CSCI 48400 Review for Test Three

Chapter 12:
Understand what it means for a problem to be decidable (or not)
Be able to state the Halting Problem and discuss its implications.
Be able to outline a proof of the Halting Problem.
Be able to carry out a reducibility argument.

Chapter 14:
Be able to give an informal definition for the sets P, NP, and NP-complete.
Be able to name some NP-complete problems.

Compiler theory:
There will be a question or two based on the compiler theory session (slides are online).

Divide and Conquer:
Be able to describe the divide-and-conquer design technique and give at least one example of an algorithm that uses this technique.
Be able to solve a divide-and-conquer recurrence relation given the solution formula.

Dynamic Programming:
Be able to describe the characteristics of problems suitable for this design approach and give at least one example of an algorithm that uses this technique.
Be able to solve a problem using dynamic programming given some hints.

Greedy Algorithms:
Be able to describe the greedy algorithm design technique and give at least one example of an algorithm that uses this technique.
Be able to carry out Dijkstra's shortest path problem on a given graph.
Be able to generate Huffman codes for a given frequency analysis

Amortized Analysis:
Be able to describe the goal of an amortized analysis for a given algorithm.
Be able to carry out the Move-To-Front access algorithm given a list and a sequence of access requests.
Be able to carry out an amortized analysis (using the aggregate method, the accounting method, or the potential method, as specified) for a given algorithm given some hints.

Approximation Algorithms:
Be able to give an informal definition for the set NP hard and its relationship to the classes P, NP, and NP-complete.
Be able to give the characteristics of a good approximation algorithm
Be able to define what it means for an approximation algorithm to give an answer "close" to the optimum result.
Be able to carry out the First Fit Decreasing bin-packing algorithm on a given list of item sizes.