|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Лабораторная работа № \_\_7\_\_**

|  |  |
| --- | --- |
| **Тема РЕАЛИЗАЦИЯ АЛГОРИТМА ОТСЕЧЕНИЯ ОТРЕЗКА РЕГУЛЯРНЫМ ОТСЕКАТЕЛЕМ**  **Вариант 4 Простой алгоритм**  **Студент \_Брянская Е.В.\_\_\_\_\_\_\_\_**  **Группа \_ИУ7-42Б\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Оценка (баллы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель \_Куров А.В.** |  |

Москва.

2020 г.

**Цель работы:** изучение и программная реализация алгоритма отсечения отрезка.

Задание:

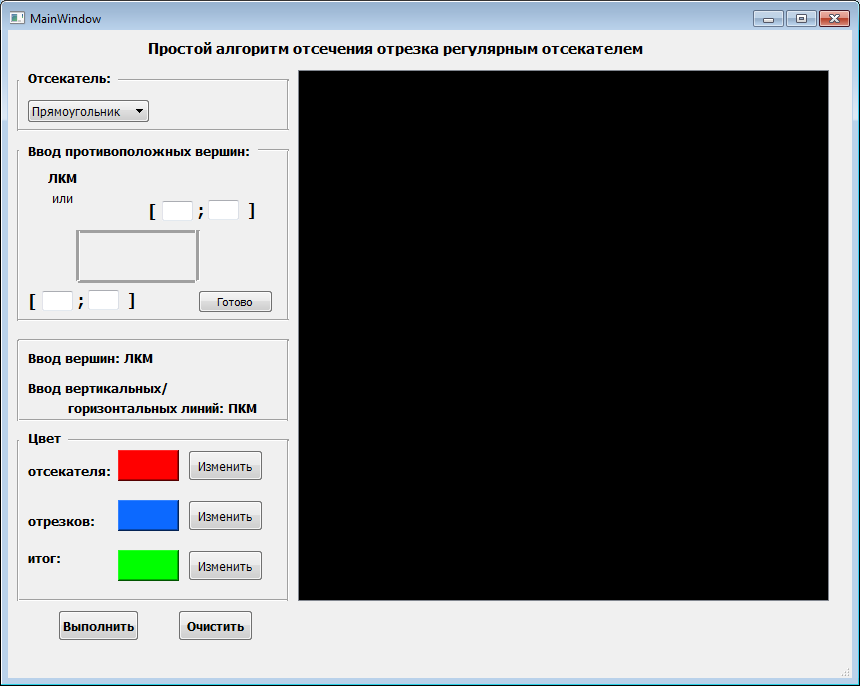
1. Реализовать Простой алгоритм отсечения отрезка

Необходимо предусмотреть:

1. Необходимо обеспечить ввод регулярного отсекателя - прямоугольника. Высветить его первым цветом.
2. Обеспечить ввод нескольких (до десяти) различных отрезков (высветить их вторым цветом). Отрезки могут иметь произвольное расположение: горизонтальные, вертикальные, имеющие произвольный наклон.
3. Ввод осуществлять с помощью мыши и нажатия других клавиш.
4. Выполнив отсечение отрезков, показать результат третьим цветом. Исходные отрезки не удалять.

**Интерфейс**

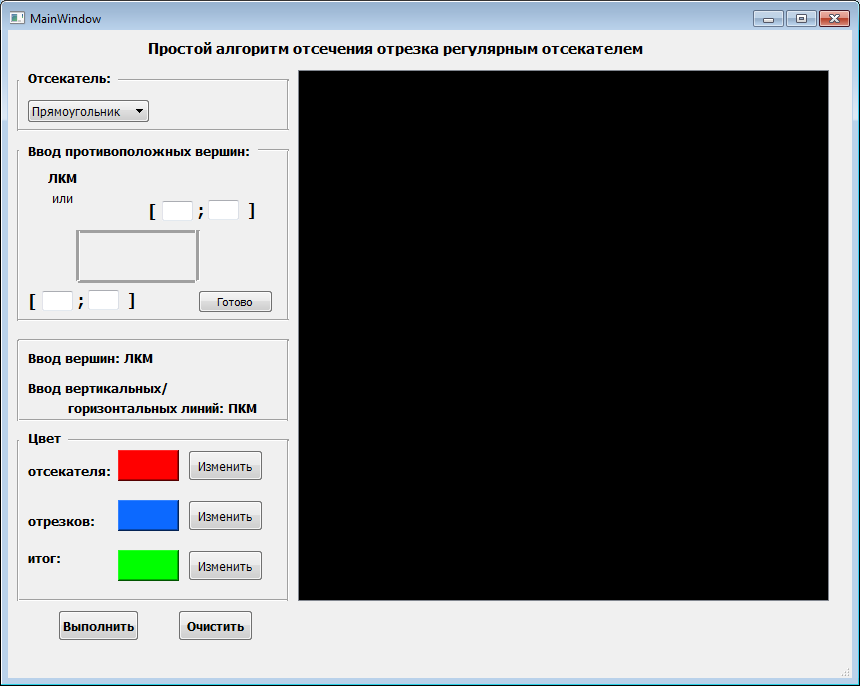
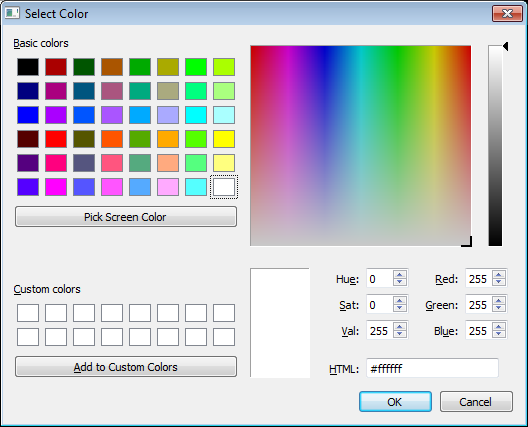
Пользователю предоставляется следующий интерфейс для работы:



