Homework 4

Due: October 11, 2022 - 14:30 ET

This homework must be typed in LaTeX and handed in via Gradescope.

Please ensure that your solutions are complete, concise, and communicated clearly. Use full sentences and plan your presentation before you write. Except in the rare cases where it is indicated otherwise, consider every problem as asking you to prove your result.

Problem 1

Tianren *really* likes **reversible** words, that is words that read the same forwards and backwards (e.g, racecar, kayak, level). He likes them so much, that he has started to try and find reversible words in any strings that he sees.

1. Given a string with n characters from the English alphabet (you may assume that they are lower case), design an efficient dynamic program that finds the longest substring (a consecutive sequence of characters) of the input string containing a *reversible* word.
2. Provide proof of the correctness of the algorithm and analyze its time complexity and memory utilization.

Problem 2

In his dorm room, Hammad has a box of textbooks, $B = \{b_1, b_2, \ldots, b_n\}$, each with corresponding weight w_i, which is a positive integer. When Hammad moves back home at the end of the semester, he wants to bring as many pounds of books home as possible. However, he has a problem: the moving company can only move at most W pounds of books, so Hammad will have to decide which books to take with him and which to leave behind.

Design an algorithm to help Hammad find maximum weight of books he can bring back that does **not** exceed the weight limit of the moving company (W). Your algorithm should have runtime $O(nW)$.

Provide proof of the correctness of the algorithm and analyze its time complexity and memory utilization.

Hint: Each element b_i is either in or out of the maximal sum.

Problem 3

Rohit gave Rob a case of N magical coins arranged in an array. Each coin has an initial value (C_i) which increases with every time-step (t_j). Time-step starts at 1 ($t_1 = 1$) and the value of any coin (C_i) at time (t_j) would be $C_i \cdot t_j$, i.e., the product of the initial value of the coin and the current time-step.

Rob wants to sell the coins but he can only sell one coin at a time. Also, at any time-step he can only sell a coin from the *start* of the array or the *end* of the array otherwise he would have to bear the curse of grading 1000 assignments (we really don't want Rob to be cursed).

1. Help Rob find the maximum profit that he can make by giving him an algorithm that runs in $O(N^2)$. Prove correctness of your algorithm, and justify its runtime and memory utilization.
2. Rob decided to go greedy by selling the biggest coin from either end at any given timestep, thereby decreasing the time complexity to $O(N)$. Is this strategy flawed? Explain why or why not.