EC330 Applied Algorithms for Engineers

2008-2009 Catalog Data:
Prereq: ENG EC 327 and CAS MA 193. Introduction to the general concept of algorithms. Efficiency and run-time of algorithms. Various approaches to design of algorithms and their applications to numerous typical numerical and non-numerical problems. 4 cr.

Class/Lab Schedule:
LEC: 4 hrs/wk (TTh10-12), DIS: 1 hr/wk (W5-6)

Status in the Curriculum: Required

Textbooks and other required materials:
Cormen, Leiserson, Rivest, and Stein, Introduction to Algorithms (Second Edition), MIT press, 2001.

Reference:
* Mark Allen Weiss, Data Structures & Algorithm Analysis in C++ (Second Edition), Addison-Wesley, 1999; Lender Ammeraal, C++ for programmers (3rd edition), John Wiley & Sons; Dietel & Deitel, C++-How to Program (5th edition) Prentice Hall, 2005.

Coordinator:
Ari Trachtenberg, Professor, Electrical and Computer Engineering

Prerequisites by topic:
CAS EC 327: Introduction to Software Engineering/CAS MA 193: Discrete Mathematics

Goals:
To provide students with:
- Understanding and analysis of fundamental computer algorithms for searching, sorting and graph problems
- Extensive experience in the design and implementation of algorithms in C++ in linux and windows development environments.
- Exposure to networking, databases, graphics, and other common system tools.
- Exposure to the use of algorithms and data structures in selected applications

Course Outcomes:
As an outcome of completing this course, students should be able to:
1) *Analyze* the complexity of algorithms.
2) *Understand* common mathematical analysis tools, including techniques for solving recurrences.
3) *Understand* common algorithm types, such as divide-and-conquer, dynamic programming, and greedy algorithms.

4) *Understand* the use of asymptotic efficiency notation.
5) *Understand* the different algorithms for comparitive and non-comparative sorting, together with information-theoretic lower bounds.
6) *Understand* string matching, hashing, and median statistics.
7) *Understand* basic graph theory and representations.
8) *Understand* spanning trees, search trees, and their underlying data structures.
9) *Understand* graph traversal and shortest paths calculations.
10) *Understand* the computational advantages of different algorithms for searching, sorting and graph problems.
11) *Understand* different representations of sparse data structures for graph and matrix computations.
12) *Understand* the use of advanced data structures such as balanced binary trees, B and B+ trees in searching and indexing.
13) *Understand* more specialized data structures such as tries, priority queues, and disjoint-set structures.
14) *Apply* known algorithms for efficient solution of common graph problems to English-language problem descriptions.
15) *Design* solutions to common English-language computation problems.
16) *Develop* team-building skills through large-scale team projects.
17) *Develop* engineering refinement skills through on-line class competitions.
18) *Refine* skills in object oriented programming using C++.
19) *Discover* different applications of efficient algorithms in communication networks, coding theory, string parsing and other fields.
20) *Write* documentation and reports for projects on the course wiki.
21) *Experience* coding in development environments on windows and linux operating systems.

Course Outcomes mapped to Program Outcomes:

Program:	**a**	**b**	c	d	e	f	g	h	i	j	k
Course:	1-14	2, 17	14-17	16, 20	15-17	20	16,20	15,19	19	15,21	14-18,21
Emphasis:	5	2	5	4	5	2	3	2	1	2	3

1=not at all; 5=a great deal;

Topics in Project Assignments:
Mathematical analysis, sorting, disk defragmentation, caching, Sudoku, Tic Tac Toe, Traveling Salesperson Problem, Shortest paths and optimization, Currency arbitrage

Contribution of Course to Meeting the Professional Component:
Engineering topics: 100%

Prepared by: Professor Ari Trachtenberg **Date:** June 12, 2009