Course Objectives and Outcomes

Course Objectives

This course assumes that students know how to analyze simple algorithms and data structures from having taken 6.006, and introduces students to design of computer algorithms, as well as analysis of sophisticated algorithms. Upon completion of this course, students will be able to do the following:

- Analyze the asymptotic performance of algorithms.
- Write rigorous correctness proofs for algorithms.
- Demonstrate a familiarity with major algorithms and data structures.
- Apply important algorithmic design paradigms and methods of analysis.
- Synthesize efficient algorithms in common engineering design situations.

Course Outcomes

Students who complete the course will have demonstrated the ability to do the following:

- Argue the correctness of algorithms using inductive proofs and invariants.
- Analyze worst-case running times of algorithms using asymptotic analysis.
- Describe the divide-and-conquer paradigm and explain when an algorithmic design situation calls for it. Recite algorithms that employ this paradigm. Synthesize divide-and-conquer algorithms. Derive and solve recurrences describing the performance of divide-and-conquer algorithms.
- Describe the dynamic-programming paradigm and explain when an algorithmic design situation calls for it. Recite algorithms that employ this paradigm. Synthesize dynamic-programming algorithms, and analyze them.
- Describe the greedy paradigm and explain when an algorithmic design situation calls for it. Recite algorithms that employ this paradigm. Synthesize greedy algorithms, and analyze them.
- Explain the major graph algorithms and their analyses. Employ graphs to model engineering problems, when appropriate. Synthesize new graph algorithms and algorithms that employ graph computations as key components, and analyze them.
• Explain the different ways to analyze randomized algorithms (expected running time, probability of error). Recite algorithms that employ randomization. Explain the difference between a randomized algorithm and an algorithm with probabilistic inputs.

• Analyze randomized algorithms. Employ indicator random variables and linearity of expectation to perform the analyses. Recite analyses of algorithms that employ this method of analysis.

• Explain what amortized running time is and what it is good for. Describe the different methods of amortized analysis (aggregate analysis, accounting, potential method). Perform amortized analysis.

• Explain what competitive analysis is and to which situations it applies. Perform competitive analysis.

• Compare between different data structures. Pick an appropriate data structure for a design situation.

• Explain what an approximation algorithm is, and what is the benefit of using approximation algorithms. Be familiar with some approximation algorithms, including algorithms that are PTAS or FPTAS. Analyze the approximation factor of an algorithm.