1 Introduction

Meetings. We meet twice a week, on Tuesdays and Thursdays, from 1:15 to 2:30pm, in room D106 LSRC.

Communication. The course material will be delivered in the two weekly lectures. A written record of the lectures will be available on the web, usually a day after the lecture. The web also contains other information, such as homework assignments, solutions, useful links, etc. The main supporting text is

TARJAN. *Data Structures and Network Algorithms.* SIAM, 1983.

The book focuses on fundamental data structures and graph algorithms, and additional topics covered in the course can be found in the lecture notes or other texts in algorithms such as

KLEINBERG AND TARDOS. *Algorithm Design.* Pearson Education, 2006.

Examinations. There will be a final exam (covering the material of the entire semester) and a midterm (at the beginning of October), You may want to freshen up your math skills before going into this course. The weighting of exams and homework used to determine your grades is

homework	35%,
midterm	25%,
final	40%.

Homework. We have seven homeworks scheduled throughout this semester, one per main topic covered in the course. The solutions to each homework are due one and a half weeks after the assignment. More precisely, they are due at the beginning of the third lecture after the assignment. The seventh homework may help you prepare for the final exam and solutions will not be collected.

Rule 1. The solution to any one homework question must fit on a single page (together with the statement of the problem).

Rule 2. The discussion of questions and solutions before the due date is not discouraged, but you must formulate your own solution.

Rule 3. The deadline for turning in solutions is 10 minutes after the beginning of the lecture on the due date.

Overview. The main topics to be covered in this course are

I Design Techniques;

II Searching;

III Prioritizing;

IV Graph Algorithms;

V Topological Algorithms;

VI String Algorithms;

VII NP-completeness.

The emphasis will be on algorithm **design** and on algorithm **analysis**. For the analysis, we frequently need basic mathematical tools. Think of analysis as the measurement of the quality of your design. Just like you use your sense of taste to check your cooking, you should get into the habit of using algorithm analysis to justify design decisions when you write an algorithm or a computer program. This is a necessary step to reach the next level in mastering the art of programming. I encourage you to implement new algorithms and to compare the experimental performance of your program with the theoretical prediction gained through analysis.