CMPS 101
Algorithms and Abstract Data Types
Winter 2009

Description: Studies basic algorithms and their relationships to common abstract data types. Covers the notions of abstract data types and the distinction between an abstract data type and an implementation of that data type. The complexity analysis of common algorithms using asymptotic (big O) notation is emphasized. Topics include sorting and searching techniques, basic graph algorithms, and algorithm design techniques. Abstract data types covered include priority queues, dictionaries, disjoint sets, heaps, balanced trees, and hashing. Familiarity with C, Java, and Unix is assumed.

Prerequisites: CMPS 12B or 13H; and CMPE 16 or 16H; and MATH 19B; and one of either MATH 21, 22, 23A, 24 or AMS 27.

Time and Place: MWF 12:30am – 1:40pm Jack Baskin Engineering 152
Class Webpage: http://www.soe.ucsc.edu/classes/cmps101/Winter09/
Class Webforum: http://forums.soe.ucsc.edu/

Instructor: Patrick Tantalo (http://www.cse.ucsc.edu/~ptantalo/)
Office: E2 257
Office Hours: MT 10:00-11:00am, WTh 2:00-4:00pm, or by appointment
Email: ptantalo@soe.ucsc.edu
Phone: 831-459-3898

Teaching Assistant: Radhakrishna Vuppala <vrk@soe.ucsc.edu>

Required Text: *Introduction to Algorithms,* second edition, by Cormen, Leiserson, Rivest, & Stein. McGraw-Hill, 2001. The following reading schedule is a rough guide to what we will discuss and when. It is subject to change. I expect that the material from appendices A.1-A.2, B.1-B.3, and C.1 is already familiar.

Week	*Sections*	*Topics*
1	1.1-1.2, handouts	Introduction, ADTs,
2	2.1-2.3, 3.1-3.2, handouts	Analysis of Algorithms, Asymptotic Growth Rates
3	4.1-4.3	Induction Proofs, Recurrences
4	B4, 22.1, 22.5, B.5	Graphs, Graph Algorithms, BFS
5	22.5, B.5	DFS, Trees
6	6.1-6.5, 21.1-21.3	Heaps, Priority Queues
7	21.1-21.3	Disjoint Sets
8	23.1-23.2, 24.1, 24.3, 24.5	Minimum Weight Spanning Trees, SSSP Problem
9	12.1-12.3, 13.1-13.4	Binary Search Trees, Red-Black Trees
10	7.1-7.4, 8.1-8.4	Sorting Algorithms

Coursework and Evaluation:
Homework will consist of written assignments taken from the exercises at the end of each section, and from the end of each chapter. Homework will be graded only as to it's completion, not it's correctness. Specifically, one point will be awarded for each problem (or each part of a multi-part problem) which is seriously attempted. The main purpose of the homework is to prepare for the exams. We will have five **Programming Assignments**, due at intervals of roughly 2 weeks. The **first Midterm Exam** will be held **Friday February 6**, and the **second Midterm Exam** will be held **Friday March 6**. The **Final Exam**

will be held on **Thursday March 19**, **4:00 pm – 7:00 pm.** Please make arrangements to be available at the appropriate times. Coursework will be weighted as follows:

Homework	5%
Programming Assignments	35%
Midterm Exam 1	15%
Midterm Exam 2	15%
Final Exam	30%

The grading scale for the class will be approximately: A+::97%-100%, A::93%-96%, A-::90%-92%, B+::87%-89%, B::83%-86%, B-::80%-82%, C+::76%-79%, C::70%-75%, D::60%-69%, F::0%-59%. Letter grade boundaries may be lowered at my discretion in order to eliminate some borderline cases.

Academic Honesty:
The Baskin School of Engineering has a zero tolerance policy towards any incident of academic dishonesty. If cheating occurs, consequences within the context of the course may range from getting zero on a particular assignment, to failing the course. In addition to these sanctions, every case of academic dishonesty is referred to the students' college Provost, who sets in motion an official disciplinary process. Cheating in any part of the course may lead to failing the course and suspension or dismissal from the university.

What is cheating? In short, it is presenting someone else's work as your own. Examples include (but are not limited to) copying another student's written homework assignment, or program, allowing your own work to be copied, or in any way facilitating the cheating of others. Although you may discuss problems with fellow students, your collaboration must be at the level of *ideas* only. Legitimate collaboration ends when you "lend", "borrow", or "trade" *written solutions* to problems, or in any way share in the act of *writing* your answers. You may freely give and receive help with the computer facilities, editors, the UNIX operating system, and the proper use and syntax of the C and Java programming languages; but you may not copy, paste, email, or in any way share *source code*. If you do collaborate (legitimately) or receive any form of help from anyone, you must credit them by placing their name(s) at the top of your paper, or in the case of programming assignments, in your README file.

Please go to http://www.ucsc.edu/academics/academic_integrity/ to see the full text of the University's policy on Academic Integrity.