

CS 4521: ALGORITHMS & DATA STRUCTURES

Catalog Course Description

Asymptotic analysis of algorithms. Methods for proving correctness. Implementation of algorithms. Survey of algorithms and data structures, such as: heaps and heapsort, quicksort, binary search trees, red-black trees, B-trees, hash tables, graph algorithms, dynamic programming, and greedy algorithms.

Prerequisites: CS 2511, Math 3355

Course Outcomes

1. Further develop ability to understand, apply and prove precise mathematical claims about space and time requirements of algorithms, and about correctness of algorithms.
 - a. Solidify understanding of definitions and notation for asymptotic notation, including the ability to prove claims involving asymptotic notation.
 - b. Improve intuitions about the practical significance of precise claims involving asymptotic notation.
 - c. Solidify ability to understand proofs of algorithm correctness, including the use of loop invariant arguments (as a variation on the theme of inductive proofs).
2. Detailed knowledge of a number of standard algorithms, and data structures, including knowledge of time and space complexity, ability to prove (or understand proofs) of properties of data structures and associated algorithms, and also the ability to implement such algorithms.
 - a. In-depth knowledge of several of the standard sorting algorithms and associated data structures.
 - b. Knowledge of a number of more advanced algorithms and data structures, such as red-black trees, B-trees, binomial heaps, Fibonacci heaps.

Relationship to Program Outcomes

CS 4521, an elective course, requires discrete math and software analysis and design as prerequisites. This course contributes to meeting the following program outcomes:

a. Students understand the mathematics and statistics that underlie scientific applications.

Students comprehend the definitions used in standard asymptotic notation, and also comprehend the use of this notation to express time complexity properties of algorithms. Students comprehend and in some cases prove mathematical properties of data structures. Students understand proofs of correctness for some algorithms, including the use of loop invariants.

b. Students can design, develop and analyze significant software systems.

Students gain additional experience in software design and implementation by implementing various algorithms and data structures. Ability of students to design, develop and analyze significant software systems is enhanced by knowledge of algorithms and data structures, including their space and time complexity properties.

c. Students understand the fundamentals of computer organization and architecture, data structures and related algorithms, and programming languages.

This course substantially increases a student's knowledge of data structures and algorithms.

d. Students can apply computer science principles and practices to a variety of problems.

Students work with a variety of new data structures and algorithms. They also gain a deeper understanding of asymptotic notation, and of proofs of correctness, including the use of loop invariants, and see a range of applications of these concepts.

e. Students can work independently and also work effectively in teams.

Students gain further experience working independently.