CS 3510: Homework 2

Due on Thursday, June 3, 11:59pm via Gradescope.
Late submission with no penalty until Friday, June 4, 11:59am.

Professor Brito

CS 3510 Staff

Instructions.

For the graded problems, you are allow to use the following theorems and algorithm from classes as black-boxes. In doing so, you do not need to explain your algorithm. Clearly state what is your input and how you use the output. You can report their runtime as presented in class with no further explanation.

- Master Theorem, QuickSort, MergeSort and the Merging subroutine, FastSelect (median of medians), Fast Multiplication.

Suggested reading.

Chapter 2 (for D&C) and chapter 6 (for Dynamic Programming). You do not need to cover topics with haven't cover in lectures.

Suggested Problems.

All problems are from Chapter 6. The numeration corresponds to the 2006 edition of the book. You do not need to turn this problem in.

- 6.1 (Max sum contiguous sequence.)
- 6.2 (Hotel stops).
- 6.3 (Yuckdonald).
- 6.4 (Reconstructing a string).
- 6.6 (Parenthesize a multiplication).
- 6.7 (palindromic sequence)($*$).
- 6.8 (Longest common substring)($*$).
- 6.17 (the coin problem).
- 6.18 (the coin problem 0/1 version).
- 6.26 (sequence alignment).

Practice problem.

problem 6.19 **(the coin problem using at most k coins).**

Problem 1

(*Exam 1, Spring 2021*) Given a sorted array $A[1, 2, \ldots, n]$ containing all the integers in the range $\{1, 2, \ldots, n-1\}$ exactly once, expect for one of them which appears twice. Design a divide and conquer algorithm to find the only repeated element. Describe your algorithm with words (no pseudocode!) and justify its correctness. Analyze and justify its running time. Make sure to include the recurrence relation.

Problem 2

You are given a list of n different denominations of coins in an array $D[1...n]$. You are also given the number of coins you have for each denomination in an array $F[1...n]$ such that $F[i]$ is the number of coins available for denomination $D[i]$. Given an input value V, design a dynamic programming algorithm to determine if it is possible to make change for it. Your algorithm must run in time $\mathcal{O}(n \cdot V \cdot \max_i F[i])$.

Please answer the following parts:

(a) Define the entries of your table in words. E.g. $T(i)$ or $T(i, j)$ is ...

(b) State recurrence for entries of table in terms of smaller subproblems. Briefly explain in words why it is correct. Remember to state your base case(s) as well.

(c) Write pseudocode for your algorithm to solve this problem.

(d) Analyze the running time of your algorithm.