Computer Engineering 70
Formal Specification and Advanced Data Structures

Winter 2010
Mondays, Wednesdays, and Fridays
10:30 am – 11:35 am

Instructor

Instructor:	Darren Atkinson
E-mail:	datkinson@scu.edu
Office hours:	Mondays 2:15–3:15 pm and Wednesdays 9:30-10:30 am
Office:	EC 245
Website:	http://www.cse.scu.edu/~atkinson/teaching/wi10/070/

Teaching Assistant

Teaching assistant:	Nicolas Treat
E-mail:	ntreat@scu.edu
Lab hours:	Wednesdays and Thursdays, 2:15–5:00 pm

Textbooks

Required: Ullman, *Elements of ML Programming, ML97 Edition*, 1998

Grading

Lab assignments:	20%
Midterm exams:	40% (1/27 and 2/17)
Final exam:	40% (3/15)

Overview

Abstract data types:	lists (including stacks, queues, dequeues, and priority queues), sets, multisets (bags), maps (dictionaries), and graphs
Data structures:	lists (linked-lists), binary trees (including binary search trees and AVL trees), 2-3 trees, heaps (including binary heaps, binomial heaps, and fibonacci heaps)
Computer science theory:	algebraic specification, axiomatic semantics, structural induction, normal forms

Course Objectives

Students will be able . . .

1. To understand the specification, representation, implementation, and use of abstract data types.
2. To understand structural induction as a generalization of mathematical induction.
3. To develop skills for working on teams.
4. To develop facility with a different style of programming than that to which they are accustomed.

Learning Outcomes

Students will . . .

1. Use a formal specification language to define ADTs.
2. Compare and contrast the tasks of specification, representation, and implementation of ADTs.
3. Use both representation dependent and representation independent programming appropriately.
4. Develop programs that implement and/or use ADTs in a language representing a programming paradigm that is different from that learned in the introductory programming sequence.
5. Given a scenario/problem: Identify/invent an appropriate ADT to use in the solution; Define and evaluate multiple options for representation of the chosen ADT; Define and evaluate multiple options for implementation of the chosen representation.
6. Use structural induction as the basis for: Inductive definition of domains; Recursive definition of functions/relations over inductively defined domains; Proofs of properties of recursive functions and elements of inductively defined domains.
7. Evaluate teamwork of three different lab teams during the quarter, based on a rubric provided by the instructor.

Policies

Disability Accommodation Policy

To request academic accommodations for a disability, students must contact Disabilities Resources located on the second floor of Benson. Phone numbers are (408) 554-4111; TTY (408) 554-5445. Students must register and provide documentation of a disability to Disabilities Resources prior to receiving academic accommodations.

Academic Integrity Policy

The University is committed to academic excellence and integrity. Students are expected to do their own work and to cite any sources they use. A student who is guilty of a dishonest act in an examination, paper, or other work required for a course, or who assists others in such an act, may, at the discretion of the instructor, receive a grade of F for the course.

In addition, a student found guilty of a dishonest act may be subject to sanctions up to and including dismissal from the University as a result of the student judicial process as described in the *Community Handbook*.

A student who violates copyright laws, including those covering the copying of software programs, or who knowingly alters official academic records from this or any other institution is subject to similar disciplinary action.