CSC265 F18: Assignment 4

Due: November 7, by midnight

Guidelines: (read fully!!)

- Your assignment solution must be submitted as a *typed* PDF document. Scanned handwritten solutions, solutions in any other format, or unreadable solutions will **not** be accepted or marked. You are encouraged to learn the LaTeX typesetting system and use it to type your solution. See the course website for LaTeX resources. Solutions typed using LaTeX receive 2 bonus marks.
- Your submission should be no more than 6 pages long, in a single-column US Letter or A4 page format, using at least 9 pt font and 1 inch margins.
- To submit this assignment, use the MarkUs system, at URL `https://markus.teach.cs.toronto.edu/csc265-2018-09`
- This is a *group assignment.* This means that you can work on this assignment with *at most one other* student. You are *strongly encouraged* to work with a partner. Both partners in the group should work on and arrive at the solution together. Both partners receive the same mark on this assignment.
- Work on all problems together. For each problem, one of you should write the solution, and one should proof-read and revise it. The first page of your submission must list the *name*, *student ID*, and *UTOR email address* of both group members. It should also list, for each problem, which group member wrote the problem, and which group member proof-read and revised it.
- You **may not** consult any other resources except: your partner; your class notes; your textbook and assigned readings. *Consulting any other resource, or collaborating with students other than your group partner, is a violation of the academic integrity policy!*
- You may use any data structure, algorithm, or theorem previously studied in class, or in one of the prerequisites of this course, by just referring to it, and without describing it. This includes any data structure, algorithm, or theorem we covered in lecture, in a tutorial, or in any of the assigned readings. Be sure to give a *precise reference* for the data structure/algorithm/result you are using.
- Unless stated otherwise, you should justify all your answers using rigorous arguments. Your solution will be marked based both on its completeness and correctness, and also on the clarity and precision of your explanation.

Question 1. (14 marks)

In this problem you are given as input a set P of n *distinct* points in the plane. Each point p is specified by two *integer* coordinates, $p.x$ and $p.y$, which are the x- and the y-coordinates of p. We have $0 \le p.x \le N-1$ and $0 \le p.y \le N-1$, where $N = \Theta(n^{100})$. You can assume that P is given as a linked list of point objects.

Suppose that, together with P, you are also given an integer $t \ge 0$. Design a *randomized* algorithm which returns a linked list of all pairs of points (p_1, p_2) such that $p_1, p_2 \in P$, $p_1 \ne p_2$ and $d(p_1, p_2) \le t$. Here $d(p_1, p_2)$ is the *distance* between p_1 and p_2, given by the formula

$$d(p_1, p_2) = \sqrt{(p_1.x - p_2.x)^2 + (p_1.y - p_2.y)^2}.$$

Let s_r be, for any integer $r \ge 0$, the size of the set $\{(p_1, p_2) : p_1, p_2 \in P, p_1 \ne p_2, d(p_1, p_2) \le r\}$. Your algorithm should run in expected time $O(n + s_{\sqrt{2}t})$ and use $O(n)$ words of space for every input P, t.

Describe your algorithm using clear and precise English, and, optionally, using pseudocode. Justify its correctness and why it satisfies the time and space requirements above.

Question 2. (14 marks)

(The set up is the same as in the last Assignment)

Let T be a complete ternary tree, i.e. a tree in which all leaf nodes are at the same distance from the root, and every tree node except the leaves has exactly three children: a left, a middle, and a right child. Then T has 3^h leaves, where h is its height. Suppose that each leaf of T is given a value 0 or 1. Then we determine values for the internal nodes of T as follows: the value of an internal node u of T equals 0 if at least two of its children have value 0, and its value equals 1 otherwise.

Assume that the values of the leaves, numbered from left to right, are given in an array $A[1\,..\,3^h]$. Give a *randomized* algorithm which computes the value of the root of T so that in the worst case (over the choice of A) the *expected number* of entries of A queried by the algorithm is at most C^h, where $C < 3$ is a constant. Give a precise value for C and justify it, and also justify the correctness of your algorithm.