CMPS 101
Algorithms and Abstract Data Types
Spring 2018

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 course from the following: MATH 21, 22, 23A, 24 or AMS 27.

Time and Place: TTh 3:20pm-4:55pm Thimann Lecture 003
Class Webpage: https://classes.soe.ucsc.edu/cmps101/Spring18/

Instructor: Patrick Tantalo http://users.soe.ucsc.edu/~ptantalo/
Office: E2 255
Office Hours: TTh 5:30pm-7:00pm & W 10:00am-1:00pm, or by appointment
Email: ptantalo@soe.ucsc.edu

Teaching Assistants:
Isaak Cherdak (icherdak@ucsc.edu)
Naishil Shah (nanshah@ucsc.edu)
Ruben Fitch (rubensamf@soe.ucsc.edu)

Course Tutors: TBA

MSI Tutors:
Elisabeth Oliver (elaolive@ucsc.edu)

LSS Tutor:
Art Parkeenvincha (tparkeen@ucsc.edu)

Required Text: *Introduction to* Algorithms (2nd or 3rd edition) by Cormen, Leiserson, Rivest and Stein (CLRS). 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:

- 5% Homework: Written exercises, mostly from CLRS, submitted through CrowdGrader.
- 25% Programming Assignments: Due at roughly 10 day intervals.
- 20% Midterm Exam 1: Thursday April 26.
- 20% Midterm Exam 2: Thursday May 24.
- 30% Final Exam: Tuesday June 12, 8:00 – 10:00 am.

Grading scale:

A+	97.0% - 100%
A	93.0% - 96.9%
A-	90.0% - 92.9%
B+	87.0% - 89.9%
B	83.0% - 86.9%
B-	80.0% - 82.9%
C+	76.0% - 79.9%
C	70.0% - 75.9%
C-	67.0% - 69.9%
D+	64.0% - 66.9%
D	61.0% - 63.9%
D-	58.0% - 60.9%
F	0% - 57.9%

Letter grade boundaries *may be* lowered at my discretion in order to eliminate some borderline cases.

Accommodations for Students with Disabilities

UC Santa Cruz is committed to creating an academic environment that supports its diverse student body. If you are a student with a disability who requires accommodations to achieve equal access in this course, please submit your Accommodation Authorization Letter from the Disability Resource Center (DRC) to me privately during my office hours or by appointment, preferably within the first two weeks of the quarter. At this time, I would also like us to discuss ways we can ensure your full participation in the course. I encourage all students who may benefit from learning more about DRC services to contact DRC by phone at 831-459-2089, or by email at drc@ucsc.edu.

Academic Honesty:

The Baskin School of Engineering has a zero tolerance policy for any incident of academic misconduct. If cheating occurs, consequences may range from getting zero on a particular assignment to failing the course. In addition every case of academic misconduct 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 include copying another students' lab or programming assignment, allowing your own work to be copied or in any way facilitating misconduct by others. You may discuss programming and lab projects with fellow students, but your collaboration must be at the level of *ideas* only. You may freely give and receive help on the UCSC computer facilities, code editors and IDEs, the UNIX operating system, and on the proper use and syntax of the Java and C programming languages. You may also freely use any *example code* that is posted by me on this quarter's web page. However, you may not *copy*, *paste*, *email*, *transfer* or *share* in any way the *source code* for projects in this class. Go to https://www.ue.ucsc.edu/academic_misconduct to see the University's official policy on Academic Misconduct.