**Государственное бюджетное общеобразовательное учреждение "Президентский физико-математический лицей № 239"**

**«Пересечения прямоугольников»**

Годовой проект по информатике

Автор: Неуймина Антонина, 10-7 класс

Санкт-Петербург

2021

**Постановка задачи**

На плоскости задано множество прямоугольников. Найти такую пару пересекающихся прямоугольников, что длина отрезка, проведенного от одной точки пересечения этих двух прямоугольников до другой, максимальна. Если прямоугольники имеют более двух точек пересечения, выбирать среди них такую пару, расстояние между которыми максимально. (Задача №12)

В качестве ответа:

выделить эту пару прямоугольников,

нарисовать отрезок между найденными точками пересечения.

Если прямоугольники пересекаются, то точек их пересечения 2 или 4 (иногда и 3, в случаях касания) штуки. Значит, нужно пересечь все прямоугольники между собой, посмотреть отрезки из точек их пересечения, измерить их длину, сравнить и выбрать самый длинный отрезок.

**Уточнение исходных и выходных данных**

**Входные данные**

В качестве входных данных выступает список из заданных прямоугольников. Прямоугольники задаются следующим образом:

«Прямоугольник. Задаётся двумя вершинами одной из сторон. А также точкой, лежащей на прямой, проходящей через две другие вершины.»

Точки задаются координатами типа double, однако специфика отрисовки точек с помощью OpenGL ограничивает эти координаты до отрезка [-1; 1].

**Выходные данные**

В качестве выходных данных выступают пара прямоугольников, которые дают максимальный отрезок, и, конечно, сам отрезок. Их нужно выделить (я выделяю отрезок фиолетовым, а прямоугольники - жёлтым).  
  
**Математическая модель**

1. **Пересечение прямых**

Прямоугольники состоят из 4 сторон. Эти стороны принадлежат некоторым прямым. Рассмотрим, как узнать точку пересечения двух прямых. Пусть у нас есть две прямые:

Ax + By + C = 0

ax + by + c = 0

Преобразуем:

y = -A/B \* x - C/B

y = -a/b \* x - c/b

-A/B \* x - C/B = -a/b \* x - c/b

A/B \* x + C/B = a/b \* x + c/b

x(A/B - a/b) = c/b - C/B

**x = (c\*B - C\*b)/(A\*b - a\*B)**

Получаем координату х точки пересечения. Подставив эту координату в уравнение одной из прямых, узнаём координату y.

1. **Расстояние между точками**

Найдя точки пересечения, которые задают отрезок, чтобы найти его длину, нужно воспользоваться формулой:

**D = √((x1 - x2)2 + (y1 - y2)2)**

**Анализ используемой структуры данных**

Нужно хранить список прямоугольников (входные данные) для того, чтобы их нарисовать, а так же концы отрезка (точки). Их я тоже храню в списке. Также нужно сохранить ссылки на те прямоугольники, которые будут выделены (жёлтым цветом). Длины отрезков хранятся на лету, так как не требуется в ответе или где-либо ещё.

**Выбор метода решения**

Для решения задачи я перебираю все пары прямоугольников. Прямоугольники состоят из сторон, которые принадлежат некоторым прямым. Я ищу пересечения прямых одного прямоугольника с прямыми другого прямоугольника. Но не всегда пересечения прямых это пересечения сторон. Тогда я проверяю, находится ли точка пересечения прямых на сторонах прямоугольников (это реализовано в методе intersection класса rectangle).

**Комментированный листинг**

public ArrayList<Rectangle> rectangles = new ArrayList<Rectangle>();  
public ArrayList<Rectangle> TrueRectangles = new ArrayList<Rectangle>();  
//хранит те самые два прямоугольника  
  
public ArrayList<Point> TruePoint = new ArrayList<Point>();  
// хранит те самые две точки  
  
public void solve() {  
TrueRectangles.clear();  
TruePoint.clear();  
//очистить ответ  
  
double mx = 0;  
//максимальное значение длины  
  
for (int i = 0; i < rectangles.size(); i++) {  
for (int j = i + 1; j < rectangles.size(); j++) {  
//циклы по всем парам прямоугольников  
  
ArrayList<Point> t = rectangles.get(i).intersection(rectangles.get(j));  
//получаем пересечение прямоугольников, размер массива всегда или 0(если нет пересечений) или 2  
  
if (t.size() > 0) {  
//если есть пересечение  
  
double tdist = t.get(0).distanceTo(t.get(1));  
//считаем новую длину  
  
if (tdist > mx) {  
TrueRectangles.clear();  
TruePoint.clear();  
//если она больше, то обновляем максимум и ответ  
  
mx = tdist;  
TruePoint.add(t.get(0));  
TruePoint.add(t.get(1));  
TrueRectangles.add(rectangles.get(i));  
TrueRectangles.add(rectangles.get(j));  
}}}}}

