BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY

BUET

CSE 204 Lab Report

Course No. : CSE 204

No of Assignment: 08

Name of Assignment: Finding the second closest points in a 2D plane using divide and conquer approach

Section : A2

Date of Submission : 18/06/2021

Submitted by : Mahdee Mushfique Kamal

ID : 1805032

Complexity analysis: The complexity of the algorithm is O(NlogN)

In the base case, where the number of points is 2 or 3. It finds the closest and 2nd closest points in a constant time or O(1)
So, T(1) = O(1) , T(2) = O(1) and T(3) = O(1)

The algorithm divides the problem into 2 subproblems of size n/2. Solve them and combine the solutions.
The **compareAndAdjust()** function takes O(1) time.
To add points in **pointsInStrip** , we iterate the whole arrY array. So complexity is O(n).
while looking at the points in the strip, there can be at most n points. So, the number of operations is 7n. That means the time complexity of the merge step is O(n).
T(n) = 2T(n/2) + θ(n)
There will be logn number of merges. So the total complexity is n*logn.

```
else if(r-l+1 > 3){
    int mid = l + (r-l)/2;
    Pair *from_left = firstAndSecondShortest(arrX , arrY,  l , mid);
    Pair *from_right = firstAndSecondShortest(arrX, arrY,  mid+1, r);

    compareAndAdjust(ans, from_left[0]);
    compareAndAdjust(ans, from_left[1]);
    compareAndAdjust(ans, from_right[0]);
    compareAndAdjust(ans, from_right[1]);

    double dlt = ans[1].dist;

    vector<Point> pointsInStrip;
    for(int i=0; i<=n; i++){
        if(arrY[i].x >= arrX[mid].x - dlt && arrY[i].x <= arrX[mid].x +dlt ){
            pointsInStrip.push_back(arrY[i]);
        }
    }
    int len = pointsInStrip.size();

    if(len > 1){
        for(int i=0;i< len;i++){
            for(int j=1; j < 8; j++){
                if(i+j < len){
                    compareAndAdjust(ans, Pair(pointsInStrip[i], pointsInStrip[i+j]));
                }
            }
        }
    }
}
return ans;
```