HW 5

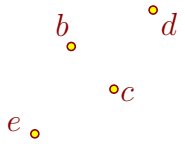
Due on **Wednesday, October 5, 2022 at 9am CST**

CS/ECE 374A: Intro. Algorithms & Models of Computation, Fall 2022 Version: **2.24**

Submission instructions as in previous homeworks.

9 (100 PTS.) Incomparable, that's what you are.

Two points $p = (x, y)$ and $p' = (x', y')$ are ***incomparable*** if $x < x'$ and $y' < y$, or alternatively, $x' < x$ and $y < y'$. Thus, in the following example, the only incomparable pair is $\{b, c\}$:



9.A. (50 PTS.) You are given as input two sequences of points in the plane: $S = s_1, s_2, \ldots, s_n$ and $T = t_1, t_2, \ldots, t_n$, such that $x(s_i) < \alpha < x(t_j)$, for all i, j, where α is some real number. Furthermore, assume that the points of S are sorted in increasing y value, and so are the points of T.

Let $I(S, T)$ denote the number of incomparable pairs $\{s, t\}$ with $s \in S$ and $t \in T$. Describe an algorithm, as fast as possible, that computes and outputs $I(S, T)$.

9.B. (50 PTS.) Given a set P of n points in the plane, let $I(P)$ denote the number of incomparable pairs of points of P. Describe a divide-and-conquer algorithm, using the algorithm in the first part, that computes and outputs $I(P)$. (The input P is not sorted in any way.) Your algorithm needs to be as fast as possible.

You can assume that all input points have unique x and y coordinates.

(Hint: Merge sort.)

10 (100 PTS.) Sort in bulk.

You are given n distinct items in an array A. You are also a given a procedure **BulkSort**(B) that sorts any array B with at most k such items. Note, that you can not compare two items directly (but you can sort them, by calling **BulkSort** on an array containing only these two items).

10.A. (50 PTS.) Describe a divide and conquer algorithm that sorts A using calls to **BulkSort**. Your algorithm should perform (asymptotically) as few calls to **BulkSort** as possible. What is the worst case number of calls to **BulkSort** that your algorithm performs as a function of n and k? (We ignore the cost of other operations - such as copying or moving things around. We also assume that given two items, or copies of two items, we can decide if they are the same item.)

(Getting the optimal algorithm for this part is difficult – an answer that is optimal up to a polylog factor is acceptable here.)

10.B. (50 PTS.) Describe a recursive algorithm that performs a minimum number of calls to **BulkSort** and outputs the median item in A. Specifically, given $A[1 \ldots n]$ and a parameter t, it outputs the item of rank t in the sorted order of the items of A. What is the number of calls your algorithm performs to **BulkSort** as a function of n and k?

(Stating a recurrence on the number of calls to **BulkSort**, together with a solution of this recurrence is enough – a formal proof of the correctness of the solution to the recurrence is not required since it is very painful.)

For sanity check purposes, think about how many calls your algorithm performs to **BulkSort** (for both parts) if $k = n/2$ or $k = \sqrt{n}$.