**Пример работы программы**

Входные данные:

(-0,76; 0,62), (-0,44; 0,57), (0,80; 0,24)

(0,06; 0,17), (-0,70; 0,44), (-0,67; -0,06)

(0,41; 0,75), (0,03; -0,86), (0,70; -0,33)

(0,60; 0,51), (0,71; 0,11), (0,72; 0,10)

(0,48; 0,06), (0,36; 0,26), (-0,98; 0,46)

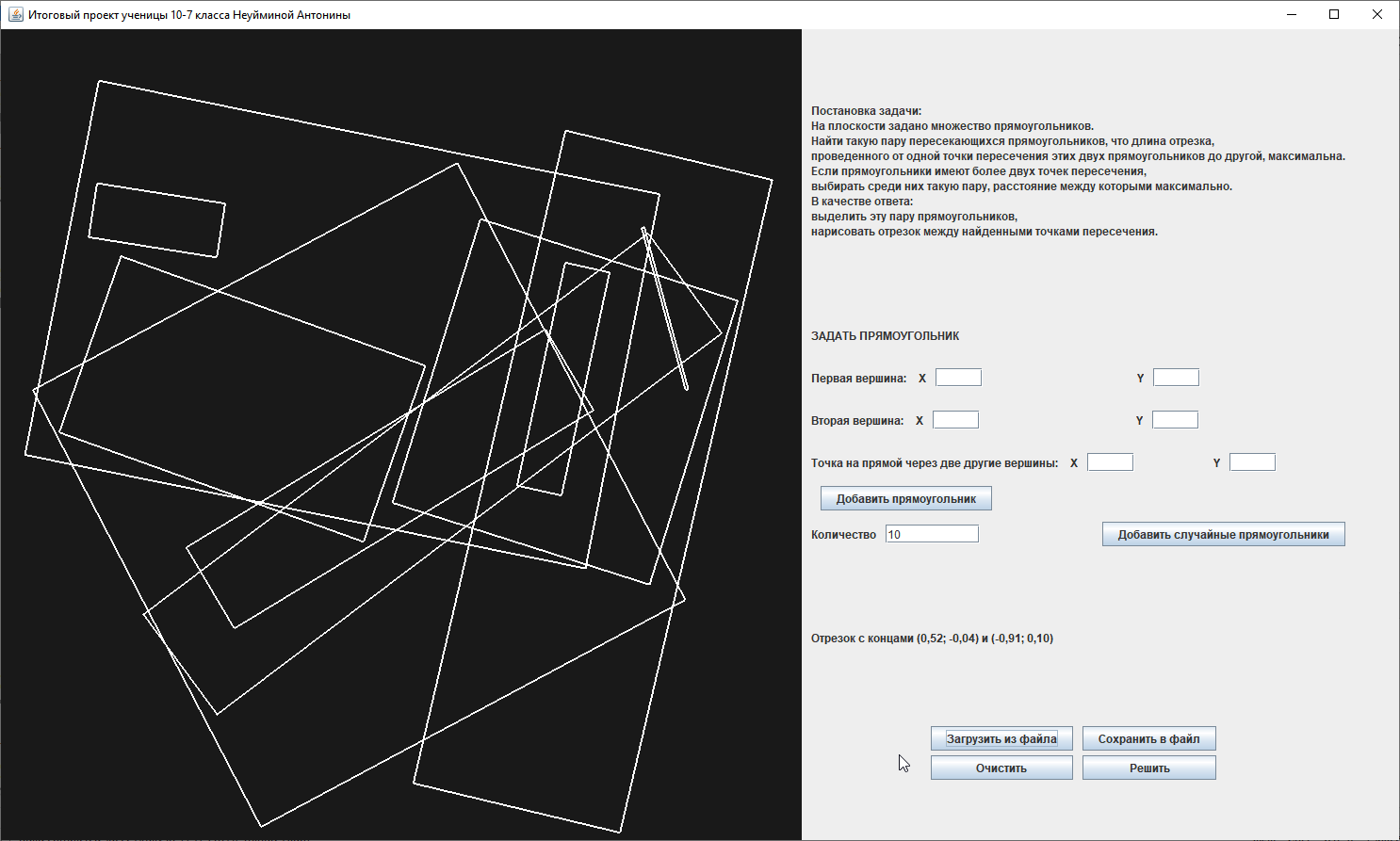
(-0,94; -0,05), (0,46; -0,33), (0,81; 0,56)

