ECS 60 Course Syllabus Summer 2015

Name	Title	E-mail	Room	Office Hours
Sean Davis	Lecturer	ssdavis@ucdavis.edu	3052 Kemper	M 10-11:15, 1-2; W 10-12, 1-2; R 8-10:30; and by appointment
Yu-Cheng Lin	Reader	ycjlin@ucdavis.edu		None

Web page: http://csiflabs.cs.ucdavis.edu/~ssdavis/60/homepage.html

Newsgroup: https://piazza.com/ucdavis/summer2015/ecs60/home

E-mail to Sean should only be regarding personal matters and not questions about assignments or tests, and must come from an ucdavis.edu e-mail account. All course questions should be posted to the piazza newsgroup.

Required Materials: Weiss, Mark Allen, *Data Structures and Algorithm Analysis in C++*, 4th ed., Berkeley, CA: Addison Wesley, 2014.

Prerequisites: ECS 20 and ECS 40 or equivalents with grade of C- or better.

Course objectives:

1. Students will be able to apply algorithm analysis to the operations on data structures and interpret the results.
2. Students will be able to understand and evaluate the operations and possible implementations of the following abstract data types: lists, stacks, queues, binary trees, AVL trees, splay trees, tries, B-trees, hash tables, and heaps.
3. Students will understand the operation and characteristics of the following sorting algorithms: insertion sort, shellsort, heapsort, mergesort, quicksort, bucket sort, radix sort, and external sorting.
4. Students will understand the operation and application of graph algorithms including depth first search, breadth first search, shortest path, minimum spanning tree, network flow, and topological sort.
5. Students will be able choose (and justify) appropriate data structures and algorithms for complex programming tasks.

Approximate Course Grading:

Homework	15%
Programs	25%
Two midterms	30%
Final	30%
Class effort/participation	5% (extra credit)

Letter grades will be approximately: A = 90+% ; B = 80-89%; C = 70-79%; D = 60-69%; F <60%

Work Input/Output

- Written Homework: All homework must be stapled together. Each student must do his or her own work. Written homework should be submitted in the ECS 60 slot in 2131 Kemper by 4:00 PM on the date due.
- Programs: Students should work together in groups of two people to write the programs. The names of both group members must appear on the first line of each file. All students may help each other with debugging, but each group must write their own code. Program source code will be submitted using the handin facility of UNIX by midnight on the date due. Each group will submit all of its programs to the handin directory of exactly one of its members. Programs that do not compile will receive no credit. You should not look online for solutions to the assignments. Assignments will be submitted to the MOSS program at Stanford for plagiarism analysis. Students that hand in suspicious programs will be reported to Student Judicial Affairs. Each student (not each group) must write his/her own program write-ups for the timetest programs. After the return of grade slips, programs may be edited for five minutes in Sean's office, and then re-tested for operation, but not design violations.
- All work: Late work will **NOT** be accepted without a doctor's excuse. All work will be returned in lecture. All work not picked up in class will be available in my office during office hours.

Discussions: Wednesdays, Weeks 1, 2, 4: 4:10 – 5:50 in 166 Chem; Weeks 3, 5, 6: 10:00 - 11:40 in 204 Art.

Exams: Exams are cumulative, closed book, closed notes. The final will be Wednesday, July 29th, 3:50 – 5:50 in 226 Wellman.

Tentative Schedule

Dates	Subjects	Reading
6/22	Intro, math review, induction, complexity, ADTs, lists.	Chapter 1, 2, 3.1, 3.2.
6/23	Lists cont'd, cursor lists, skip lists, stacks and queues.	Chapter 10.4.2, 3.3, 3.4
6/24 2:10	General trees, B-Trees, binary trees, binary search trees, tree traversals.	Chapter 4.1 – 4.3, 4.6, 4.7
6/24 4:10	Catch-up and discussion.	None
6/29	Trees cont’d: AVL trees, splay trees, amortized analysis	Chapter 4.4, 4.5.
6/30	Amortized analysis cont'd, tries, Huffman encoding	Chapter 10.1.2, 11.1
7/1 2:10	Hashing: idea, hash functions, separate chaining, open addressing.	Chapter 5.1-5.4
7/1 4:10	Catch-up and discussion.	None
7/6	Hashing cont’d: rehashing, extendible hashing	Chapter 5.5-5.6.
7/7	Priority Queues: binary heaps, d-heaps, applications.	Chapter 6.1- 6.5
7/8 2:10	Catch-up and discussion.	None
7/8 4:10	Midterm #1, Chapters 1-5, 10.1.2, 10.4.2, 11.5	None
7/13	Disjoint Set: intro and basic data structures, smart union, path compression.	Chapter 8.1-8.5
7/14	Graph Algorithms: Definitions, topological sort, shortest path algorithms.	Chapter 9.1-9.3
7/15 2:10	Graph Algorithms cont’d: Minimum spanning tree, network flow algorithms	Chapter 9.4 – 9.5
7/15 4:10	Catch-up and discussion.	None
7/20	Graph Algorithms cont'd: Depth first search	Chapter 9.6
7/21	Sorting: insertion sort, lower bound, shellsort, heapsort, mergesort, quicksort.	Chapter 7.1-7.7
7/22 2:10	Catch-up and discussion.	None
7/22 4:10	Midterm #2, Chapters 1-6.5, 8 – 10	None
7/27	Sorting cont’d: quicksort cont'd, indirect sorting, lower bound, bucket sort, radix sort, external sorting.	Chapter 7.7 -7.12
7/28	Algorithm design	Chapter 10
7/29 2:10	Catch-up and discussion.	None
7/29 **3:50**	Final	None

Written Homework Assignments

#	Pts	Problems	**Due date** (4:00 PM)
1	62	1.7ab (must use induction for a), 1.8 abc, 1.12 (must use induction for b), 2.1, 2.2 (provide counter example for false), 2.7 a, 2.15, 2.25, 2.31, 2.33, 3.4, 3.20a, 3.25a (Hint: You’ll need some data structure besides the original stack), 3.29, 3.34a	6/30
2		Combined with #1 for the summer	
3	24	4.5, 4.27, 4.35 (Hint: it is recursive), 4.44, 10.3	7/6
4	22	5.1 (a is FILO), 5.2 (rehash to 23), 5.4, 5.27	7/8
5	12	6.1, 6.2, 6.3 (both heaps), 6.14.	7/13
6	12	8.1 (In the case of ties, the second tree is made the child of the first.), 8.2 using deepest node(s)	7/15
7	12	9.1, 9.5, 9.11, 9.15, 9.26	7/22
8	10	7.5a, 7.19, 7.24, 7.30, 7.44	7/28

Programming Assignments

#	Program Description	**Due date** (11:59pm)
1	Simple ADT Timing Tests, queens.cpp	6/29
2	BTree Delete program	7/6
3	Advanced ADT Timing Tests, quadratic probing pointer hash	7/13
4	Challenge(s) 1	7/20
5	Challenge 2	7/27