**Комитет по образованию г. Санкт-Петербурга**

**ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБЩЕОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**ПРЕЗИДЕНТСКИЙ ФИЗИКО-МАТЕМАТИЧЕСКИЙ**

**ЛИЦЕЙ № 239**

**Отчет о практике**

**“Создание графических приложений на языке Java”**

Учащийся 10-3 класса

Кощеев Г.О.

Преподаватель :

Клюнин А.О.

Санкт-Петербург - 2022 год

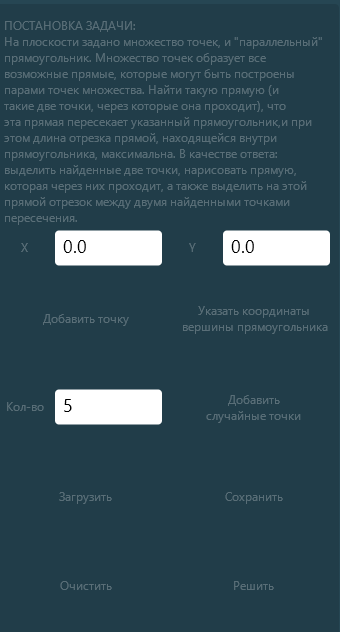
**Постановка задачи.**

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 множества, 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.



**Элементы управления.**

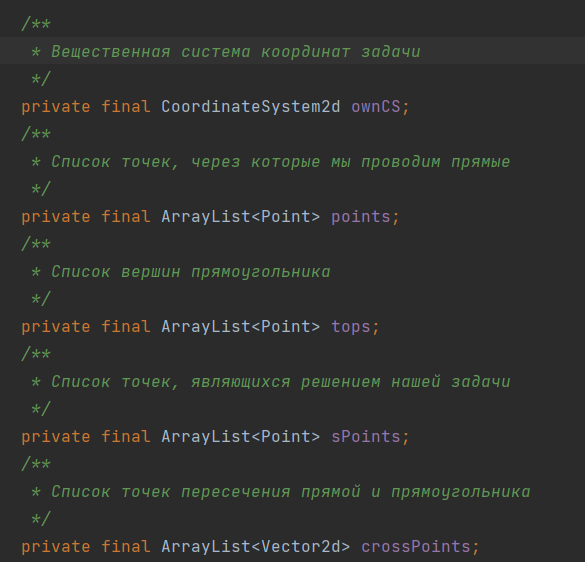
В рамках данной задачи необходимо было реализовать следующие элементы управления :



Для добавления точки по координатам было создано два поля ввода: «X» и «Y». Первая кнопка добавляет точку во множество точек, вторая же делает ее одной из вершин прямоугольника. Прямоугольник в этой задаче задается двумя противоположными вершинами. Также программа позволяет добавлять точки во множество точек с помощью клика мыши.

**Структуры данных.**

Для хранения точек, через которые мы проводим прямые, был создан массив **ArrayList<Point> points**. Вершины прямоугольника хранятся в массиве **ArrayList<Point> tops.** Нашу задачу мы решаем в вещественной системе координат **ownCS.** Для решения задачи были также созданы 2 массива : **ArrayList<Point> sPoints**(в нем хранятся точки, через которую мы проведем прямую, являющуюся решением задачи)и **ArrayList<Point> crossPoints**(здесь хранятся точки пересечения нужной нам прямой и прямоугольника).

