Laboratory work #3

Student: CAO Xinyang  
Student ID: 20321308  
Timus Name: hduads2022\_20321308

Mail: c.x\_yang@foxmail.com

Problem #1444

Screenshot from Timus:



Explanation of algorithm:

Set the first point as the origin and find the Angle between the other points and the origin. Order the points according to the Angle between them. Then walk through the points in descending order of Angle. If the included angles are equal, the order is from the smallest to the largest from the origin. If there is a corner greater than 180 degrees, change the order (find the point with a corner greater than 180 degrees by using the variable s).

Computational complexity of algorithm:

F(N) = N + N^2 + N + N = 3N + N^2

T(N) = O(N^2)

Source code:

import java.io.\*;

public class App {

public static void main(String[] args) throws Exception {

StreamTokenizer in = new StreamTokenizer(new BufferedReader(new InputStreamReader(System.in)));

PrintWriter out = new PrintWriter(new OutputStreamWriter(System.out));

in.nextToken();

int n = (int) in.nval;

int[] x = new int[n];

int[] y = new int[n];

int[] index = new int[n];

double[] atg = new double[n];

in.nextToken();

int x0 = (int) in.nval;

in.nextToken();

int y0 = (int) in.nval;

for (int i = 1; i < n; i++) {

in.nextToken();

x[i] = (int) in.nval - x0;

in.nextToken();

y[i] = (int) in.nval - y0;

index[i] = i;

atg[i] = Math.atan2(y[i], x[i]);

}

int tempPosition = 0;

int len = n - 1;

for (int i = 1; i < n - 1; i++) {

boolean flag = true;

for (int j = 1; j < len; j++) {

int x1 = x[index[j]];

int x2 = x[index[j + 1]];

Double d = atg[index[j + 1]] - atg[index[j]];

if (d < 0 || d == 0 && x1 \* x1 > x2 \* x2) {

int temp = index[j];

index[j] = index[j + 1];

index[j + 1] = temp;

flag = false;

tempPosition = j;

}

}

len = tempPosition;

if(flag)

break;

}

int s = 0;

for (int i = 0; i < n - 1; i++) {

int x1 = x[index[i]];

int y1 = y[index[i]];

int x2 = x[index[i + 1]];

int y2 = y[index[i + 1]];

int dx = x1 \* y2 - x2 \* y1, d = x1 \* x2 + y1 \* y2;

if (dx < 0 || dx == 0 && d < 0) {

s = index[i];

break;

}

}

System.out.println(n);

System.out.println(1);

for (int i = 0; i < n - 1; i++) {

int ans = index[(s + i) % (n - 1) + 1] + 1;

out.println(ans);

}

out.flush();

}

}