CS 312: Algorithms **Fall 2018**

Homework 8

Your Name: _____

Collaborators and sources: _____

You may work in groups, but you must write solutions yourself. List collaborators on your submission.

If you are asked to design an algorithm, please provide: (a) either pseudocode or a precise English description of the algorithm, (b) an explanation of the intuition for the algorithm, (c) a proof of correctness, (d) the running time of your algorithm and (e) justification for your running time analysis.

Submission instructions. This assignment is due by noon on Wednesday, November 14 in Gradescope (as a pdf file). Please review the course policies on the course home page about Gradescope submissions.

1. **(20 points) Texting.** You are competing with your friends to type text messages on your smartphone as quickly as possible. Here are the rules: you use two thumbs for texting and they start out on the bottom left and bottom right keys of the keyboard. To type a character, you move either thumb from its current key to the key you need to press, and it takes time equal to the distance between the keys. You can assume the following:

   - The keyboard has keys labeled $\{1, 2, \ldots, k\}$ and there is a function `dist`(i,j) to calculate the distance between two keys i and j. (To visualize this, you may want imagine the digits 1 though 9 arranged on a standard numeric keypad).
   - Your left thumb starts on key a, and your right thumb starts on key b. (For example, on the 9-digit numeric keypad, $a = 7$ is the bottom left key, and $b = 9$ is the bottom right key.)
   - You can press any key with either thumb
   - Both thumbs can rest on the same key if necessary
   - The characters to by typed are $c_1 c_2 \cdots c_n$, where $c_i \in \{1, 2, \ldots, k\}$ is the ith key to push

   Design an algorithm that finds the fastest way to type the message. In other words, your algorithm needs to decide which thumb to use to type each character, and it should minimize the total distance moved by your two thumbs. Try to use $O(nk^2)$ time.

   Example. Imagine the 9-digit numeric keypad where your thumbs start at $a = 7$ and $b = 9$, with input message $c_1 c_2 c_3 = 589$. The solution "left, right, left" would look like this:

   0 Left/right thumbs start at 7/9
   (a) Left thumb moves from 7 to $c_1 = 5$. Time = `dist`(7, 5). Thumbs end at 5/9.
   (b) Right thumb moves from 9 to $c_2 = 8$. Time = `dist`(9, 8). Thumbs end at 5/8.
   (c) Left thumb moves from 5 to $c_3 = 9$. Time = `dist`(5, 9). Thumbs end at 9/8.

   Total time = `dist`(7, 5) + `dist`(9, 8) + `dist`(5, 9).

2. **(5 points extra credit) Chicken Wings.** *You only need to give the algorithm, the running time, and enough explanation so we can understand your solution — not a formal proof, etc. Partial credit is given sparingly for extra credit problems.*

   Suppose a restaurant allows you to buy chicken wings in n different quantities. For each i from 1 to n, you can order v_i wings for price w_i.

   Now, suppose you want to buy exactly V chicken wings. Write a dynamic programming algorithm to find the cheapest set of orders to buy exactly V wings. State its running time.

   Notes

- Assume each price w_i is an integer (e.g., the cost in cents).
- You can order the same quantity multiple times. E.g., you can order four orders of size 50 to buy 200 wings.
- You can assume there is always *some* combination of orders to buy exactly V wings.