This course is an introduction to the design and analysis of algorithms. The study of algorithms and algorithmic thinking is the most quintessential topic in computer science. It forms a foundation for all areas of computer science, with numerous applications to all subjects having to do with computing and communication.

In this course, we will introduce a collection of algorithmic ideas and techniques that underlie the ubiquity of every day computing and communication in the modern world. The basic algorithmic design ideas include: greedy algorithms, divide-and-conquer strategy, sorting and searching, hashing, dynamic programming, network flows and matchings, as well as randomization and approximate algorithms. We will emphasize techniques for constructing correct and efficient algorithms, and for mathematically analyzing about them. We will also introduce computational intractability, in particular NP-completeness.

However, the course should not be viewed primarily as a collection of techniques. The main focus of the course is a coherent and unifying narrative, that weaves together a perspective that is increasingly relevant in today’s interconnected world. This is a fascinating story of an intellectual enterprise that is crucially fueling today’s computer revolution.

Students are expected to learn and think hard. This course is likely to be one where you may spend many hours per week. Your work should be your own, not copied from any source. The use of materials (such as homework solutions) from previous versions of the course or from the Internet is considered plagiarism and will warrant strict action in accordance with university policy. For each homework, students should self organize teams. Each team consists of two students. The two students of a team should discuss about all the problems of the problem set, as well as lecture notes. The team hands in one set for both students. Each member of the team should contribute equally.

We expect to have 8 to 10 homework sets. Each homework set designates two problems, to be handed in and will be graded. Each team should hand in two designated homework problems in
separate sheets of papers. The solution to each problem should not exceed a maximum of 2 pages. You
should try on your own for all the problems in the set.

Every student is expected to attend a weekly recitation class, to be run by the TAs. There are
4 one-hour weekly recitation classes, spread on Tuesdays, Wednesdays and Thursdays every week.
All four sessions are identical; attend any one. At the recitation classes, the TAs will discuss the
homework problems, especially the other ungraded homeworks.

Computer programming skills and experience may be helpful in guiding intuition, but primarily
students are expected to exercise mathematical and algorithmic thinking and logical deduction.
Mathematical rigor in presentation is valued. Essays or philosophical discussions are not acceptable
as answers to precise questions. Homework grades count for approximately 30% of the grades. Each
graded homework problem is weighed equally. We take best of $n - 1$ out of $n$ homework sets to
calculate your total homework grades. Each homework set is due at the beginning of the class on
the due date. **No late homeworks** will be accepted. Homeworks should be written legibly. (It is
your responsibility that I and the TA can understand your answers, and not the other way around.)
There will be a written midterm, which counts for roughly 30%, and a written final exam, which
counts for roughly 40%. Doing homeworks, and truly understand how to solve the problems, is
important, not only for the 30% of the grades, but also, some homework problems may appear on
the exams.

All students are expected to attend **all classes**. (For exceptions which are not unforeseeable, such
as for religious observances, students should notify the Professor within the first two weeks of class.
Exception to attendance can be granted for emergencies such as illnesses.) Attendance is vitally
important to your understanding of the course material.

Students are encouraged to discuss among themselves on the course material. The midterm and
final exam are closed book, closed notes. All students are expected to observe faithfully the code
of academic integrity. Cheating of any sort is subject to an F grade for the entire course. Class
participation and/or extra credits on homeworks will be taken into account for the final grade.

You are encouraged to ask questions, in class or after class, and during TA’s or my office hours.
(You should ask questions that pertain to the subject matter of the class, and avoid the “How do
I do this problem” variety.) See your TA first regarding any grading questions, and only if such a
problem is not resolved you should come to see me.

Tentative Schedule:

- **Week 1** (one lecture 9/7): Introduction, Class overview, Insertion sort, and Asymptotics.
- **Week 2** (9/12 & 9/14): Induction, Divide and conquer, Merge sort, Recurrence, Binary search,
  Matrix multiplication, Master theorem.
- **Week 3** (9/19 & 9/21): Randomized Algorithm, Quicksort and Its analysis.
- **Week 4** (9/26 & 9/28): Selection in linear time, Lower bound on comparison sorts, Hashing.
- **Week 5** (10/3 & 10/5): Universal hashing, Binary search trees.
- **Week 6** (10/10 & 10/12): Balanced tree structures, Union-Find.
- **Midterm Wed., 10/18, from 7-9 PM. Closed book. Closed notes.) Ag Hall Room 125.**
- **Week 8** (10/24 & 10/26): Optimal BST. Amortized Analysis.
- **Week 9** (10/31 & 11/2): Graphs. BFS, DFS.
• Week 10 (11/7 & 11/9): Topological sort, Strongly connected components, Minimum spanning tree. Kruskal and Prim’s Algorithms.

• Week 11 (11/14 & 11/16): Dijkstra’s and Bellman-Ford Algorithms. All-pairs shortest paths.


• Week 14 (12/5 & 12/7): Combinatorial Optimization, P, NP, and NP-completeness.

• Week 15 (12/12): More NP-completeness.

• Final Exam: Fri, 12/15, 5:05PM - 7:05PM