Distance between two polygons

by mrtempo

Given two convex polygons P and Q. Determine the distance between them.

The distance between two convex polygons P and Q is defined as the length of the shortest distance between points p and q, such that point p lies inside polygon P, and point q lies inside polygon Q.

A point p is said to be inside polygon P if point p is inside polygon P or if it is on any of its edges.

Input Format

The first line contains two integers N and M indicating the number of vertices of polygons P and Q respectively.

Each of the next N lines contains two integers X_n and Y_n specifying the vertices of polygon P.

Each of the next M lines contains two integers X_m and Y_m specifying the vertices of polygon Q.

Constraints

- $3 \le N, M \le 5000$
- $-10^6 \le X_n, Y_n \le 10^6$
- $-10^6 \le X_m, Y_m \le 10^6$
- X_n, Y_n will be pairwise distinct, i.e. polygon P will not contain duplicate points.
- X_m, Y_m will be pairwise distinct, i.e. polygon Q will not contain duplicate points.
- Polygons might intersect, and might not be strictly convex. It is possible that 3 consecutive points could be collinear in any polygon.
- Each polygon will have a strictly positive area.
- In each polygon, points could either be sorted clockwise or anticlockwise.

Output Format

A single line with the distance between the two polygons.

Print the output as precisely possible. Solution will be considered correct within 10^{-6} of error.

Sample Input 0

```
4 4
2 1
10 1
10 5
2 5
15 10
20 10
20 15
15 15
```

Sample Output 0

```
7.071067811865
```

Contest ends in 19 hours

Submissions: 112
Max Score: 100
Difficulty: Medium

Rate This Challenge:

More

Current Buffer (saved locally, editable) Java 8

```java
import java.io.*;
import java.math.*;
import java.security.*;
import java.text.*;
import java.util.*;
import java.util.concurrent.*;
import java.util.function.*;
import java.util.regex.*;
import java.util.stream.*;
import static java.util.stream.Collectors.joining;
import static java.util.stream.Collectors.toList;

public class Solution {

    // Complete the solve function below.
    static double solve(List<List<Integer>> p, List<List<Integer>> q) {


    }

    public static void main(String[] args) throws IOException {
        BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(System.in));
        BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT_PATH")));

        String[] nm = bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");

        int n = Integer.parseInt(nm[0]);

        int m = Integer.parseInt(nm[1]);

        List<List<Integer>> p = new ArrayList<>();

        IntStream.range(0, n).forEach(i -> {
            try {
                p.add(
                    Stream.of(bufferedReader.readLine().replaceAll("\\s+$", "").split(" "))
                        .map(Integer::parseInt)
                        .collect(toList())
                );
            } catch (IOException ex) {
                throw new RuntimeException(ex);
            }
        });

        List<List<Integer>> q = new ArrayList<>();

        IntStream.range(0, m).forEach(i -> {
            try {
                q.add(
                    Stream.of(bufferedReader.readLine().replaceAll("\\s+$", "").split(" "))
                        .map(Integer::parseInt)
                        .collect(toList())
                );
            } catch (IOException ex) {
                throw new RuntimeException(ex);
            }
        });

        double reslut = solve(p, q);

        bufferedWriter.write(String.valueOf(result));
        bufferedWriter.newLine();

        bufferedReader.close();
        bufferedWriter.close();
    }
}
```