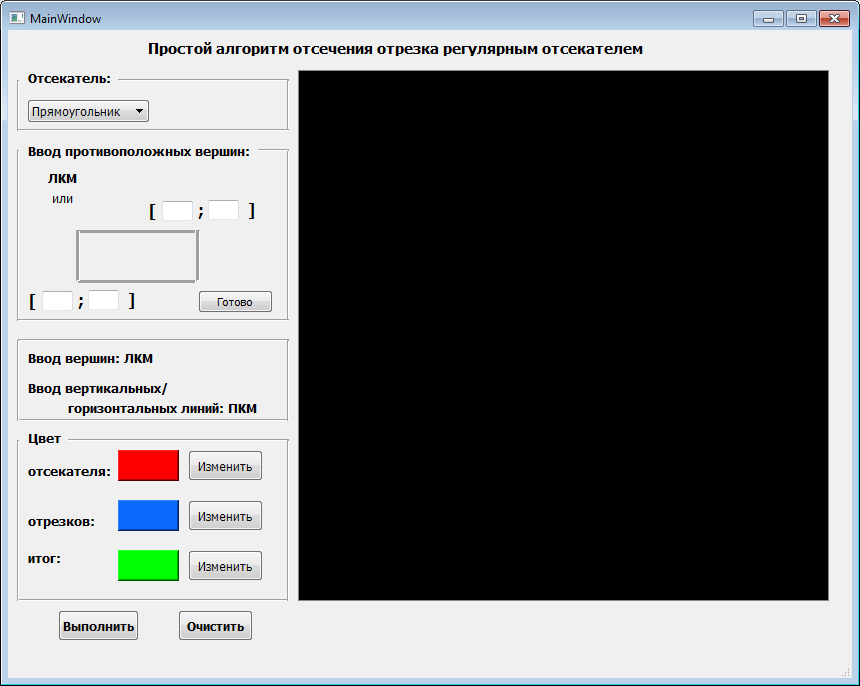
Пользователь сначала должен ввести отсекатель. Это можно сделать двумя способами: «заполнить схему», указав координаты противоположных вершин регулярного отсекателя, либо на чёрном поле сделать два последовательных клика левой кнопкой мыши, тем самым указав вершины, задающие прямоугольник. Он рисуется сразу же после того, как будет получено необходимое количество данных.

Чтобы ввести отрезки (до 10 штук), пользователь должен указать их расположение на холсте, используя левую кнопку мыши. Если необходимо ввести горизонтальный или вертикальный отрезок, то тогда нужно сместить курсор на необходимое расстояние и нажать правую кнопку мыши (в программе предусмотрена соответствующая коррекция координат).

Также пользователь может выбирать цвет отсекателя, отрезков и итогового результата:



Ещё предусмотрены такие команды, как «Выполнить» и «Очистить» для соответствующих задач.



**Простой алгоритм отсечения отрезка**

*Об алгоритме*

В качестве отсекателя выступает прямоугольник со сторонами, параллельными осям координат. То есть для осуществления отсечения необходимы такие параметры, как .

Для того чтобы сэкономить время, в первую очередь проверяется признак тривиальной видимости отрезка (обе вершины видимы) и признак тривиальной невидимости отрезка (распознаются только те, вершины которых расположены одинаковым образом от отсекателя).

Для этого производится анализ четырёхразрядного кода вершин, характеризующий расположение относительно регулярного отсекателя. Если сумма разрядов кода равна нулю, то вершина видима, и соответственно, если обе суммы концов отрезка равны нулю, то отрезок полностью видим. Дальнейший анализ производить не нужно, можно сразу вывести результат.

Если одна из двух сумм равна нулю, а вторая нет, то отрезок заведомо частично видим. И последующий анализ нужно производить только с одной вершиной.

