Assignment 06 | Advance Algorithms CE-092

Assignment submission for Advance Algorithms subject week 6.

nevilparmar24@gmail.com

Task 1:

Write a Program to check whether two line segments intersect or not.

Code:

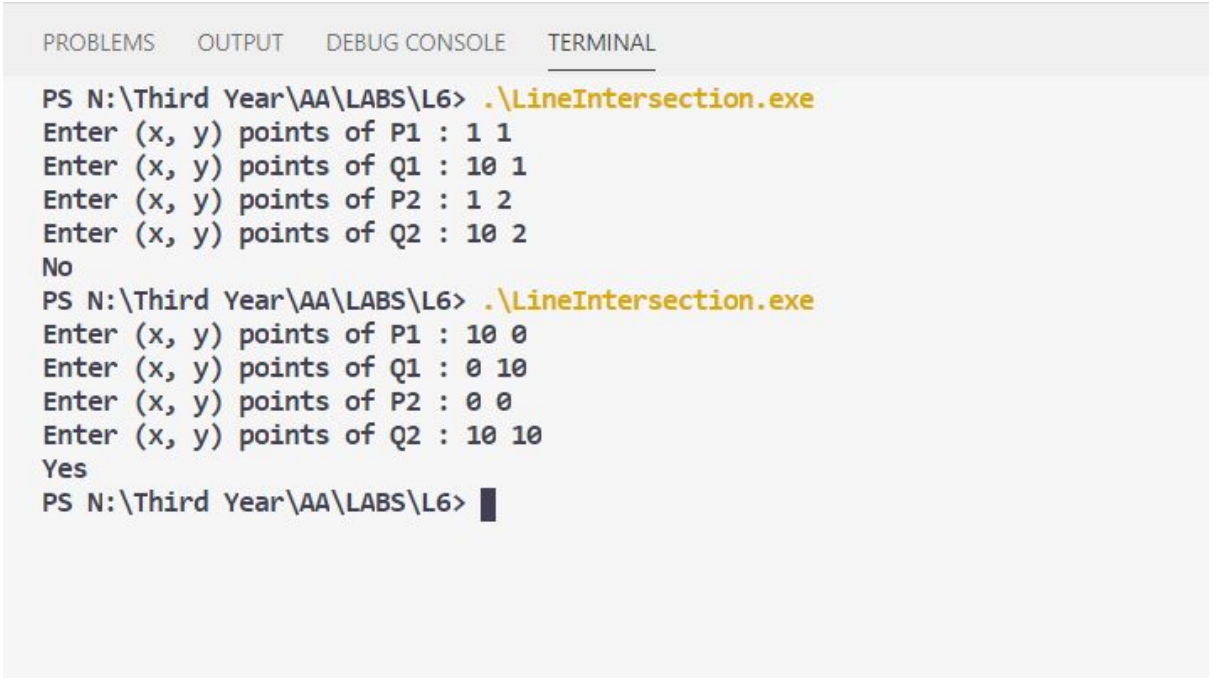
```
/*
* @Author: nevil
* @Date:   2020-08-14 16:10:50
* @Last Modified by:   nevil
* @Last Modified time: 2020-09-07 12:34:01
*/
#include <bits/stdc++.h>
using namespace std;

struct Point
{
    int x;
    int y;
};

bool onSegment(Point p, Point q, Point r)
{
    if (q.x <= max(p.x, r.x) && q.x >= min(p.x, r.x) &&
        q.y <= max(p.y, r.y) && q.y >= min(p.y, r.y))
```

```
    return true;

    return false;
}

int orientation(Point p, Point q, Point r)
{
    int val = (q.y - p.y) * (r.x - q.x) -
              (q.x - p.x) * (r.y - q.y);

    if (val == 0) return 0; // colinear

    return (val > 0)? 1: 2; // clock or counterclock
wise
}

bool doIntersect(Point p1, Point q1, Point p2, Point
q2)
{
    int o1 = orientation(p1, q1, p2);
    int o2 = orientation(p1, q1, q2);
    int o3 = orientation(p2, q2, p1);
    int o4 = orientation(p2, q2, q1);

    // General case
    if (o1 != o2 && o3 != o4)
        return true;

    // Special Cases
    // p1, q1 and p2 are colinear and p2 lies on
segment p1q1
    if (o1 == 0 && onSegment(p1, p2, q1)) return true;
```

```
    // p1, q1 and q2 are colinear and q2 lies on
segment p1q1
    if (o2 == 0 && onSegment(p1, q2, q1)) return true;

    // p2, q2 and p1 are colinear and p1 lies on
segment p2q2
    if (o3 == 0 && onSegment(p2, p1, q2)) return true;

    // p2, q2 and q1 are colinear and q1 lies on
segment p2q2
    if (o4 == 0 && onSegment(p2, q1, q2)) return true;

    return false; // Doesn't fall in any of the above
cases
}

int main()
{
    struct Point p1, q1, p2, q2;

    cout << "Enter (x, y) points of P1 : " ;
    cin >> p1.x >> p1.y;

    cout << "Enter (x, y) points of Q1 : " ;
    cin >> q1.x >> q1.y;

    cout << "Enter (x, y) points of P2 : " ;
    cin >> p2.x >> p2.y;

    cout << "Enter (x, y) points of Q2 : " ;
    cin >> q2.x >> q2.y;
```

```
    doIntersect(p1, q1, p2, q2)? cout << "Yes\n": cout
<< "No\n";

    return 0;
}
```

Output:



Task 2:

Write a Program to find the closest pair of points using Brute Force approach.

Code:

```
/*
* @Author: nevil
* @Date:   2020-09-07 12:31:12
* @Last Modified by:   nevil
* @Last Modified time: 2020-09-07 21:18:51
*/
#include <bits/stdc++.h>
using namespace std;
```

```
struct Point
{
    int x;
    int y;
};
float dist(Point p1, Point p2)
{
    return sqrt( (p1.x - p2.x)*(p1.x - p2.x) +
                 (p1.y - p2.y)*(p1.y - p2.y)
           );
}
float bruteForce(Point P[], int n)
{
    float min = FLT_MAX;
    for (int i = 0; i < n - 1; ++i)
        for (int j = i+1; j < n; ++j)
            if (dist(P[i], P[j]) < min)
                min = dist(P[i], P[j]);
    return min;
}
int main()
{
    int n;
    cout << "Enter the total number of points : ";
    cin >> n; // number of points
    struct Point points[n];
    cout << "Enter " << n << " points in the (x , y)
format" << endl;
    int i;
    for(i = 0 ; i < n; i++)
    {
```

```
        cin >> points[i].x ;
        cin >> points[i].y;
    }
    cout << "The smallest distance is " <<
bruteForce(points, n);
    return 0;
}
```

Output :

```
PS N:\Third Year\AA\LABS\L6> g++ .\closestPairBrute.cpp
PS N:\Third Year\AA\LABS\L6> .\a.exe
Enter the total number of points : 6
Enter 6 points in the (x , y) format
2 3
12 30
40 50
5 1
12 10
3 4
The smallest distance is 1.41421
PS N:\Third Year\AA\LABS\L6>
```

Complexity :

Line Intersection :

The complexity of the above program is O(1). All the operations performed are of constant complexity therefore the overall time complexity is also constant.

Closest Pair Of Points :

The complexity of the above program is O(N^2). The time needed to find the distance is constant , but since we find the distance between all the possible pairs so the overall complexity of the brute force becomes O(N^2).

Nevil Parmar
CE-092
https://nevilparmar.me