CMPS 101
Algorithms and Abstract Data Types
Winter 2015

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:
Computer Science 12B or 13H, Computer Engineering 16 or 16H, Mathematics 19B, 20B or 11B, and one course from the following: Mathematics 21, 22, 23A, or Applied Mathematics and Statistics 10.

Time and Place: MW 5:00 – 6:45 pm Oakes 105
Class Webpage: http://people.ucsc.edu/~ptantalo/cmps101/Winter15/

Instructor: Patrick Tantalo http://users.soe.ucsc.edu/~ptantalo/
Office: E2 257
Office Hours: TTh 10:00 am - 1:00 pm, or by appointment
Email: ptantalo@soe.ucsc.edu
Phone: 831-459-3898

Teaching Assistants:
Larissa Munishkina (mlarissa@soe.ucsc.edu)
Subhag Ragi (sragi@ucsc.edu)

Course Tutors:
Alexey Munishkin (amunishk@ucsc.edu)
Ryan Connors (rmconnor@ucsc.edu)

MSI Tutor:
Andrew Ringer (ajringer@ucsc.edu)

Required Text: *Introduction to* Algorithms (2nd or 3rd edition) by Cormen, Leiserson, Rivest and Stein. McGraw-Hill 2001 (ISBN 9780262033848). The following reading schedule is a rough guide to what we will discuss and when. Section numbers are from the 3nd edition. I expect that the material from appendices A.1-A.2, B.1-B.3, and C.1-C.2 is already familiar.

Week	*Sections*	*Topics*
1	1.1-1.2, handouts	ADTs, Analysis of Algorithms
2	2.1-2.3, 3.1-3.2, handouts	Asymptotic Growth Rates
3	4.3-4.5, handouts	Induction Proofs, Recurrences
4	B4, B.5 handouts	Graphs, Trees
5	22,1-22,5	Graph Representations, BFS, DFS
6	6.1-6.5	Heaps, Heapsort, Priority Queues
7	21.1-21.3, 23.1-23.2	Disjoint Sets, Minimum Weight Spanning Trees
8	24.1, 24.3	SSSP Problem, Bellman-Ford and Dijkstra's Algorithms
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 in the Cormen text
- **Programming Assignments:** Due at roughly 10 day intervals
- **Midterm Exam 1:** Monday February 2
- **Midterm Exam 2:** Monday March 2
- **Final Exam:** Thursday March 19, 7:30-10:30 pm

Coursework will be weighted as follows:

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

Grading scale:

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.

Accommodations for Students with Disabilities:

Any student who believes s/he needs an accommodation, based on the impact of a disability, should contact the Disability Resource Center (DRC) at 831-459-2089 in room 125 Hahn Student Services or by email at drc@ucsc.edu to coordinate those accommodations. If you qualify for classroom accommodations, please contact me privately to submit your Accommodation Authorization and to discuss specific needs, preferably within the first two weeks of the quarter. See the DRC webpage http://drc.ucsc.edu/ for more information.

Academic Honesty:

The Baskin School of Engineering has a zero tolerance policy for any incident of academic dishonesty. If cheating occurs, consequences may range from getting zero on a particular assignment to failing the course. In addition 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, suspension or dismissal from the Baskin School of Engineering, or from UCSC.

What is cheating? In short, it is presenting someone else's work as your own. Examples would include copying another student's programming assignment, or allowing your own work to be copied. You may discuss projects with fellow students, but your collaboration must be at the level of *ideas* only. Legitimate collaboration ends when you in *any way* share in the act of *writing* solutions. 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*, *transfer* or in any way share *source code*. Please go to http://www.ucsc.edu/academics/academic_integrity/ to see the full text of the University's policy on Academic Integrity.