Computer Science CSC373H
St. George Campus

September 21, 2022
University of Toronto

Homework Assignment #2

Due: October 5, 2022, by 11:00 am

- ***You must submit your assignment through the Crowdmark system.*** You will receive by email an invitation through which you can submit your work. If you haven't used Crowdmark before, give yourself plenty of time to figure it out!
- You must submit a ***separate*** PDF document with for **each** question of the assignment.
- To work with one or two partners, you and your partner(s) must form a ***group*** on Crowdmark (one submission only per group). We allow groups of ***up to three*** students. Submissions by groups of more than three students will not be graded.
- The PDF file that you submit for each question must be typeset (***not*** handwritten) and clearly legible. To this end, we encourage you to learn and use the LaTeX typesetting system, which is designed to produce high-quality documents that contain mathematical notation. You can use other typesetting systems if you prefer, but handwritten documents are not accepted.
- If this assignment is submitted by a group of two or three students, for each assignment question the PDF file that you submit should contain:
  1. The name(s) of the student(s) who ***wrote*** the solution to this question, and
  2. The name(s) of the student(s) who ***read*** this solution to verify its clarity and correctness.
- By virtue of submitting this assignment you (and your partners, if you have any) acknowledge that you are aware of the homework collaboration policy for this course, as stated here.
- For any question, you may use data structures and algorithms previously described in class, or in prerequisites of this course, without describing them. You may also use any result that we covered in class (in lectures or tutorials) by referring to it.
- Unless we explicitly state otherwise, you should justify your answers. Your paper will be marked based on the correctness and efficiency of your answers, and the clarity, precision, and conciseness of your presentation.
- The total length of your pdf submission should be no more than 4 pages long in a 11pt font.

Question 1. (10 marks) Consider Huffman's algorithm.

a. Find a small set of symbols Γ and associated frequencies f, such that: (i) there is exactly one symbol with frequency $\frac{1}{3}$, (ii) all the other symbols have frequency strictly less than $\frac{1}{3}$, and (iii) executing Huffman's algorithm on (Γ, f) can produce a codeword of length 1. Justify your answer.

b. Prove that for all sets of symbols Γ with associated frequencies f such that $f(x) < \frac{1}{3}$ for all $x \in \Gamma$, executing Huffman's algorithm on (Γ, f) does not produce any codeword of length 1.

Question 2. (10 marks) You own a company of trucks and you want to use them to supply a set of *cities*. Each pair of cities may be connected by a *road* that traverses no other city (some roads may be one-way). Each road may go through a single *tunnel*. Each tunnel has a *height* that specifies the maximum height of trucks that can traverse it: a tunnel of height h can be traversed only by trucks of height *at most* h.

An *itinerary* from a city s to a city t is a sequence of alternating cities and roads that start at s and finishes at t (without repeating any city). The *height H of an itinerary* is the *minimum height* of all the road tunnels that it traverses; note that a truck with height more than H can *not* take this itinerary. So for your truck company, a *best* itinerary from s to t is one whose height H is *maximum among all itineraries from s to t*. We want an efficient algorithm to find best itineraries starting from a given city s.

More precisely, the algorithm's input is:

1. A set of n cities $\{1, 2, \ldots, n\}$.
2. For each city i, a list $L[i]$ containing pairs of the form (j, h_{ij}) with $j \neq i$ and $h_{ij} \geq 0$.

   $L[i]$ contains (j, h_{ij}) iff: (a) there is a road *from* city i *to* city j, and (b) the height of the tunnel on that road is h_{ij}; if there is no tunnel on that road $h_{ij} = +\infty$.
3. A start city $s \in \{1, 2, \ldots, n\}$.

The algorithm must find *for every city t*: (1) a best itinerary from s to t, and (2) the height of this itinerary. If no itinerary from s to t exists, the algorithm should report that the height of the best itinerary from s to t is 0.

Your task is to modify Dijkstra's algorithm to do so. First describe these modifications clearly and concisely in English, and then give the full algorithm's pseudocode. Do *not* prove its correctness.

Question 3. (20 marks) Isabella Rosselini has n assignments $A_1, A_2, \ldots, A_n$ due ***now***, and she has not started working on any of them (not surprisingly, she was making a movie). Assignment A_i takes her d_i days to do, and has penalty p_i for ***each day*** it is submitted late (assume that d_i and p_i are positive integers). For example, if Isabella starts by doing assignment A_i first, her penalty on this assignment will be $p_i \cdot d_i$, and if Isabella does the assignments in the order $S = A_1, A_2, \ldots, A_n$, the total penalty (sum of penalties) is:

$$P_S = p_1 d_1 + p_2(d_1 + d_2) + p_3(d_1 + d_2 + d_3) + \cdots + p_n(d_1 + d_2 + \cdots + d_n)$$

because if she does them in this order, she will finish assignment A_i after $d_1 + d_2 + \cdots + d_i$ days (and then she will immediately start working on assignment A_{i+1}). Note that Isabella can only work on one assignment at a time.

There are $n!$ different orders in which the assignments can be done. An ordering of the assignment is ***optimal*** if it minimizes the total penalty. Isabella just learned about Greedy Algorithms, so she hopes that this approach will help her find an optimal ordering.

For each of the following three greedy strategies for ordering the assignments, either show that it always results in an optimal ordering of the assignments or provide a counterexample:

a. Do the assignments in order of increasing length (number of days that they take).
That is, schedule them in an order such that if $d_i < d_j$ then A_i occurs before A_j in that order.

b. Do the assignments in order of decreasing daily penalty.
That is, schedule them in an order such that if $p_i > p_j$ then A_i occurs before A_j in that order.

c. Do the assignments in order of decreasing penalty-to-length ratio.
That is, schedule them in an order such that if $p_i/d_i > p_j/d_j$ then A_i occurs before A_j in that order.

In the three strategies above, do ***not*** assume that the p_i's, d_i's, or p_i/d_i's are necessarily distinct.