

## COURSE DESCRIPTION

Dept., Number	CS 242	Course Title	Data Structures and Algorithms II
Semester hours	4	Course Coordinators	Benjamin, Courtney, Gargano

### 2004-2006 Catalog Description

Study of the design and analysis of sorting, searching, matrix multiplication, and other important algorithms with emphasis on structure, complexity, and efficiency. Topics chosen from logic, graph theory, and the theory of functions. Data structures such as balanced binary trees, AVL trees, B-trees.

### New Description

Algorithms in graph theory including the topological sort and critical path analysis, finding minimal weight spanning trees, and shortest paths. An assortment of sorting algorithms with examination of their structural properties (time complexity, space complexity, natural versus unnatural behavior, stability versus instability, adaptable for externally stored files) Algorithmic strategies (greedy, divide and conquer, dynamic programming, branch-and-bound), proofs of correctness, NP hard and NP-complete problems. Topics chosen from logic, string pattern matching, and other areas.

May 2006

**Textbook:** Faculty may choose the latest edition of either of the following:

Robert Sedgwick; *Algorithms in Java Parts 1-5, Fundamentals, Data Structures, Sorting, Searching, and Graph Algorithms*; Addison Wesley.

Mark Allen Weiss; *Data Structures & Problem Solving Using Java*; Addison Wesley

### References (latest edition is preferred)

Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman; *Data Structures and Algorithms*; Addison-Wesley

Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman; *The Design and Analysis of Computer Algorithms*; Addison Wesley

Sara Baase and Allen Van Gelder; *Computer Algorithms: Introduction to Design and Analysis*; Addison Wesley

Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest;  
*Introduction to Algorithms*; MIT Press and McGraw Hill

Michael R. Garey and David S. Johnson; *Computers and Intractability:  
A Guide to the Theory of NP-Completeness*; W. H. Freeman

Michael T. Goodrich and Roberto Tamassia; *Foundations, Analysis, and  
Internet Examples*; Wiley

Ellis Horowitz and Sartaj Sahni; *Fundamentals of Computer Algorithms*;  
Computer Science Press.

Ellis Horowitz and Sartaj Sahni; *Fundamentals of Data Structures*;  
Computer Science Press

Donald E. Knuth; *The Art of Computer Programming - Volumes 1 and 3*;  
Addison Wesley

Sartaj Sahni; *Data Structures, Algorithms, and Applications in C++*; McGraw Hill

Mark Allen Weiss; *Data Structures and Algorithm Analysis Using Java*;  
Benjamin Cummings

Mark Allen Weiss; *Data Structures & Problem Solving Using Java*. Addison Wesley

Niklaus Wirth; *Algorithms and Data Structures*; Prentice Hall

## **Course Goals**

**Objective 1:** Students will learn the data structures for representing a graph.

Outcomes: Students will demonstrate the ability to:

- ◆ Explain when a topological sort may be successfully completed
- ◆ Explain the use of the topological sort
- ◆ Diagram the action of a topological sort
- ◆ Code a topological sort or some other algorithm with an adjacency matrix
- ◆ Code a topological sort or some other algorithm with an adjacency list

**Objective 2:** Students will learn and implement algorithms in graph theory.

Outcomes: Students will demonstrate the ability to:

- ♦ Solve minimum weight spanning tree problems and write the code for computing the minimum weight spanning tree for a connected, nondirected graph using both Kruskal's algorithm and Prim's algorithm
- ♦ Explain why Kruskal's algorithm and Prim's algorithm are considered "greedy"
- ♦ Find the shortest path between two vertices in a weighted, directed graph using Dijkstra's algorithm and write the corresponding code.
- ♦ Explain why Dijkstra's algorithm exemplifies dynamic programming.
- ♦ Diagram and code depth-first and breadth-first traversals of nondirected graphs
- ♦ Diagram and code a graph's exhaustive search tree.
- ♦ Define an Euler circuit and state the necessary and sufficient conditions for one to exist within a digraph or graph.
- ♦ Define a Hamiltonian circuit and explain that the traveling salesperson's problem is that of finding the minimal cost tour (i.e. the shortest Hamiltonian circuit)
- ♦ Explain and diagram the differences among the exhaustive search, the backtracking, and the branch-and-bound approaches to solving the traveling salesperson problem

**Objective 3:** Students will be able to explain and exemplify the designations of P, NP, NP-complete in the classification of problems.

Outcomes: Students will demonstrate the ability to:

- ♦ explain the designation "polynomial" and exemplify it
- ♦ explain the designation "nondeterministic polynomial" and exemplify it
- ♦ explain the designation "NP-complete" relative to the satisfiability problem and exemplify it with the partition problem, the bin packing problem, and/or others.

**Objective 4:** Students will learn an assortment of sorting algorithms; and, from these, that different algorithms have properties making them appropriate for different applications.

Outcomes: Students will demonstrate the ability to:

- ♦ to diagram the mechanisms of the following sorting techniques: the selection, bubble, and insertion sorts; the quick sort and the heap sort; a family of merge sorts; and the radix sort
  
- ♦ compare and contrast the above algorithms based on their formal characteristics; for instance: the bubble sort is natural; the selection sort can be implemented so that it is stable; the radix sort, which does not operate by way of comparing keys, has a time complexity  $O(N)$ ; the merge sort can be adapted to work with externally stored files; the heap sort uses a tree stored in an array; the quicksort exemplifies the approach of divide and conquer

**Objective 5:** Student will teach themselves an algorithm, write an explanation of it, and explain it to the class. This will afford each student an experience with independent learning and with the written and oral communication of technical content to technical colleagues. (Algorithms assigned for independent study and presentation include the Boyer-Moore algorithm for string pattern matching; Huffman compression; Strassen's method for matrix multiplication; determining 2-reachability, 3-reachability, ..., n-reachability in digraphs; determining whether a graph is biconnected; and the like.)

Outcomes: Students will demonstrate the ability to:

- ♦ learn technical material independently
  
- ♦ create a piece of technical writing
  
- ♦ make a technical oral presentation

**Objective 6:** Students will be able to perform a proof of program correctness.

Outcomes: Students will demonstrate the ability to:

- ♦ explain the aim of program verification
- ♦ explain the critiques of program verification
- ♦ use the technique of program verification (establishing precondition assertions, postcondition assertions, and loop invariants) to prove that a method sets **max** to the largest value in an array and **index** to the subscript where it may be found, to prove that a method traverses a linked list from head to rear, to prove an implementation of Knuth's "right-to-left binary method for exponentiation," or something similar

### **Prerequisites by Topic**

- ♦ ability to use stacks, queues, fifo queues in problem-solving (e.g, the evaluation of postfix expressions)
- ♦ ability to explain the concept of time complexity analysis and explain why a linear search through an unsorted array has a worst case performance of  $O(N)$ , and why a height-balanced binary search tree has a worst case performance of  $O(\lg N)$ ;
- ♦ ability to describe and contrast the properties of some information storage/retrieval systems (e.g. unsorted array, binary search, binary search tree, hashing) and to implement these

## Major Topics Covered in the Course

- ♦ Graphs and digraphs -- methods for representation; strong components, minimum weight spanning trees, shortest path trees, breadth-first and depth-first spanning trees; the topological sort, Euler circuits and Hamiltonian circuits
- ♦ Sorting -- extending familiarity with sorting to include the heap sort and the quicksort, the radix sort (sorting by distribution as opposed to key comparison), the family of merge sorts (for both internal and external sorting)
- ♦ Other topics selected in accordance with interest and time, such as data compression (Huffman's algorithm), cryptology, string searching (e.g. the Boyer-Moore algorithm), Strassen's method for matrix multiplication, the generation of pseudo-random numbers, additional graph algorithms
- ♦ NP-completeness
- ♦ Program verification