Ненулевой результат поразрядного логического умножения кодов концов отрезка даёт информацию о том, что он полностью невидим и какое-либо дальнейшее исследование уже не нужно (но на этом этапе отрабатываются не все невидимые отрезки, только те, концы которых расположены одинаковым образом от прямоугольника).

Оставшиеся отрезки обрабатываются уже путём анализа координат концов и параметров отсекателя.

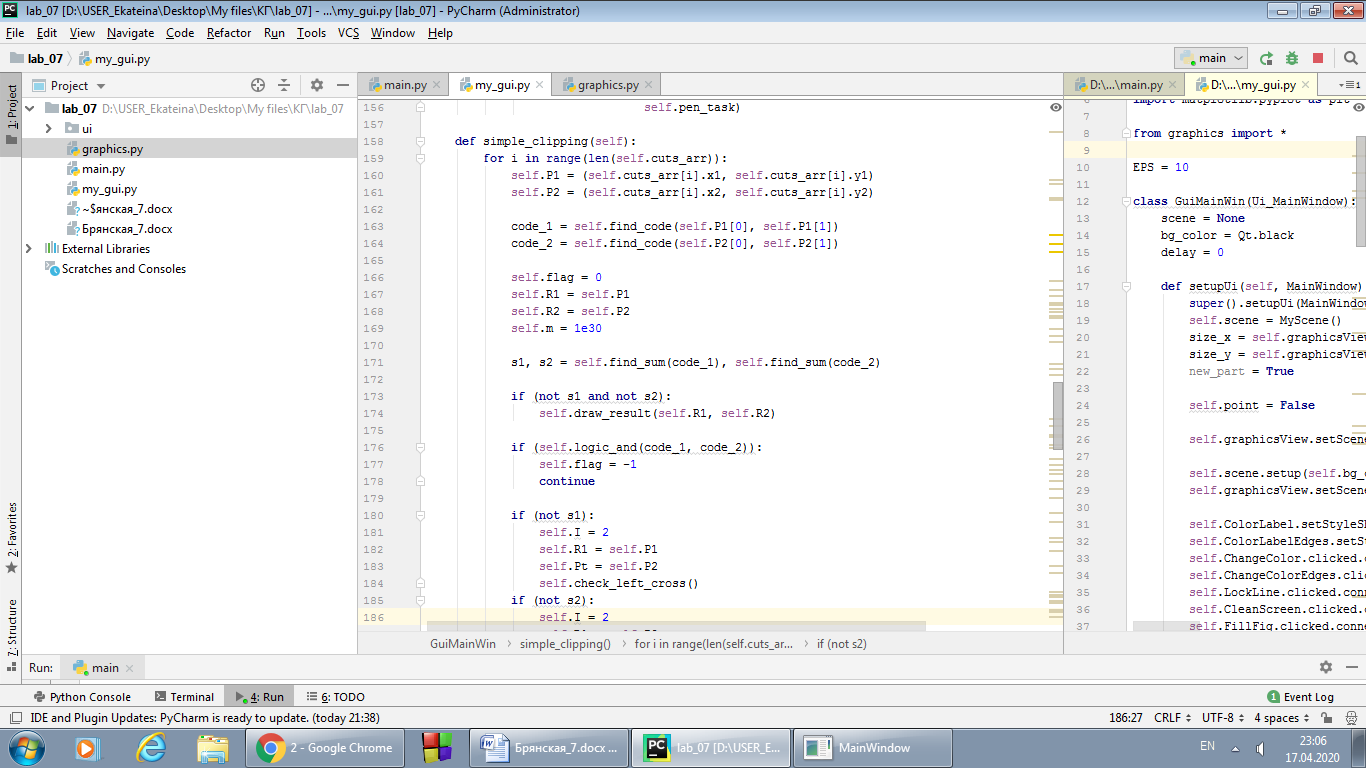
Особенностью данного алгоритма является в необходимости следить за тем, чтобы точка пересечения отрезка с отсекателем находилась в нужном диапазоне, так как может располагаться за его пределами, на продолжении стороны.

