

Lab-10

Name: E. Jayanth Reddy

Reg.no:19BCE7548

1Q. Nearest_neighbor.

Java code:

```
import java.io.*;
import java.util.*;
import java.util.stream.*;

import static java.lang.Math.*;

public class Main {

    static class Point implements Comparable<Point> {
        long x, y;

        public Point(long x, long y) {
            this.x = x;
            this.y = y;
        }

        public int compareTo(Point o) {
```

```

        return o.y == y ? Long.signum(x - o.x) : Long.signum(y -
o.y);
    }
}

```

```

static double dist(Point p1, Point p2) {
    return Math.sqrt((p1.x - p2.x) * (p1.x - p2.x) + (p1.y -
p2.y) * (p1.y - p2.y));
}

```

```

static double stripMin(List<Point> strip, int stripLength ,
double d) {

```

```

    strip = strip.stream().filter(i -> i != null)
        .sorted((p1, p2) -> Long.compare(p1.y, p2.y))
        .collect(Collectors.toList());

```

```

    for (int i = 0; i < strip.size(); i++) {
        for (int j = i + 1; j < stripLength &&
Math.abs(strip.get(i).y - strip.get(j).y) < d; j++){

```

```

        d = Math.min(d, dist(strip.get(i), strip.get(j)));
    }
}

return d;
}

static double minDist(Point[] points, int left, int right) {

    if (left >= right) return Double.POSITIVE_INFINITY;

    if (right - left == 1) {
        return dist(points[0], points[1]);
    }

    double d = 0;

    int mid = left + (right - left) / 2;
    Point midPoint = points[mid];

    double d1 = minDist(points, left, mid);
    double d2 = minDist(points, mid+1, right);

```

```
d = Math.min(d1,d2);
```

```
List<Point> strip = new ArrayList<>();
```

```
for (int i = left; i <= right; i++) {  
    if (Math.abs(points[i].x - midPoint.x) < d) {  
        strip.add(points[i]);  
    }  
}
```

```
return Math.min(d, stripMin(strip, strip.size(), d));  
}
```

```
static double minimalDistance(int[] x, int y[]) {  
    double ans = Double.POSITIVE_INFINITY;  
    Point[] points = new Point[x.length];  
    for (int i = 0; i < x.length; i++) {  
        points[i] = new Point(x[i], y[i]);  
    }
```

```
// SORT BY x
Arrays.sort(points, (p1, p2) -> Long.compare(p1.x, p2.x));

ans = minDist(points, 0 , points.length -1);
return ans;
}
```

```
public static void main(String[] args) throws Exception {
    reader = new BufferedReader(new
InputStreamReader(System.in));
    writer = new PrintWriter(System.out);
    int n = nextInt();
    int[] x = new int[n];
    int[] y = new int[n];
    for (int i = 0; i < n; i++) {
        x[i] = nextInt();
        y[i] = nextInt();
    }
    System.out.println(minimalDistance(x, y));
    writer.close();
}
```

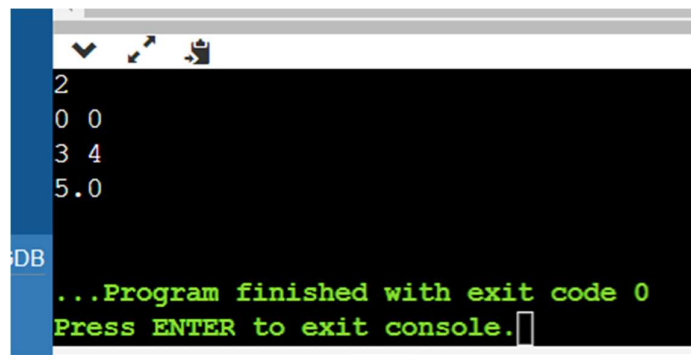
```
static BufferedReader reader;  
static PrintWriter writer;  
static StringTokenizer tok = new StringTokenizer("");
```

```
static String next() {  
    while (!tok.hasMoreTokens()) {  
        String w = null;  
        try {  
            w = reader.readLine();  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
        if (w == null)  
            return null;  
        tok = new StringTokenizer(w);  
    }  
    return tok.nextToken();  
}
```

```
static int nextInt() {
```

```
        return Integer.parseInt(next());  
    }  
}
```

Output:



```
2  
0 0  
3 4  
5.0  
...Program finished with exit code 0  
Press ENTER to exit console.
```