20 CS 2928: Data Structures --- Fall 2012

Time: MW 9:05-10:00 AM, Swift 720

Professor: Paul Talaga
Office: Rhodes 487, Office Hrs: Tues/Thurs 9am-noon or by appt (email)
E-mail: paul.talaga@uc.edu
Website for course: Blackboard

Catalog Data:
CS-2928 Covers Data Structures, such as stacks, queues, priority queues, trees, and graphs. Introduction to balanced trees, hashing, searching, and complexity of algorithms. Practice implementing concepts of data structures, encapsulation of structure, and behavior of data types.

Real-World Relevance:
Computing job interviews use many data structure questions to weed out applicants, so mastery is essential to securing, and performing well, in a great job. Coding proficiency comes with practice, which you’ll gain through programming projects in C++.

Prerequisites:
20-CS-122 (Computer Science II)

Textbook:
F. M. Carrano, Data Abstraction and Problem Solving with C++: Walls and Mirrors, 5th ed.

Goals:
Students should learn several standard abstract data types and their implementations. They should learn several basic algorithms common to these data types and be familiar with standard measures of algorithmic complexity. They should practice program design based upon data requirements, choosing abstract data types based upon general requirements and choosing implementations of those data types to optimize programmer productivity, machine efficiency, and/or space utilization.

Approximate Schedule:
1. Review of Linear Data Structures: Linked Lists, Stacks and Queues (Chapters 4, 6, 7)
2. Review of files & UNIX (Notes)
3. Intro to Algorithms: Computing Powers, Efficiency of Algorithms (Chapters 9)
4. Sorting (Chapter 9)
5. Binary Trees (Chapter 10)
6. Binary Search Trees (Chapter 10)
7. General Trees (Chapter 10) Prefix-Trees (notes)
8. Buffer Week / Selected Topics
9. MIDTERM TEST
10. Tables, Priority Queues and Heapsort (Chapter 11)
11. Balanced Trees (Chapter 12.1)
12. Hashing (Chapter 12.2)
13. Graphs and Digraphs (Chapter 13)
14. Graph Traversals: DFS, BFS (Chapter 13)
15. In-class group project (tentative)
16. FINAL EXAM

You are responsible for checking Blackboard for announcements, assignments, or any class changes.

**Homework:**

There will be approximately 5 computer programing projects. The programming language will be C++ and the platform Unix. You may use any Unix platform you wish, e.g., Linux, Mac, UCFileSpace, etc. However, programs must compile using the C++ compiler `g++` on the Unix shell or UCFileSpace using `make`. Projects are to be handed in hard-copy (4 pages per sheet is encouraged, duplexed) as well as submitted electronically to Blackboard (zip or tar.gz all files).

Projects will be graded upon style as well as upon correctness. See posted example project in Blackboard. Projects are expected to be finished and hard-copy handed in on time at the beginning of class. $2^{d+3}$ points will be deducted where $d$ is the number of days late from the start of class. Time stops when files are deposited in Blackboard and hardcopy is submitted within 24hrs.

Students may discuss programming assignments with each other, but may not write programs together or copy. Any significant help, from student, book, web, etc. must be acknowledged in a comment in the program. Any duplication of code will result in a grade divided by the number of students copying. The Internet is considered a student in this case. Write your own code and you’ll be fine.

**Quizes:**

5 or so quizzes may be given at the beginning of class, covering topics in a previous class or required reading. Cheating on a quiz or exam will result in a F for the course.

**Grading:**

Projects 40%,
Quizes: 10%
Tests: Midterm 25%, Final Exam 25%