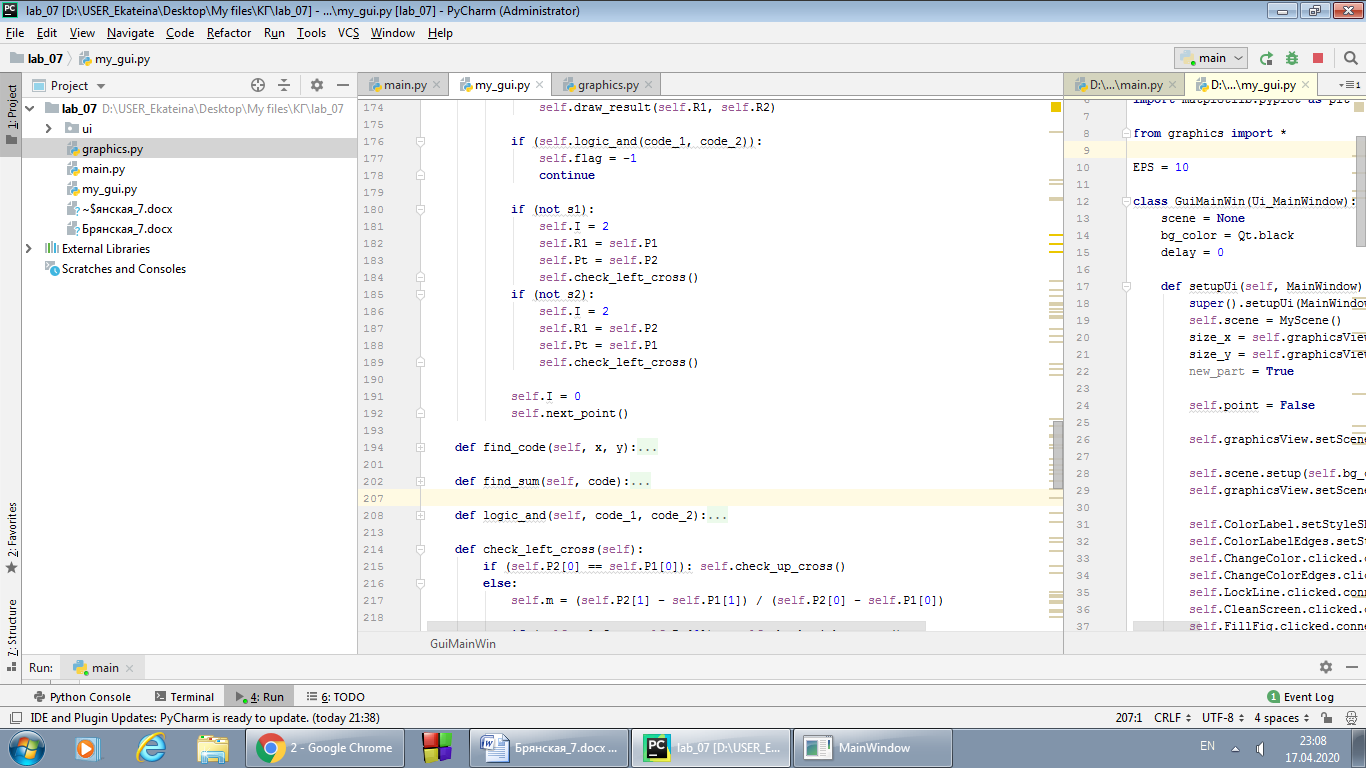
Если отрезок вертикальный, то в данном алгоритме не рассматриваются его пересечения с левой и правой стороной отсекателя, сразу анализируется верхнее/нижнее.

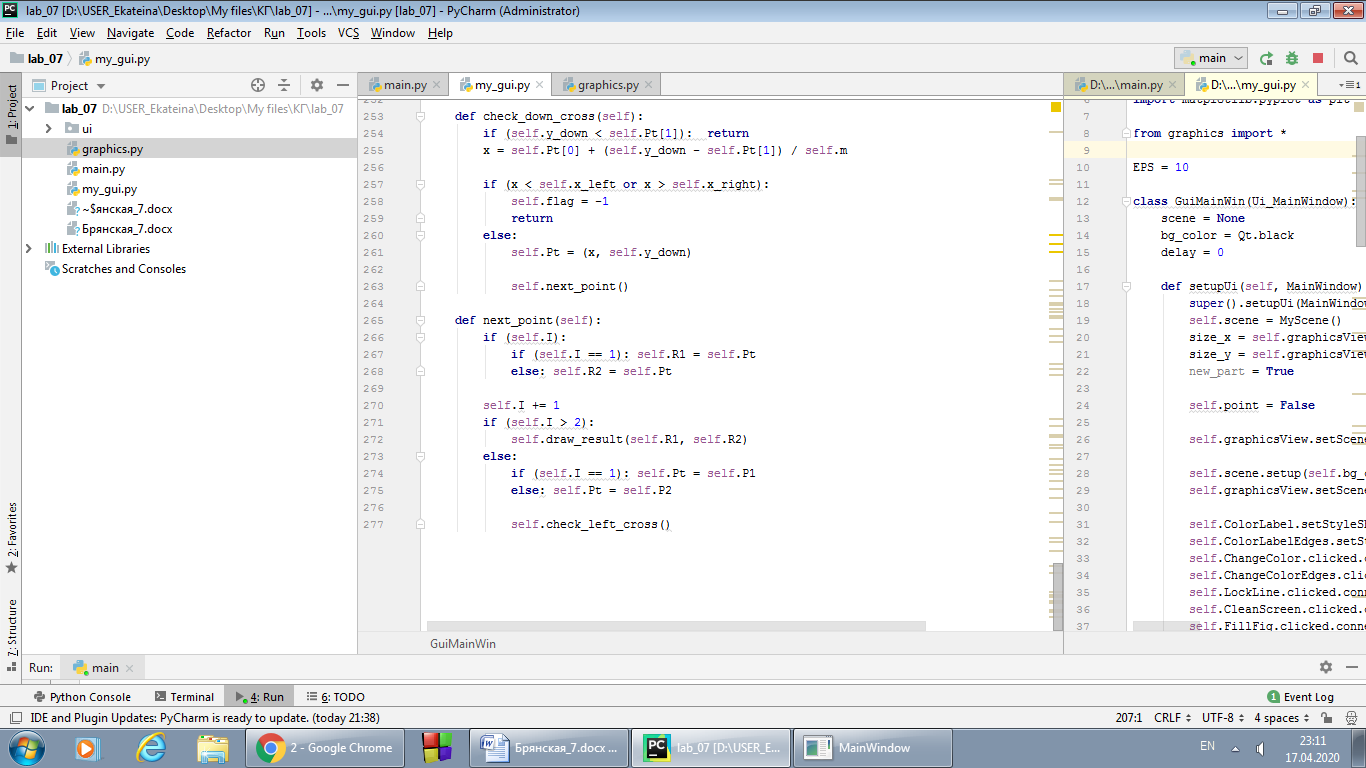
Важно отметить, что граница отсекателя принадлежит внутренней области.

