Homework 3

Due: 7 November, 2025 6pm PT

Unless stated otherwise, you should always prove the correctness of your algorithms and analyze their runtimes.

Each problem is graded on the coarse scale of $\checkmark^+$, $\checkmark$, $\checkmark^-$ and no $\checkmark$. It is also assigned a multiplier, denoting the relative importance of the problem. Both correctness and presentation are grading criteria.

Please read and make sure you understand the collaboration policy on the course missive. Extra credit problems are clearly marked below (see course missive for the details of grade calculations).

Remember to prove all your (non-elementary and not shown in class) mathematical claims, unless stated otherwise.

Each group of students should submit only 1 pdf to the corresponding Gradescope assignment.

Proving correctness of greedy algorithms

Recall the generic strategy for proving the correctness of a greedy algorithm:

1. Our algorithm makes a series of choices, $c_1, c_2, \ldots, c_k$.
2. We will show by induction on i from 0 to k (or 1 to k) that there exists an optimal solution OPT_i that is "compatible with" (e.g. contains) our first i choices (you should be very precise about exactly what "compatible" means in the context of the algorithm, in the induction hypothesis).
3. To show the induction step from i to $i+1$, we start from OPT_i that is compatible with choices $c_1, \ldots, c_i$. We will then modify OPT_i into OPT_{i+1} that is compatible also with choice c_{i+1}. This can usually (but not always) be done with a swapping argument as presented in class.
4. Wrapping up: use the induction hypothesis at $i = k$ to show that the final solution we produced is in fact an optimal solution. This wrapping up is necessary for some algorithms and unnecessary for some others.

Problem 1

(3 ✓s total)

Picnic Day is getting fun. In the year 2943 iteration, the day is jam-packed with concerts, many of which are overlapping. As an avid music-lover, you want to attend as many concerts as possible (also because they serve free food, but that's irrelevant, clearly).

It turns out that the concerts are classical music concerts, so it would be very rude to arrive late or to leave early — you have to stay for the entirety of a concert that you choose to go to. Given that concerts have overlaps, our goal is to come up with an algorithm to compute the most concerts you can attend.

The input to the algorithm is a list of events, with start time s_i and end time t_i for event i.

For the following three greedy strategies, either *prove* its correctness or find a *counterexample* to show that the algorithm is not optimal. If you decide to show a counterexample, be sure to also explain why it is a counterexample or sub-optimal solution, including any assumptions in tie-breaking if applicable.

1. Greedily pick concerts in increasing order of start time. (Sort concerts in increasing order of start time s_i, and repeatedly run the following process: pick the first date in the list, then delete all overlapping dates).
2. Greedily pick concerts in decreasing order of start time. (Sort concerts in decreasing order of start time s_i and repeatedly run the following process: pick the first date in the list, and delete all overlapping dates).
3. Greedily pick the shortest concerts, i.e. in increasing order of length $(t_i - s_i)$.

Problem 2

(2 ✓s)

A video of you saving Cheeto from being eaten by Gunrock went viral. While you are basking in your newfound fame, it does present an annoying challenge: everyone wants to come to your birthday party (that your roommate is organizing), and they would be offended if they don't get to see you.

Being an introvert, you want to find the fewest time points to briefly show up to the party, say hi to people there, then go back into your room. Knowing that the i^{th} guest will arrive at time s_i and leave at time t_i, design a polynomial-time algorithm to figure out which time points to show up to, and prove its optimality.

Problem 3

(5 ✓s total)

You are an inhabitant of Lineland (the 1-dimensional space), going on a spring break road trip. Without loss of generality, you are starting from the origin (coordinate 0), with a final destination 5000 miles away. Each day, your friends and you are only able to drive at most 200 miles before you get too tired and have to sleep. Fortunately, you know the towns along the way where there are motels for you to sleep in. These are locations x_i for $i \in [1..n]$, for $x_i \in [0, 5000]$.

For the following problems, the intended algorithm may or may not be a greedy algorithm. Be sure to think through different algorithmic possibilities, and propose a correct algorithm with a (correct) proof of correctness/optimality.

1. (3 ✓s) Find a polynomial time (in n) algorithm that computes a trip with fewest stops, going from location 0 to location 5000.
2. (2 ✓s) You just remembered that different motels in different towns actually cost drastically differently. The town/motel at location x_i has a cost $p_i \geq 0$. The goal now is to minimize the

total motel cost along the trip. Give a polynomial (in n) time algorithm to compute such a trip.

Problem 4

(**Extra credit** 1 ✓)

Pick one problem and propose a natural (but incorrect) greedy algorithm, and show a counterexample. When you show a counterexample, be sure to write out all the greedy steps, including explanations of any potential tie-breaking that you chose.

Problem 5

(**Extra credit, not easy** 4 ✓s)

Suppose we are given a binary string s of length n, and an input non-negative integer k.

Give a linear time algorithm that finds the longest subsequence of s whose binary value is at most k. The subsequence is allowed to have leading zeroes.

You will only get credit if you prove the correctness of your algorithm.

(**Note**: Feel free to not use the greedy algorithm proof template at the beginning of the homework, if you find a more direct way of arguing.)