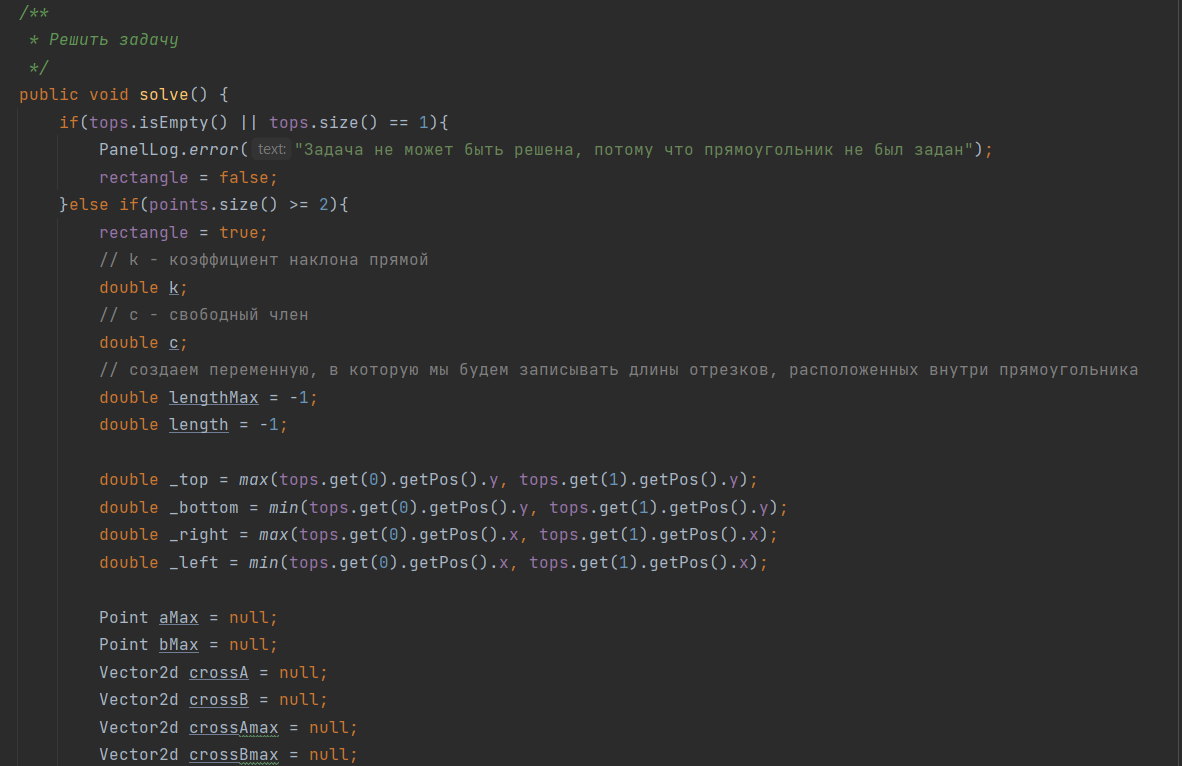


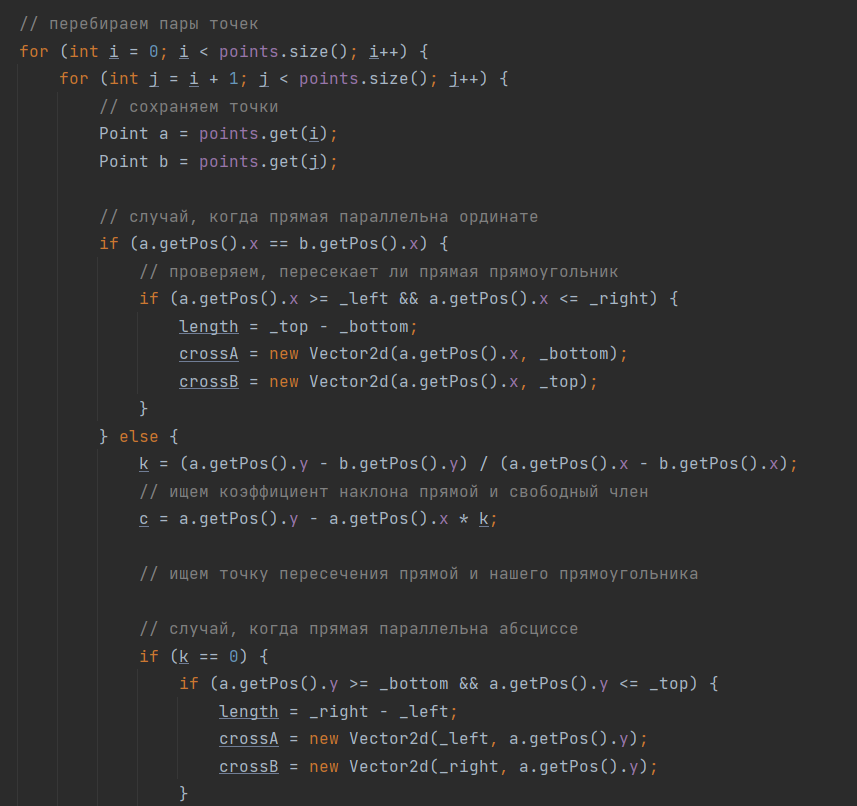
**Рисование.**

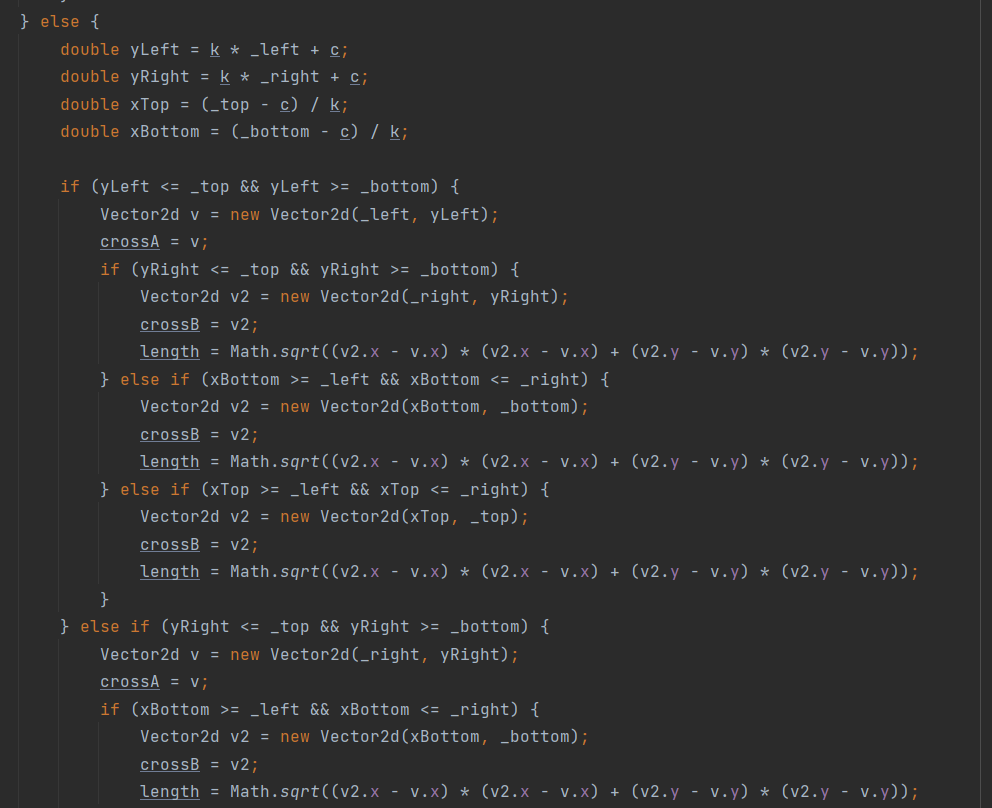
Для рисования точек использовалась команда **canvas.drawRect().** Прямоугольник же мы рисовали, используя команду рисования линии **canvas.drawLine()** 4 раза.

**Решение задачи.**

Для решения поставленной задачи в классе **Task** был разработан метод **solve()**.



