CMSC 420: Homework 4: kd-Trees

- Handed out: 4/30 Due: 5/9 before class starts. Late homeworks will not be accepted.
- Please answer in the space provided
- (Optional) Please write (if true) in your handwriting and sign: "I pledge on my honor that I have not given or received any unauthorized assistance on this assignment or any previous homework." Note that you are permitted and encouraged to discuss programming assignments, not homework.

1. (Building kd-trees statically). Assume that the points (35, 90), (70, 80), (10, 75) (80, 40), (50, 90), (70, 30), (90, 60), (50, 25), (25, 10), (20, 50), and (60, 10) are given in advance, so you can pre-process them in a sorted manner. The following conventions apply: Sorting is in descending order; while sorting on a coordinate, break ties by considering the other coordinate; the median of a sorted list $A_i, i = 0 \ldots 2k - 1, k > 0$ is $k$; splitting on $x$ corresponds to $cd == 0$;

(a) Complete the following pseudocode used in buildTree(). $S[0]$ is an array containing points sorted on $x$. $S[1]$ is an array containing points sorted on $y$.

```c
void leftPoints(S, cd) {
    int n = S.length;
    m = median(S); // S now does NOT contain m
    if (cd) {
        for (i = 0; i < n/2; i++) left[1][i] = S[1][i];
        a = qsort(left[1], x); // a is a sorted on x now
        for (i = 0; i < n/2; i++) left[0][i] = a[i];
    } else {
```

(b) Draw the tree that would be produced when buildTree() is called on the given points. Circle an internal node and write the coordinates of the relevant point inside the circle. Also label the node on its left by stacking two arrays one on top of each other.
(c) What is the big-Oh time complexity of `leftPoints()` using the procedure in Step (a)? Explain briefly.

(d) Update `leftPoints()` so that `buildTree()` runs in \(O(n \log n)\) time.

2. (Nearest neighbor, no Priority Q version) In the example shown, the distance between the query point \((40, 50)\) and two nodes is *not* computed. Move the query point \((40, 50)\) to an alternate location so that all 11 distances must be necessarily computed by `process()`. State the order in which the nodes are processed. What is the alternate location? (If no such location is possible, state that this is the case).

3. (Optional) As in question 2, but using a priority queue.