CSE 100: Algorithm Design and Analysis
Key Topics of Midterm Exam 3 — Spring 2023

---

* Disclaimer: I do not guarantee that the following list is complete.

This exam will cover Chapters **15, 16**.

1. Dynamic Programming. Chapter 15.
   (a) Recall that you learned 4 examples of dynamic programming during class: Fibonacci numbers, Rod Cutting, Matrix chain multiplication, and Computing LCS in Chapter 4. Also the discussion session includes additional examples.
   (b) Bottom up approach vs Top down approach. For example, given a bottom-up DP, can you give a pseudocode of top-down approach using memoization? We saw how to do this for Fibonacci and Rod Cutting.
   (c) Fill out the DP table. Find an optimum solution.
   (d) Given a recursion, can you give an efficient implementation (pseudocode) of the recursion using DP?
   (e) Can you describe a DP based algorithm? (I'll often just ask you to give an algorithm for computing the optimum, for example, the maximum profit one can get out of a rod of length n). To get full points, you must have the following: 1. DP table entries, 2. Recursion 3. In which order you compute the entries, and 4. What is the optimum.
   (f) If you have a DP based algorithm for computing the optimum (objective), can you modify it, so you can construct an optimal solution.
   (g) Can you analyze the running time? You must specify 1. the number of DP entries/subproblems, 2. RT for computing each entry 3. RT for computing the optimum.
   (h) Bonus: a new DP problem (35pts).
2. Greedy Algorithms. Chapter 16.
   (a) We covered two examples of greedy algorithms. Interval Selection (Activity Selection) and Huffman code.
   (b) There could be many different greedy algorithms. If a greedy algorithm is not optimal, can you find a counter example? (that is, an example for which the algorithm fails to produce an optimal solution).
   (c) We learned a key lemma to prove the optimality of EF for the Interval Selection problem. You should be able to prove it. Do you understand why the key lemma implies the optimality of EF?
   (d) Running the Huffman algorithm on an instance: Find the resulting Huffman code. (character, code) representation. Tree representation.
   (e) In Huffman code, understanding prefix (prefix-free) code and understanding the structure of the tree