Физико-математический лицей № 239

**«Дваждытреугольное множество точек»**

Отчет о домашней работе по информатике

Работу выполнил

ученик 10-2 класса

Мамаев Фёдор

Санкт-Петербург

2021

1. **Постановка задачи**

Множество точек на плоскости назовем дваждытреугольным, если каждая точка этого множества является вершиной хотя бы двух правильных треугольников, построенных по точкам множества. Определить, удовлетворяет ли заданное множество точек этому свойству.

1. **Уточнение входных и выходных данных и ограничений на них**
   1. Входные данные

Входные данные – координаты точек. Ограничений на них нет, но точки с координатой больше 10 или меньше -10 не будут видны на экране. На количество точек нет ограничения. Множество точек также можно загрузить из файла и сохранить в файл, а также добавить случайные точки с координатами от -10 до 10.

2.2. Выходные данные

Программа должна определить, является ли данное множество точек дваждытреугольным и вывести ответ («ДА» или «НЕТ»), а также построить по точкам множества все возможные равносторонние треугольники.

1. **Выбор метода решения**
   1. Анализ исходных данных и выбор используемой структуры данных

Для каждой точки мы храним 2 ее параметра: расстояние по горизонтальной и вертикальной оси от начала координат. Также у точки есть параметр, который необходим для решения поставленной задачи: количество равносторонних треугольников, в которые точка входит. Мы будем хранить и список самих треугольников, чтобы вывести их на экран. Также нужно указать точность вычислений (количество учитываемых знаков после запятой), потому что точные координаты вершин равносторонних треугольников могут быть иррациональными числами. В нашем проекте точки и треугольники описываются классами, а точность – числом (количеством знаков после запятой).

Данные вводятся пользователем в текстовые поля справа от окна, где будут выводиться точки или загружаются из файла, после чего нажимается кнопка «Решить».

* 1. Выбор метода решения

Считываем координаты точек и сохраняем их в список точек. Если точек меньше 4, то по ним в принципе нельзя построить больше одного треугольника. Если нет, то последовательно перебираем все тройки точек, находим по теореме Пифагора расстояния между ними, и если расстояния равны с учетом n знаков после запятой, то запоминаем этот треугольник и прибавляем 1 к параметру точки, определяющему, во сколько треугольников она входит. После того, как мы перебрали все тройки точек, мы проходим по всем точкам в списке. Если хоть одна из них входит в меньше, чем 2 треугольников, то ответ к задаче отрицательный, если нет – положительный.

1. **Составление алгоритма**
   1. Обобщённая блок-схема алгоритма

Начало

Выяснение, все ли точки входят в 2 или более треугольников

Проверка, является ли тройка правильным треугольником

Чтение входных данных. Сбор статистики

Вывод ответа на экран

Перебор всех возможных троек точек

* 1. Блок-схема алгоритма

Начало

points – список точек precision - точность

points.size < 3?

НЕТ

ДА

Вернуть НЕТ

a = 0

b = 1

c = 2

while a < points.size – 3:

distance(pt1, pt2) = round(√((pt1.x – pt2.x)2 + (pt1.y – pt2.y)2), precision)

c += 1

если c >= n:

b += 1

c = b + 1

если b >= n + 2:

a += 1

b = a + 1

c = b + 1

distance(points[a], points[b]) = distance(points[a], points[c]) = distance(points[b], points[c])?

ДА

НЕТ

points[a].num\_of\_triangles += 1

points[b].num\_of\_triangles += 1

points[c].num\_of\_triangles += 1

i = от 0 до points.size - 1

Вернуть ДА

ДА

Вернуть НЕТ

НЕТ

points[i].num\_of\_triangles > 2?

1. **Листинг программы**

// Функция определения расстояния между 2 точками с учётом точности

private double dist(Point p1, Point p2)

{

return Math.round(Math.sqrt((p1.x - p2.x) \* (p1.x - p2.x) + (p1.y - p2.y) \* (p1.y - p2.y)) \* Math.pow(10, precision) \* 2) / Math.pow(10, precision) / 2;

}

if (points.size() < 3) return false;

for (int i = 0; i < points.size(); i++)

{

// Устанавливаем у всех точек параметр num\_of\_triangles в 0

Point d = points.get(i);

d.num\_of\_triangles = 0;

points.set(i, d);

}

// перебираем все возможные тройки точек

int a = 0, b = 1, c = 2;

while (a <= points.size() - 3) {

// если точки не совпадают...

if (points.get(a) != points.get(b) && points.get(a) != points.get(c)) {

// если расстояния между точками одинаковые (с учётом точности)...

double t = dist(points.get(a), points.get(b));

if (t == dist(points.get(a), points.get(c)) && t == dist(points.get(b), points.get(c))) {

// ... то мы увеличиваем у каждой точки параметр

// num\_of\_triangles на единицу

Point pt = points.get(a);

pt.num\_of\_triangles++;

points.set(a, pt);

pt = points.get(b);

pt.num\_of\_triangles++;

points.set(b, pt);

pt = points.get(c);

pt.num\_of\_triangles++;

points.set(c, pt);

