/maps, hashing
10. Graphs

Topics won't be presented strictly in this order, since some of the themes like complexity and recursion will reappear as we discuss their application to different data structures throughout the course.