*Реализация:*

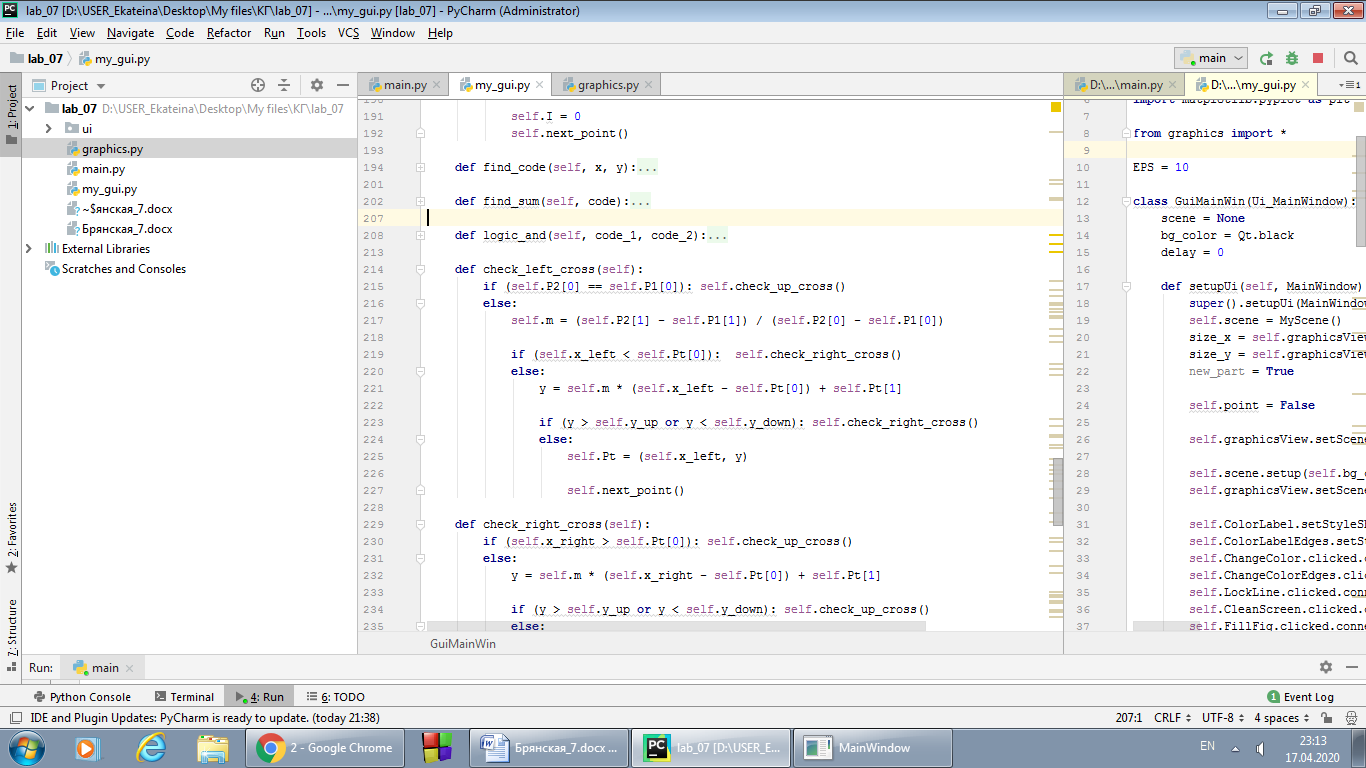
Основной цикл

**

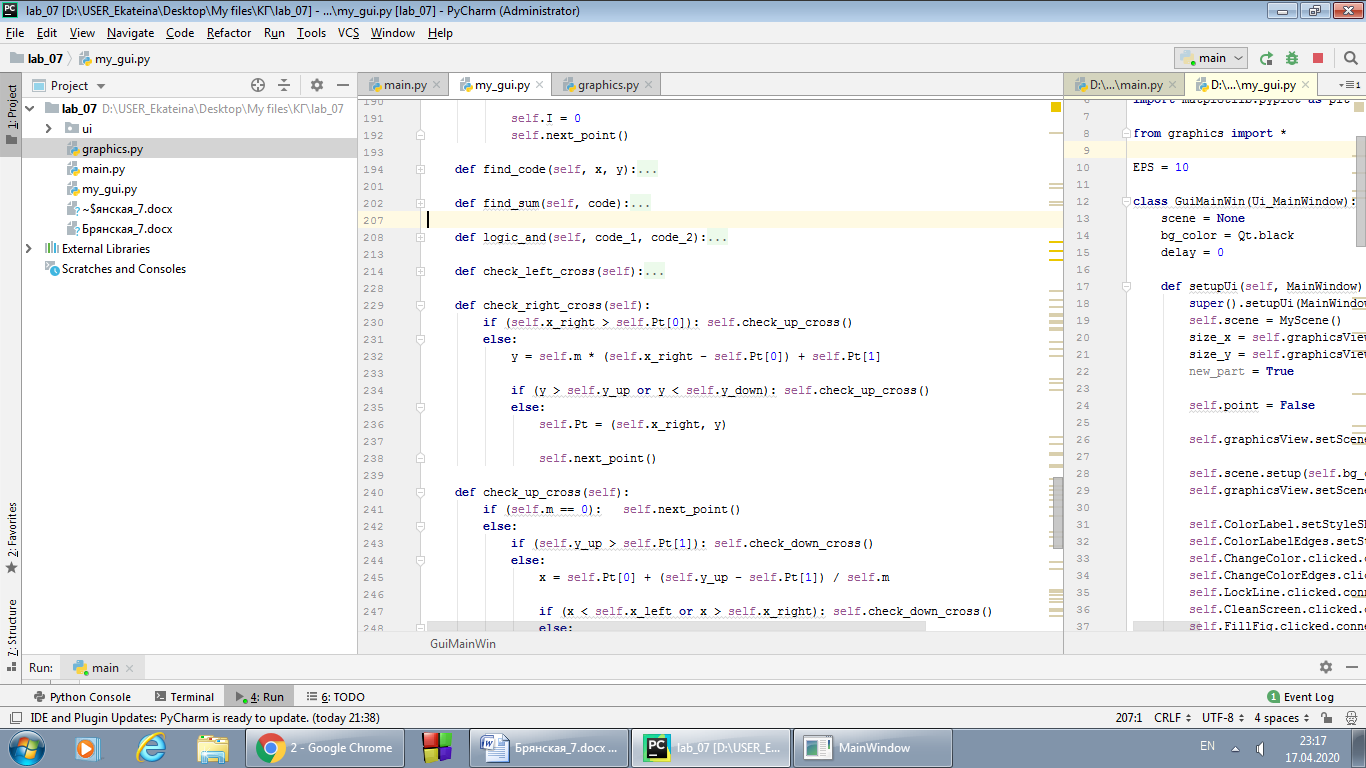