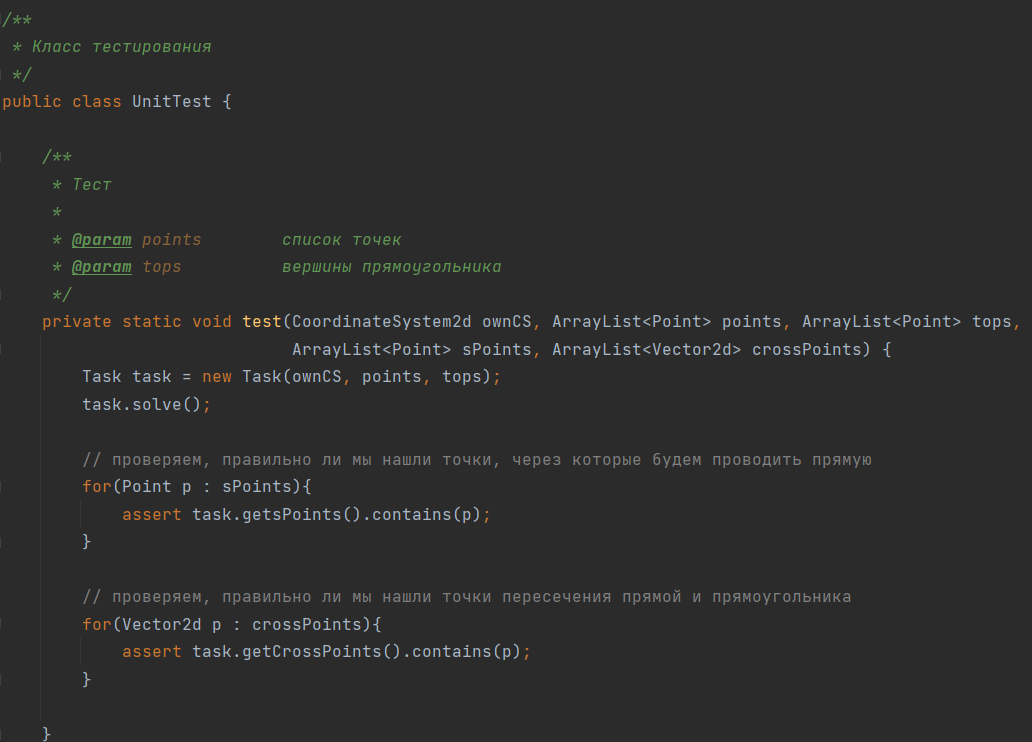






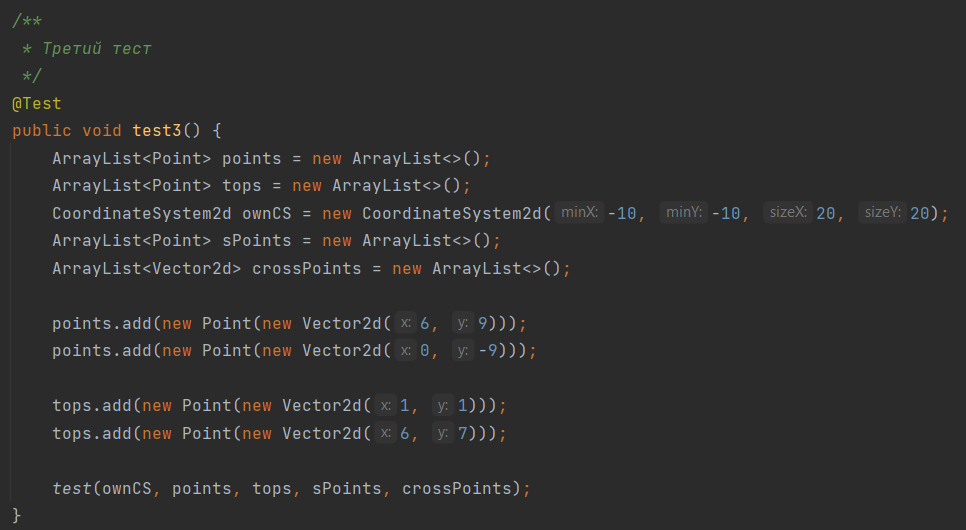
**Проверка.**

Для проверки правильности решённой задачи были разработаны unit-тесты.









**Заключение.**

В рамках выполнения поставленной задачи было создано графическое приложение с требуемым функционалом. Правильность решения задачи проверена с помощью юнит-тестов.