Problem Set 8

All parts are due on May 1, 2020 at 6PM. Please write your solutions in the LaTeX template provided. Aim for concise solutions; convoluted and obtuse descriptions might receive low marks, even when they are correct. Submit solutions to Gradescope; **there is no coding part to submit.**

Please solve each of the following problems using **dynamic programming**. For each problem, be sure to define a set of subproblems, relate the subproblems recursively, argue the relation is acyclic, provide base cases, construct a solution from the subproblems, and analyze running time. Correct but inefficient dynamic programs will be awarded significant partial credit.

For each problem below, please indicate whether the requested running time is either:
(1) **polynomial**, (2) **pseudopolynomial**, or (3) **exponential** in the size of the input.
This categorization will be worth **3 points per problem**.

Problem 8-1. [25 points] **Oil Well that Ends Well**

The oil wells of tycoon Ron Jockefeller will produce m oil barrels this month. Ron has a list of n orders from potential buyers, where the ith order states a willingness to buy a_i barrels for a total price of p_i (not per barrel), which may be negative.[1] Each order must be filled completely or not at all, and can only be filled once. Ron does not have to sell all of his oil, but he must pay s dollars per unsold barrel in storage costs. Describe an $O(nm)$-time algorithm to determine which orders to fill so that Ron can maximize his profit (which may be negative).

Problem 8-2. [25 points] **Splits Bowling**

In Lecture 15, we introduced **Bowling**: a one-player game played on a sequence of n pins, where pin i has integer value v_i (possibly negative). The player repeatedly knocks down pins in two ways:

- knock down a single pin, providing v_i points; or
- knock down two adjacent pins i and $i + 1$, providing $v_i \cdot v_{i+1}$ points.

Pins may be knocked down at most once, though the player may choose not to knock down some pins. A Bowling variant, **Split Bowling**, adds a third way the player can knock down two pins forming a **split**, specifically:

- knock down two pins i and $j > i + 1$ if all pins in $\{i + 1, \dots, j - 1\}$ between them have already been previously knocked down, providing $v_i \cdot v_j$ points.

Describe an $O(n^3)$-time algorithm to determine the maximum score possible playing Split Bowling on a given input sequence of n pins.

[1] Earlier this year, oil futures contract prices went negative: people were paying money to not accept delivery of oil because demand for oil had fallen dramatically and there was a shortage of places to store oil.

Problem 8-3. [25 points] **Quarter Partition**

Given a set $A = \{a_0, \ldots, a_{n-1}\}$ containing n distinct positive integers where $m = \sum_{a_i \in A} a_i$, describe an $O(m^3 n)$-time algorithm to return a **partition** of A into four subsets $A_1, A_2, A_3, A_4 \subsetneq A$ (where $A_1 \cup A_2 \cup A_3 \cup A_4 = A$) such that the maximum of their individual sums is as small as possible, i.e., such that $\max\left\{\sum_{a_i \in A_j} a_i \mid j \in \{1, 2, 3, 4\}\right\}$ is minimized.

Problem 8-4. [25 points] **Corrupt Chronicles**

Kimmy Jerk is the captain of the USS Exitcost, a starship charged with exploring new worlds. Each day, Capt. Jerk uploads a **captain's log** to the ship's computer: a string of at most m lowercase English letters and spaces, where a **word** in a log is any maximal substring not containing a space.

One day, Capt. Jerk is abducted, and Communications Officer Uhota Nyura goes to the captain's logs looking for evidence. Unfortunately, the log upload system has malfunctioned, and has **corrupted** each of the last n logs by dropping all spaces. Officer Nyura wants to restore the spaces based on Capt. Jerk's speech patterns in previous logs. Given a list L_c of the n corrupted logs, as well as a list L_u of $O(m^2 n)$ uncorrupted logs from before the malfunction, Officer Nyura wants to:

- for each word w appearing in any log in L_u, compute $f(w)$: the positive integer number of times word w appears in L_u (note, $f(w)$ is zero for any word w not appearing in L_u); and
- for each log $\ell_i \in L_c$, return a **restoration** R_i of ℓ_i (i.e, a sequence of words R_i whose ordered concatenation equals ℓ_i), such that $\sum_{w \in R_i} f(w)$ is maximized over all possible restorations.

Describe an $O(m^3 n)$-time algorithm to restore Capt. Jerk's logs based on the above protocol.