МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное

образовательное учреждение высшего образования

«Санкт-Петербургский национальный исследовательский университет

информационных технологий, механики и оптики»

**ФАКУЛЬТЕТ СИСТЕМ УПРАВЛЕНИЯ И РОБОТОТЕХНИКИ**

**ЛАБОРАТОРНАЯ РАБОТА №1**

**(1207)**

Выполнили:

Хюинь Тан Куонг(336231)

Преподаватель:

Tropchenko Andrey Aleksandrovich

**A picture containing shape

Description automatically generated**

Санкт-Петербург, 2023

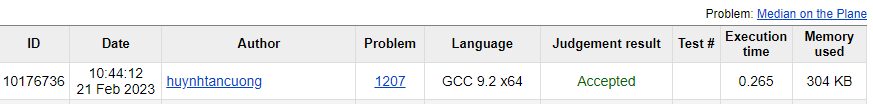
# 1005.

## Problem description

Graphical user interface, text, application

Description automatically generated

## Timus system acceptance:



## Code:

#include <stdio.h>

#include <math.h>

#include <stdlib.h>

#include <inttypes.h>

struct line

{

    // line: ax + by + c = 0

    int64\_t a, b, c;

};

struct point

{

    // point: (x, y)

    int64\_t x, y;

};

struct line get\_line(struct point p1, struct point p2) {

    struct line d;

    d.a = p1.y - p2.y;

    d.b = p2.x - p1.x;

    d.c = p1.x\*p2.y - p2.x\*p1.y;

    return d;

}

float line\_point\_distance(struct line d, struct point M) {

    // float dis = (d.a\*M.x + d.b\*M.y + d.c)/(sqrt(d.a\*d.a + d.b\*d.b));

    float dis = (d.a\*M.x + d.b\*M.y + d.c);

    return dis;

}

int is\_equal\_sets(struct point p[], size\_t size, struct line d) {

    int positive\_counter = 0;

    int negative\_counter = 0;

    for (size\_t i = 0; i<size; i++) {

        if (line\_point\_distance(d, p[i]) > 0) positive\_counter++;

        if (line\_point\_distance(d, p[i]) < 0) negative\_counter++;

        // bypass distance == 0

    }

    return positive\_counter == negative\_counter;

}

int main() {

    int N;

    scanf("%d", &N);

    struct point \*points;

    points = (struct point \*) malloc(sizeof(struct point)\*N);

    for (size\_t i = 0; i<N; i++) {

        scanf("%" SCNd64 " %" SCNd64, &points[i].x, &points[i].y);

    }

    for (size\_t i = 0; i<N; i++) {

        for (size\_t j=i+1; j<N; j++) {

            struct line d = get\_line(points[i], points[j]);

            if (is\_equal\_sets(points, N, d)) {

                printf("%d %d", i+1, j+1);

                return 0;

            }

        }

    }

    return 0;

}

## Output:

// Input

4

0 0

1 0

0 1

1 1

// Output

1 4

## Algorithm description:

- The main idea of the algorithm is to brute force all possible the pair of point and check if they are the line which separates the plane or not.

- To check the condition, we calculate the distance between points and a line. If the point is on the left of the line, the distance value > 0. And if the point is on the right side of the line, the distance value < 0.

- So as to speed up the program, I cut down the division operator.