**

**

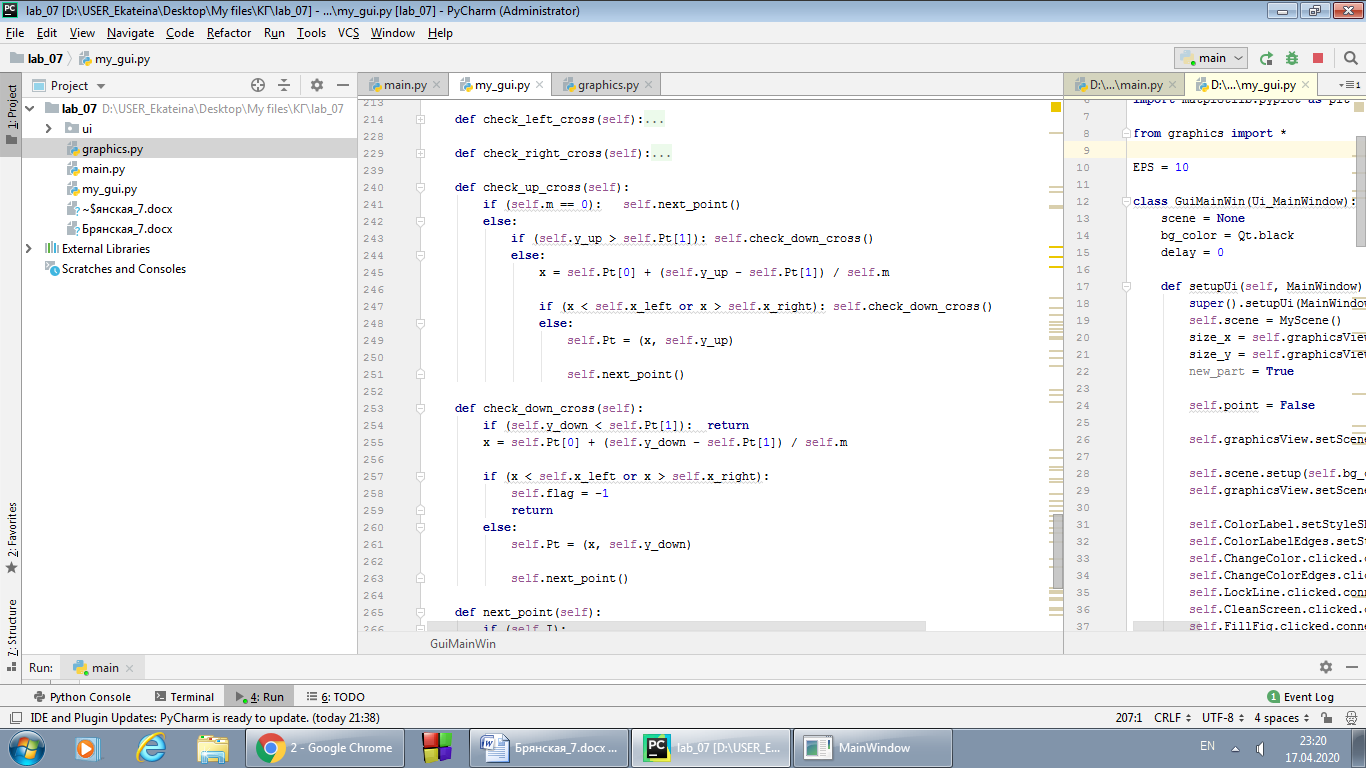
Проверка пересечения с левой стороной отсекателя (и проверка на то, является