(0,40; -0,15), (0,52; 0,40), (0,43; 0,52)

(-0,46; -0,69), (0,80; 0,25), (-0,17; -0,09)

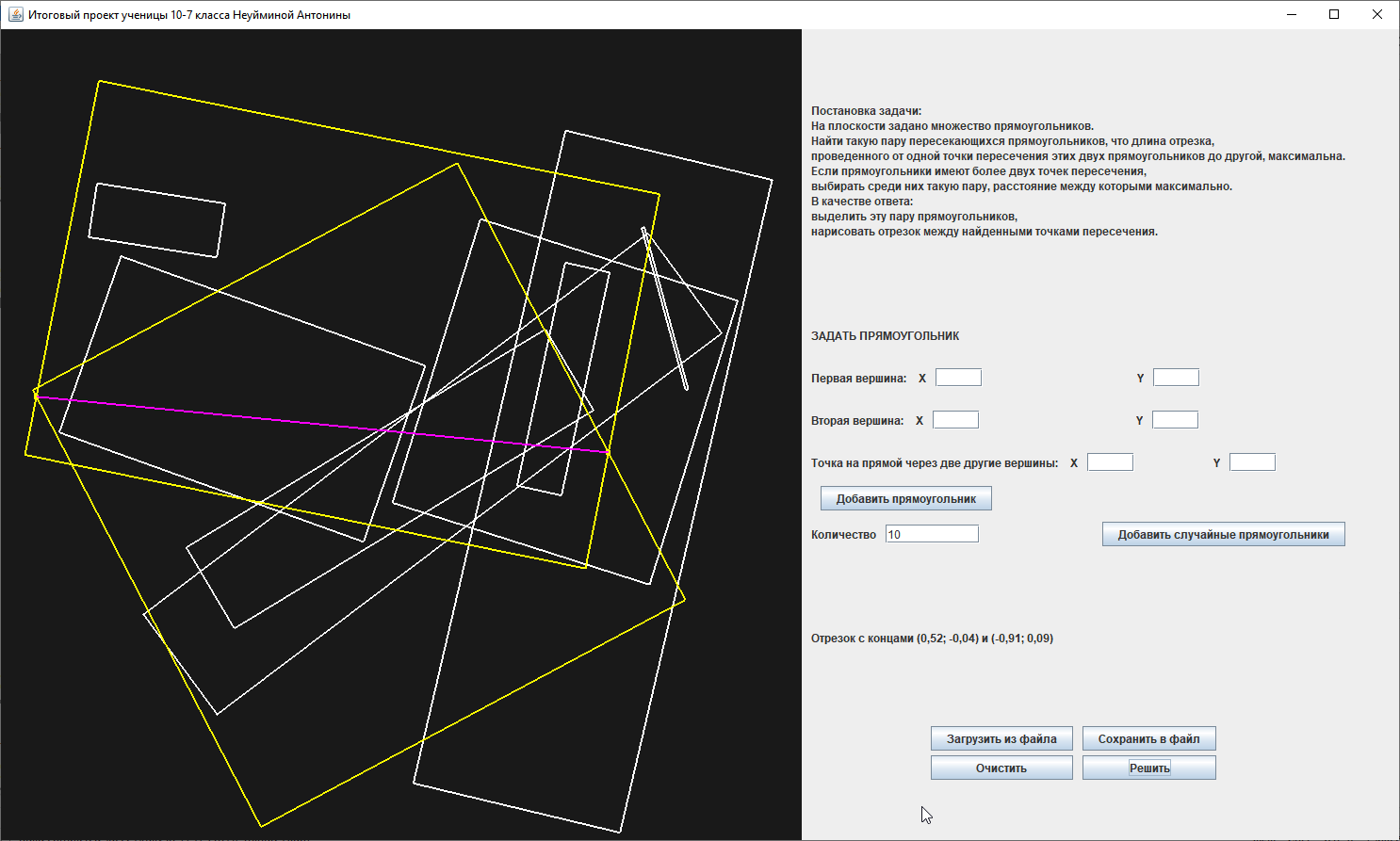
(0,62; -0,37), (0,84; 0,33), (0,26; 0,73)

(0,14; 0,67), (-0,92; 0,11), (0,97; -0,27)



Выходные данные:

Отрезок с концами (0,52; -0,04) и (-0,91; 0,10)



**Анализ правильности решения**

В основе решения задачи используются точные и простейшие математические модели: пересечение прямых и расстояние между точками. Программа точно вычисляет точки пересечения и округляет их координаты до сотых из соображений эстетики. Приведу несколько примеров задач с маленьким количеством прямоугольников, решение которых очевидно, и программа его же и выдаёт.