// и добавляем треугольник в список треугольников,

// которые потом выведем на экран для

// визуализации решения

triangles.add(new Triangle(points.get(a), points.get(b), points.get(c)));

}

}

// переходим к следующей тройке

c += 1;

if (c >= points.size()) {

b += 1;

c = b + 1;

if (b >= points.size() - 1) {

a += 1;

b = a + 1;

c = b + 1;

}

}

}

// перебираем все точки

for (int i = 0; i < points.size(); i++) {

// если какая-то точка входит в менее чем 2 треугольника...

if (points.get(i).num\_of\_triangles < 2) {

// то ответ - НЕТ

return false;

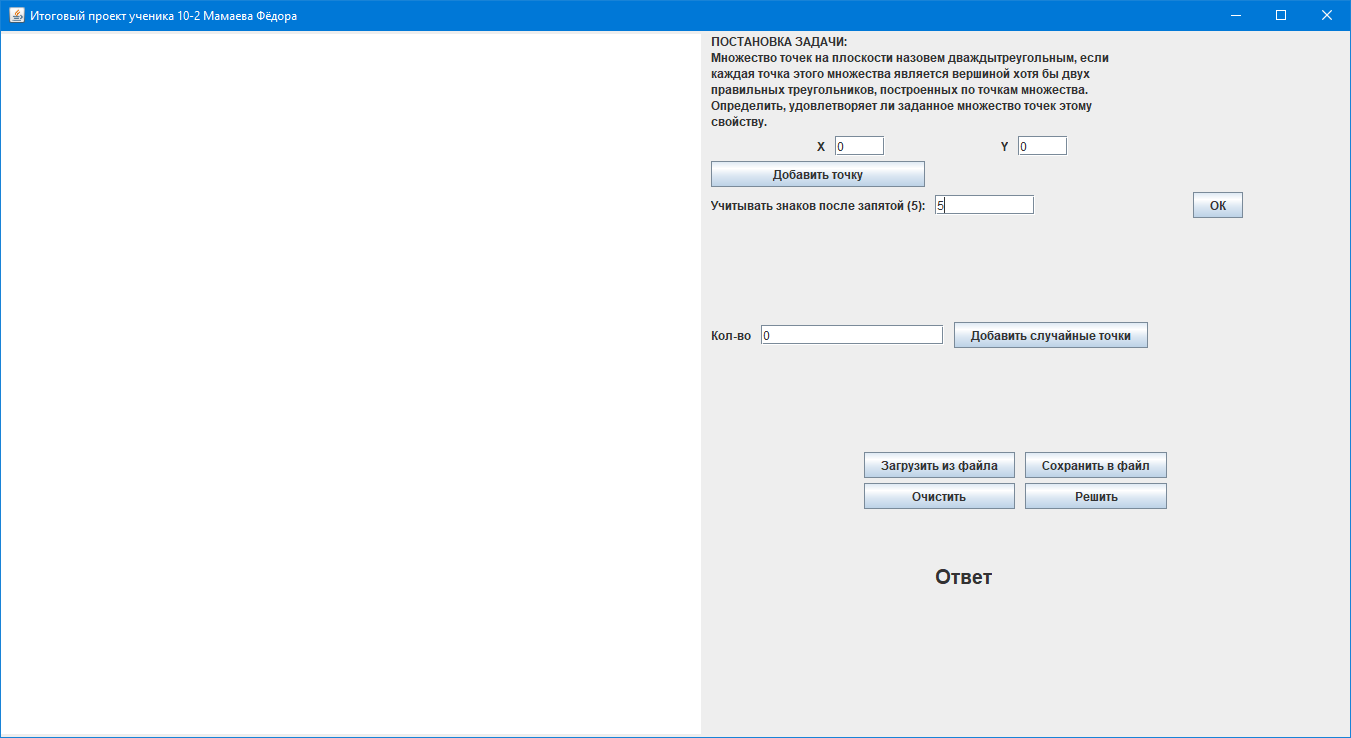
}

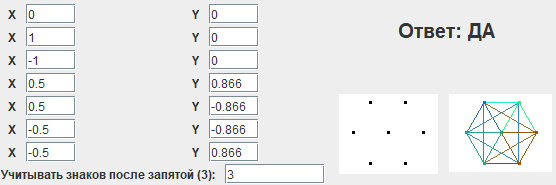
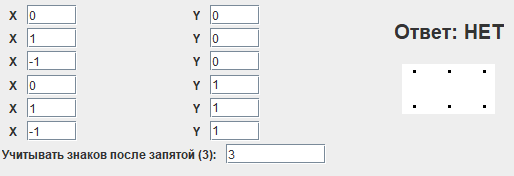
}

// если все точки входят в 2 или более треугольников, то ответ - ДА

return true;

1. **Пример работы программы**



** **