Written Problem Set 3: Greedy

The first thing you should do in ps3.tex is set up your name as the author who is submitting by replacing the line, `\submitter{TODO: your name}`, with your name and UVA email id, e.g., `\submitter{Grace Hopper (gmh1a)}`.

Next, put your response for problem 1 into problem1.tex, problem 2 into problem2.tex, etc.

Before submitting , also remember to:

– List your collaborators and resources, replacing the TODO in `\collaborators{TODO: replace ...}` with your collaborators and resources. (Remember to update this before submitting if you work with more people.)

– Replace the second line in ps2.tex, `\usepackage{algo}` with `\usepackage[response]{algo}`.

Collaborators: TODO: replace this with your collaborators (if you did not have any, replace this with *None*)

Resources: TODO: replace this with your resources (if you did not have any, replace this with *None*)

Problem 1: The Fearless Hyena

Jackie Chan is trying to plan a stunt for his next movie. He will flee from the bad guys by rapidly descending a tall building. Periodically, there are balconies on the face of the building. Jackie will leap from balcony to balcony until he eventually reaches the ground. For this problem you will write an algorithm to help Jackie to complete the stunt unharmed.

The building is n meters tall, and you have a list of the heights of the m balconies $[h_1, \ldots h_m]$ (ordered from highest balcony to lowest). Through trial and error, Jackie knows that any fall greater than k meters will cause injury. We will devise an algorithm which returns the shortest sequence of balconies Jackie can traverse to get to the ground without injury.

1. First show that this problem has optimal substructure, i.e. show that the optimal solution to the problem for the building of height n is an extension of the optimal solution for a shorter building.

2. Next give the greedy choice function for this problem, i.e. give a way to identify *some* balcony guaranteed to be used in *some* optimal solution. Show the correctness of your greedy choice function using an exchange argument.

3. Finally, Use the previous two items to devise a greedy algorithm to solve this problem. Analyze your algorithm's running time.

Problem 2: Paletas

It's a hot sunny day, and Nate has just bough n paletas (popsicles) from La Flor Michoacana, his favorite place in town for frozen treats. In his excitement to try all the flavors, Nate did not consider that as soon as they left the freezer they would all begin to melt! Each paleta melts at a different rate, and Nate is able to eat different paletas at different speeds before getting a brain freeze. Help him to minimize how much melts!

Suppose that we have paletas $1, \ldots, n$. Let m_i represent the amount of paleta i that melts per minute (in milliliters). Let t_i be the time it takes Nate to eat paleta i. Once he begins eating a paleta, Nate must finish that one before beginning to eat another. Let f_i represent the cumulative number of minutes passed before Nate finishes paleta i, note that its value depends on the order he chooses to eat each paleta.

We can express the total amount that paleta i melted as $m_i \cdot f_i$. Devise a greedy algorithm to help Nate pick the order to eat the paletas. The optimal order will minimize the total amount melted across all paletas, i.e. minimize the quantity:

$$\sum_{i=1}^{n} m_i \cdot f_i$$

Greedy Choice. Describe the greedy choice function for your algorithm.

Optimality of Greedy Choice. Using an exchange argument, show that your greedy choice yields an optimal solution to this problem. Similar to the Huffman Coding exchange argument, all possible solutions contain all items in the list, just in different orders. Therefore your exchange argument should compare your greedy solution to one that has at least two items out of order (as a hint, if there are two items out of order, then there are two *adjacent* items out of order).

Algorithm. Describe your greedy algorithm using the greedy choice function you identified above.

Running Time. Analyze, asymptotically, the worst-case running time of your algorithm.

Problem 3: Deadline Management

You probably have, like, a lot of homework. Let's say you have n total assignments that we'll label $a_1, \ldots, a_n$ in your product. Each assignment $a_i = (t_i, d_i)$ has a deadline d_i when it is due and an estimated amount of time it will take to complete, t_i.

Let us assume that when you work on one assignment, you give it your full attention and do not begin any other assignment (yes, this is different from how I recommend you do CS3100 homework, but it makes the problem harder without this assumption). In some cases, your demanding professors ask too much of you. It may not be possible to implement all of your homework on time given the deadlines $d_1, \ldots, d_n$; indeed, some homework may *have* to be delivered late. Your goal as a responsible student is to minimize the lateness of your latest assignment.

If you begin assignment a_i at time s_i, you will complete it at time $c_i = s_i + t_i$. The lateness value, denoted ℓ_i, for a_i is the value

$$\ell_i = \begin{cases} c_i - d_i & \text{if } c_i \geq d_i \\ 0 & \text{otherwise} \end{cases}$$

Using your greedy choice function, we will produce schedule, call it O, that specifies the order in which you implement features which minimizes the maximum ℓ_i for all assignments, i.e.

$$\min_O \max_i \ell_i$$

In other words, you do not want to complete in any assignment too late, so you minimize the lateness of the latest assignment you submit.

Greedy Choice. Describe the greedy choice function for your algorithm.

Optimality of Greedy Choice. Using an exchange argument, show that your greedy choice function yields an optimal solution to this problem.

Algorithm. Describe your greedy algorithm using the greedy choice function you identified above.

Running Time. Analyze, asymptotically, the worst-case running time of your algorithm.

Problem 4: MST Optimal Substructure

For this problem we will prove that the Minimum Spanning Tree problem exhibits optimal substructure.

Consider a minimum spanning tree T of a graph G. Supposer that the edge (u, v) is used in T. We can think of T as being in three parts: The edge (u, v), the subset of T connecting all nodes reachable from u without crossing (u, v) (call this T_u), and the subset of T connecting all nodes reachable from v without crossing (u, v) (call this T_v).

Prove the following claim: If T is a minimum-spanning tree for G, then T_v must be a minimum spanning tree for its portion of the graph. Hint: If T_v is NOT optimal, then what must be true?

Proof: