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

**«Пересечения прямых и треугольника»**

Годовой проект по информатике

Автор: Марюшко Виктор, 10-7 класс

Санкт-Петербург

2021

**Постановка задачи**

На плоскости задано множество точек, и "параллельный" прямоугольник. Множество точек образует все возможные прямые, которые могут быть построены парами точек множества. Найти такую прямую (и такие две точки, через которые она проходит), что эта прямая пересекает указанный прямоугольник, и при этом длина отрезка прямой, находящейся внутри прямоугольника, максимальна. В качестве ответа: выделить найденные две точки, нарисовать прямую, которая через них проходит, а также выделить на этой прямой отрезок между двумя найденными точками пересечения.

**Уточнение исходных и выходных данных**

**Входные данные**

В качестве входных данных выступает параллельный прямоугольник и список точек. Прямоугольник задается двумя точками-вершинами. Точки попарно не совпадают. Точки задаются координатами типа double, однако отрисовка точек с помощью OpenGL ограничивает эти координаты до отрезка [-1; 1].

**Выходные данные**

В качестве выходных данных выступают две точки, дающие максимальный отрезок, прямая, проходящая через них, и отрезок. Их нужно выделить.  
  
**Математическая модель**

1. **Пересечение прямых**

Прямоугольник состоит из 4-х 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 в = 0

Преобразуем:

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

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

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

**y = -(A \* X + C) / B**

Кроме того, нужно проверить, принадлежит ли эта точка прямоугольнику. Что бы сделать это необходимо рассмотреть 4 случая в которых точка может принадлежать отрезкам прямоугольник, то есть лежит в промежутке координат точек, задающих этот прямоугольник.

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 прямоугольник, список точек(входные данные) для того, чтобы их нарисовать, а так же концы отрезка (ответ). Они хранятся в отдельном списке.

**Выбор метода решения**

Для решения задачи я перебираю все пары точек, строю по ним прямую и ищу ее пересечение с прямоугольником. Если есть пересечение, то проверяю его длину, если оно длиннее того, что находил до этого, то сохраняю эти две точки.

**Комментированный листинг**

ArrayList<Point> points;

//массив точек(входные данные)

Rectangle rectangle;

//прямоугольник

public void solve(){

for (Point p : points) {

for (Point p2 : points) {

// если точки являются разными

if (p != p2) {

// если координаты у них не совпадают

if (Math.abs(p.x - p2.x) > 0.0001 && Math.abs(p.y - p2.y) > 0.0001) {

double k = ((p.y - p2.y) / (p.x - p2.x));

double b = (p.y - k \* p.x);

double x1 = (rectangle.a.y - b) / k;

double y1 = rectangle.a.x \* k + b;

double x2 = (rectangle.b.y - b) / k;

double y2 = rectangle.b.x \* k + b;

//находим точки пересечения

Point o1 = new Point(x1, rectangle.a.y);

Point o2 = new Point(x2, rectangle.b.y);

Point o3 = new Point(rectangle.a.x, y1);

Point o4 = new Point(rectangle.b.x, y2);

ArrayList<Point> pointpl = new ArrayList<>();

pointpl.add(o1);

pointpl.add(o2);

pointpl.add(o3);

pointpl.add(o4);

//добавляем точки пересечения

for (Point point : pointpl) {

if (((point.x <= rectangle.a.x) & (point.x >= rectangle.b.x)& (point.y <= rectangle.a.y) & (point.y >= rectangle.b.y))||

((point.x >= rectangle.a.x)& (point.x <= rectangle.b.x) & (point.y >= rectangle.a.y) & (point.y <= rectangle.b.y)) ||

((point.x <= rectangle.a.x)& (point.x >= rectangle.b.x) & (point.y >= rectangle.a.y) & (point.y <= rectangle.b.y))||

((point.x >= rectangle.a.x)& (point.x <= rectangle.b.x) & (point.y <= rectangle.a.y) & (point.y >= rectangle.b.y))) {

for (Point point1 : pointpl) {

if (((point1.x <= rectangle.a.x) & (point1.x >= rectangle.b.x)& (point1.y <= rectangle.a.y) & (point1.y >= rectangle.b.y))||

((point1.x >= rectangle.a.x)& (point1.x <= rectangle.b.x) & (point1.y >= rectangle.a.y) & (point1.y <= rectangle.b.y)) ||

((point1.x <= rectangle.a.x)& (point1.x >= rectangle.b.x) & (point1.y >= rectangle.a.y) & (point1.y <= rectangle.b.y))||

((point1.x >= rectangle.a.x)& (point1.x <= rectangle.b.x) & (point1.y <= rectangle.a.y) & (point1.y >= rectangle.b.y))) {

//Проверяем точки на принадлежность прямоугольнику

double l = Math.sqrt((point.x-point1.x)\*(point.x-point1.x) + (point1.y - point1.y)\*(point1.y - point1.y));

if (l > maxLength) {

maxLength = l;

//Ищется длина отрезка по найденным точкам и проверяется на максимальность

length = new Length(point, point1);

line = new Line(point, point1);

pointred = new Point(p.x, p.y);

pointres = new Point(p2.x, p2.y);

//По найденным точкам строится прямая и отрезок пересечения, а также добавляются в список

System.out.println(maxLength);

}

}

}

}

}

}

}

}

}

}

**Пример работы программы**

Rectangle (0,4; 0,3), (0,5; 0,64)

Points

-0,20; -0,40

-0,04; 0,92

0,48; -0,76

-0,48; -0,84

-0,92; 0,64

0,76; 0,64

0,48; 0,60

-0,36; -0,28

-0,04; -0,56

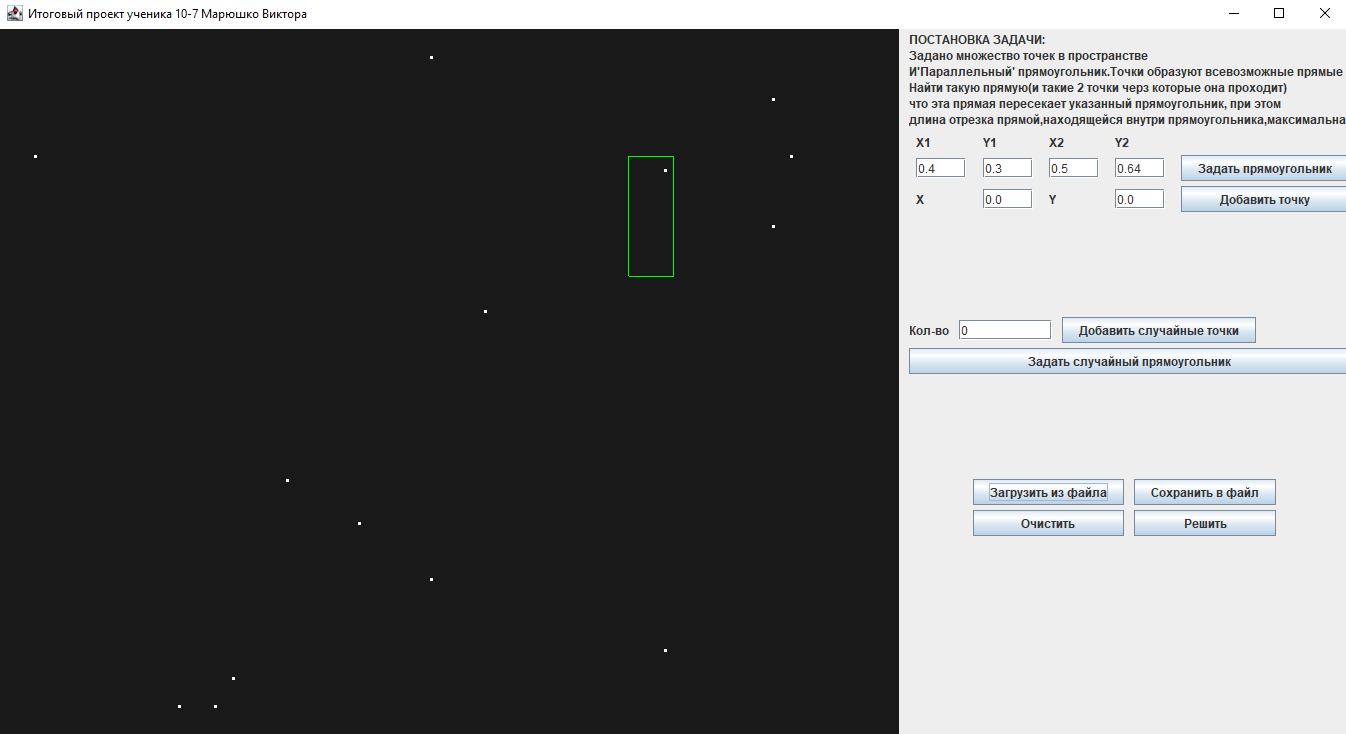
0,08; 0,20

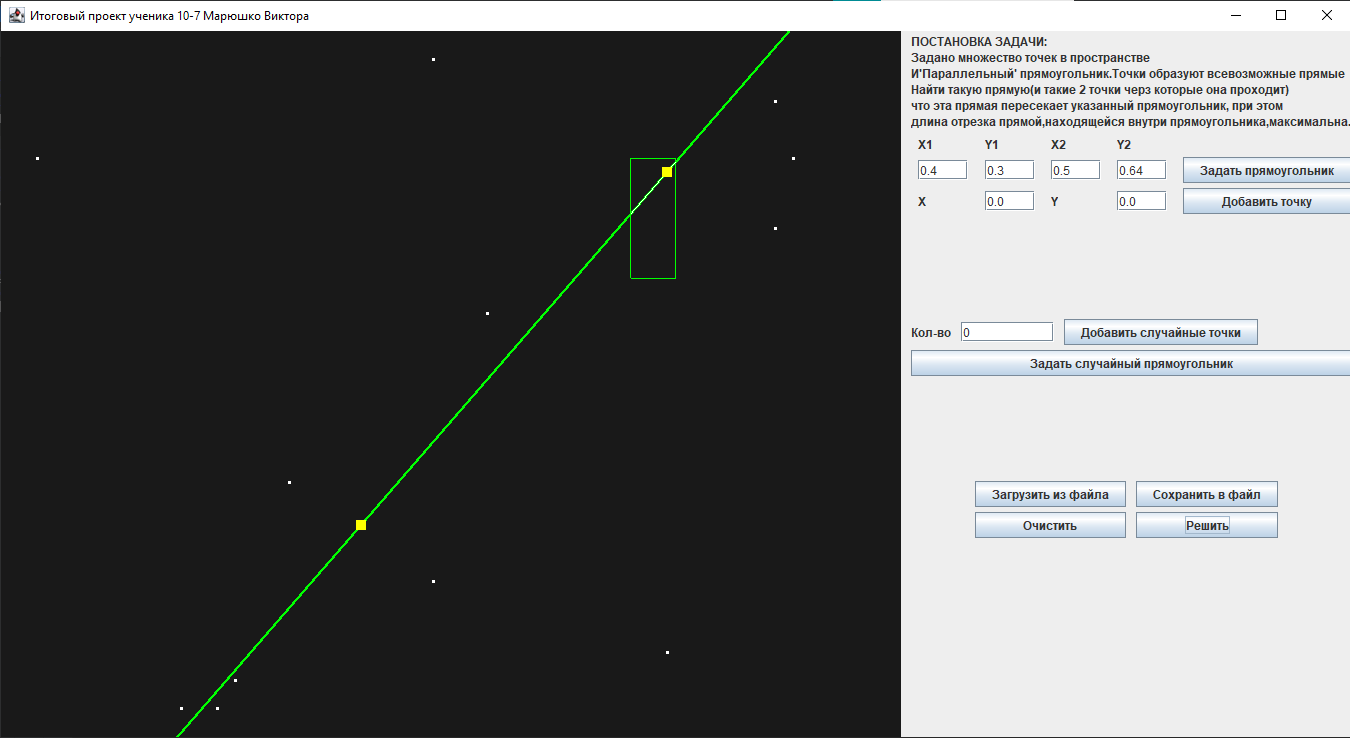
-0,52; -0,92

-0,60; -0,92

0,72; 0,44

0,72; 0,80





0.09999999999999998 //длина отрезка пересечения

Точка с координатами: {-0.2,-0.4}

Точка с координатами: {0.48,0.6}

**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, решение которых очевидно, и программа его же и выдаёт.