ли отрезок вертикальным):

**

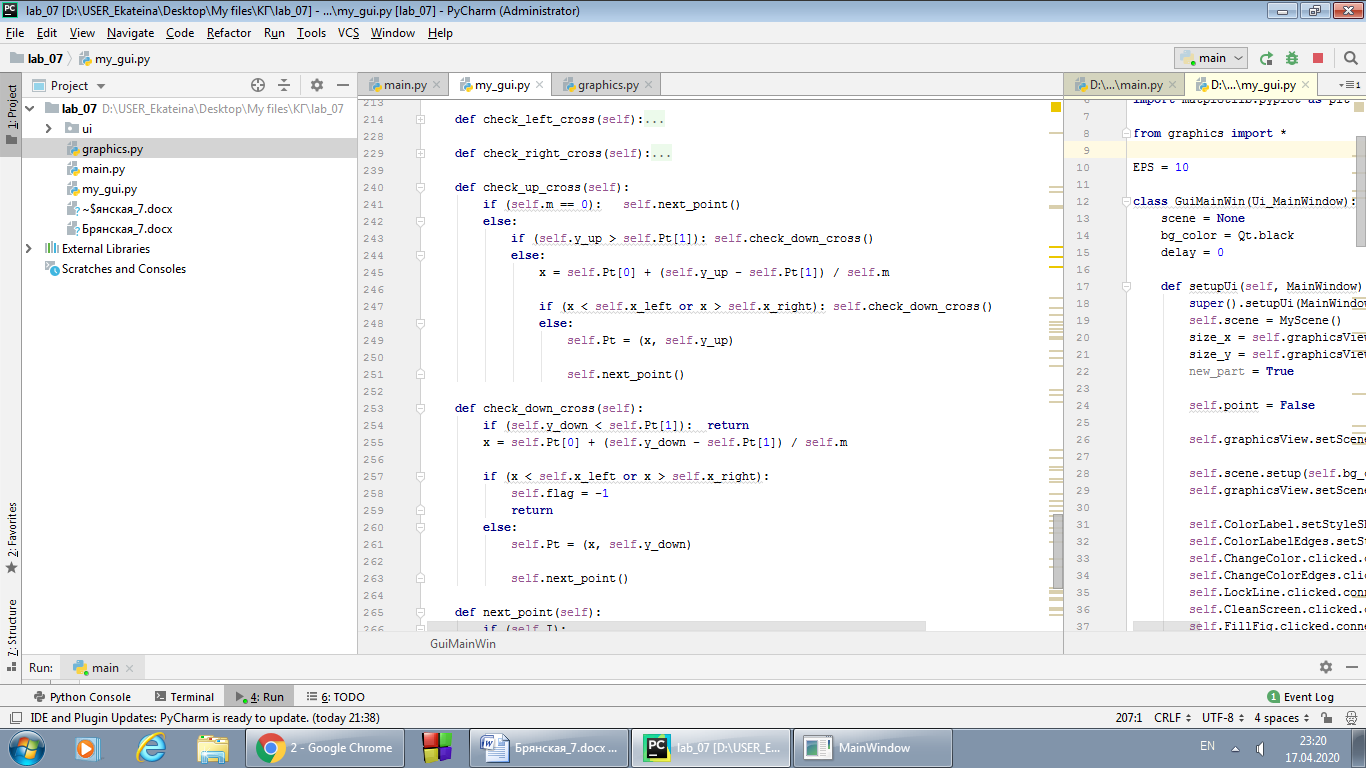
Проверка пересечения с правой стороной:



