Assignment 6: Divide and Conquer

Due: 11th week

Note: You are strongly encouraged to learn how to solve problems for all sections for the quiz and CSE 203 final.

Assignment for Section B2

Counting Inversions: This problem arises in the analysis of rankings. Consider comparing two rankings. One way is to label the elements (books, movies, etc.) from *1* to *n* according to one of the rankings, then order these labels according to the other ranking, and see how many pairs are "out of order".

We are given a sequence of n distinct numbers $a_1, ..., a_n$. We say that two indices $i < j$ form an inversion if $a_i > a_j$ that is if the two elements a_i and a_j are "out of order". Provide a divide and conquer algorithm to determine the number of inversions in the sequence $a_1, ..., a_n$ in time O(*n log n*). You can modify merge sort to count during merging (Reference: Kleinberg & Tardos – Section 5.3).

Input:

First line of the input file will contain the size of the array, *n* and second line will contain *n* integers separated by spaces. For example:

10

5 2 10 8 1 9 4 3 6 7

Output:

Number of inversions in the array. Output for the above input should be

22

Assignment for Section B1

Quicksort using medians (D&C within a D&C): You will implement the quicksort algorithm (Reference: Cormen *et al.* – Section 7.1). However, instead of using the last element as the pivot, you will first find the median using a randomized divide and conquer algorithm (Reference: Dasgupta *et al.* – Section 2.4) and use that as the pivot. The expected running time of median finding should be O(*n*) giving an expected O(*n log n*) algorithm for quicksort.

Input:

First line of the input file will contain the size of the array, *n* and second line will contain *n* integers separated by spaces. For example:

10

5 2 10 8 1 9 4 3 6 7

Output:

The median and the sorted array. Output for the above input should be

5 or 6

1 2 3 4 5 6 7 8 9 10

Assignment for Section A2

Skyline problem: Consider a budget traveler looking to stay in a hotel by the Cox's Bazar beach. Naturally, hotels near the beach tend to be more expensive than ones a bit further away. Our traveler will only consider a hotel if it is either closer or cheaper than each of the other hotels. Formally, we say a point (x_1,y_1) dominates a point (x_2,y_2) if $x_1 \geq x_2$ and $y_1 \geq y_2$. Given a set of points, provide a divide and conquer algorithm to find all points that do *not* dominate any other point. Your algorithm should run in O(*n log n*) time. (Ref: http://www.cs.sfu.ca/~ssa121/personal/spring08/705/dnc.pdf)

Input:

First line of the input file will contain the number of points, *n* followed by two positive integers per line giving co-ordinates of the points. For example:

11
9 2
1 8
3 7
10 5
8 5
6 8
2 5
4 4
11 7
7 3
5 6

Output: The co-ordinates of the non-dominating points.

9 2
1 8
4 4
2 5

Assignment for Section A1

Closest pair of points: You are given *n* points in the plane (i.e. two dimensions). Provide a divide and conquer algorithm to determine pair of points that are closest to each other in time O(*n log n*) (Reference: Kleinberg & Tardos – Section 5.4).

Input:

First line of the input file will contain the number of points, *n* followed by two numbers per line (may not be integers) giving co-ordinates of the points. For example:

```
5
0 0
-4 1
-7 -2
4 5
1 1
```

Output: The co-ordinates of the two closest points and the smallest distance.

```
0 0
1 1
1.414214
```

- Contact Atif Hasan Rahman if you have queries