CPSC 320: Intermediate Algorithm Design and Analysis
Assignment #4, due Monday, June 6^{th}, 2016 at 2:15pm in Room x235, Box 32

[15] 1. After the end of term 2, a student decides to ride her bicycle from Vancouver to Halifax. She starts on the road at kilometer 0. Along the way there are n hotels $H_1, \ldots, H_n$, at distances $h_1 < h_2 < h_3 < \ldots < h_{n-1} < h_n$ from the trip's starting point respectively. The only places where the student is willing to stop for the night are these hotels, but she can choose which hotels she stops at. She must stop at the final hotel H_n since this is her destination (she is planning on flying back).

Ideally, this student would like to travel 100Km per day, but this may not be possible, depending on the spacing of the hotels. In order to determine how close the student got to this target distance, she defines the *penalty* for a day to be $|100 - x|^{1.5}$ where x is the distance traveled that day. Her objective is to plan her trip in a way that minimizes the sum of the daily penalties over all days of the trip.

[3] a. One possible solution would be to use a greedy algorithm that determines where to stop on day i by choosing the hotel whose coordinate is closest to $x_{i-1} + 100$, where x_{i-1} is the coordinate of the hotel that the student stopped at on day $i - 1$.

Give an example to show that this greedy algorithm does **not** always succeeds in minimizing the sum of the daily penalties over all days of the trip.

[4] b. Let D_k be the minimum sum of the daily penalties, over all possible trips that end at hotel H_k (that is, the student does not go any further). Write a recurrence relation for D_k, and explain why your recurrence relation is correct.

[6] c. Using this recurrence relation, describe a dynamic programming algorithm that computes D_n. Your algorithm should return the list of hotels at which the student will spend the nights to achieve the minimum sum of daily penalties (not only the value D_n).

[2] d. Analyze the running time of your algorithm as a function of n.

[17] 2. Consider the following game: there is a set of letters, where each letter c is assigned a value $value(c)$. A random string S of $2n$ letters is chosen, and each player will select, in turn, either the first or the last element of the string (which is then removed). At the end of the game, each player computes the sum of the values of his/her letters, and the player with the largest total value wins. In this question, you will design an $O(n^2)$ time algorithm to compute an optimal strategy for the first player.

Let $V[i, len]$ denote the maximum total value of the letters that the first player can obtain if only the letters in positions $i, i+1, \ldots, i+len-1$ are left, assuming that the second player does not make any mistake (i.e., the second player follows his optimal strategy).

(a) (5 points) Describe a recurrence relation for $V[i, len]$. Your recurrence will need to treat the case where len is odd differently from the case where len is even, since the first player tries to maximize the sum of his own letters, while the second player tries to minimize the sum of the first player's letters.

(b) (2 points) List the base cases for the recurrence from part (a) and specify the value of V for them.

(c) (10 points) Using your recurrence relation and your base cases, describe an $O(n^2)$ time dynamic programming algorithm that computes $V[1, 2n]$ (the total value the first player wins if the second player plays optimal strategy). How would you use table V to play the game against a player that does not necessarily follow an optimal strategy?

[1] 3. (Bonus) How long did it take you to complete this assignment (not including any time you spent revising your notes before starting)?