

# CMSC 451: Design and Analysis of Algorithms

Fall 2005

<http://www.cs.umd.edu/class/fall2005/cmsc451/>

**Instructor:** Samir Khuller - Office: AVW 3369. Office phone: (301) 405-6765.

E-mail: samir@cs.umd.edu.

Office hours: Monday 3:00pm - 4:00pm and Thursday: 2:00pm - 3:00pm.

**Class Time:** TuTh 9:30 - 10:45, Room: CSI 3120

**Teaching Assistant:** Maryam Farboodi. Office hours will be held in 1112 AVW.

Email: farboodi@cs.umd.edu.

Office Hours: Will be posted on the class page and by appointment.

The BEST way to contact the TA is to send her email.

**Course Overview:** This course presents the fundamental techniques for designing efficient computer algorithms, proving their correctness, and analyzing their complexity. General topics include graph algorithms, and basic algorithm design paradigms (such as divide-and-conquer, dynamic programming and greedy algorithms), lower bounds and NP-completeness.

**Texts:** The primary book we follow will be *Algorithm Design* by J. Kleinberg and E. Tardos. ISBN 0-321-29535-8. Published by Addison Wesley (2005).

Another book that you may be familiar with is the one by Thomas Cormen, Charles Leiserson, Ron Rivest, *Introduction to Algorithms*, McGraw Hill and MIT Press, 1990. There is a second edition out now (2001). Either of these books has comprehensive coverage of the material, and it will not hurt if you try to solve some of the problems from this book

**Prerequisites:** CMSC 112, CMSC 150/250, and CMSC 351. Each student is expected to know the basic concepts of programming (e.g. loops, pointers, recursion), discrete mathematics (proof by induction, sets), simple data structures (lists, stacks, queues, trees, heaps), and calculus (logarithms, differentiation, integration). I will assume knowledge of algorithm analyses techniques (material normally covered in CMSC 351).

**Course Work:** Course work will consist of 6-8 homework assignments, a quiz and two exams (one midterm and a comprehensive final). Homework problems will be mathematically oriented.

The quiz will be in class on October 6. The midterm will be on Oct 25. Homeworks are to be turned in at the start of class on the due date. Since homework solutions will be handed out on the day the homework is due **NO LATE HOMEWORKS WILL BE ACCEPTED**. (In other words, hand in whatever you have finished. You are also welcome to turn in homeworks *before* the due date if you cannot come to class on the due date.) If you cannot come to class for some reason, please mail the homework to me (should be postmarked a day **before** the due date).

All homeworks are to be done independently, with no help from the web, or other sources. If you have questions, please talk to the TA or the Instructor. Assignments are to be written up **NEATLY**. Badly written assignments **WILL NOT** be graded. Please staple your homework (every semester students lose parts of homeworks due to them not being

stapled). **It is your responsibility to make sure that you pick up all homeworks and handouts. All course information and handouts will be available on the web page.**

**Grading:** Final grades will be based on homework assignments, the quiz, the midterm exam, and the comprehensive final exam. The relative weights (these are subject to change) of these will be 15% for the homework total, 15% for the quiz, 30% for the midterm, and 40% for the final exam. The Instructor will make the use of the grading system that allows for + and – grades. Graduate students in this class will be given extra work on homeworks and exams.

**Syllabus:** The topics and order listed below are tentative and subject to change.

1. Graph exploration: connected components, topological sorting, strongly connected components (6 lectures).
2. Greedy algorithms: minimal spanning trees, shortest paths, scheduling (4 lectures).
3. Divide and Conquer algorithms: geometric algorithms, selection, lower bounds for minimum and sorting, Strassen’s matrix multiplication (5 lectures).
4. Dynamic programming: shortest paths, Warshall’s algorithm, optimal search trees (5 lectures).
5. String matching, other string type algorithms (2 lectures).
6. NP-completeness: introduction to reductions, the classes P and NP, NP-complete problems, approximation algorithms (6 lectures).