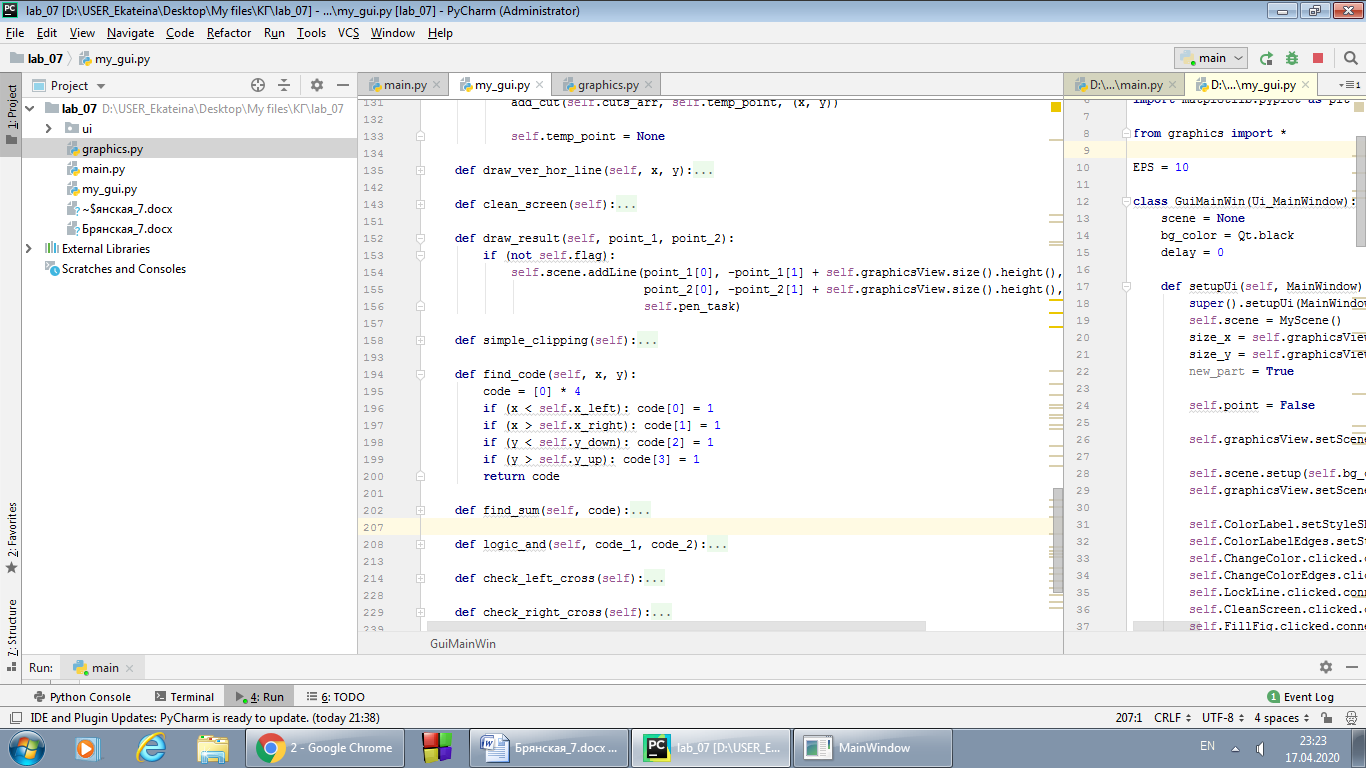
Проверка пересечения с верхней стороной:



Проверка пересечения с нижней стороной:



Функция для нахождения кодов:



*Примеры работы:*

|  |  |
| --- | --- |
| Полностью видимые отрезки | |
|  |  |
| Невидимые отрезки (для наглядности того, что программа работает, добавлен один видимый отрезок) | |
|  |  |
| Частично видимые отрезки | |
|  |  |
| Все виды вместе | |
|  |  |
| “Граничные случаи” | |
| *Безымянный.png* | *Безымянный1.png* |

**Вывод**

В ходе лабораторной работы был изучен и реализован простой алгоритм отсечения отрезков. Несмотря на то, что в этом алгоритме используются приёмы для сокращения времени работы (анализ кодов вершин, идентификация отрезков, которые легко распознаются и т.д.) алгоритм в целом не является эффективным. Так как помимо анализа его расположения ещё важно контролировать корректность точки пересечения отрезка и регулярного отсекателя.