|  |  |
| --- | --- |
| 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 № \_\_9\_\_**

|  |  |
| --- | --- |
| **Тема Программная реализация алгоритма отсечения произвольного многоугольника выпуклым отсекателем (Алгоритм Сазерленда-Ходжмена)**  **Студент \_Брянская Е.В.\_\_\_\_\_\_\_\_**  **Группа \_ИУ7-42Б\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Оценка (баллы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель \_Куров А.В.** |  |

Москва.

2020 г.

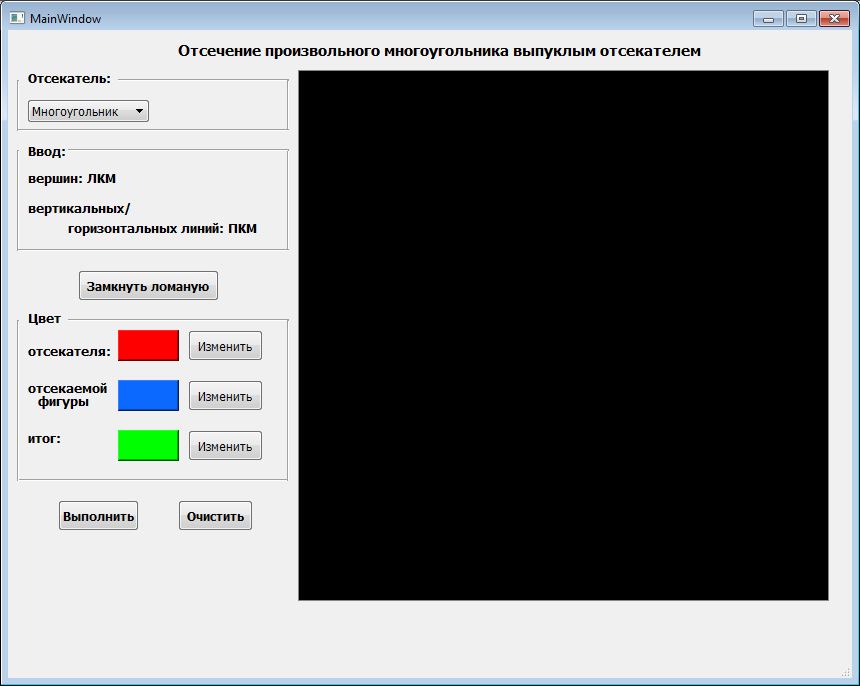
**Цель работы:** изучение и программная реализация алгоритма Сазерленда-Ходжмена отсечения многоугольников.

Необходимо предусмотреть:

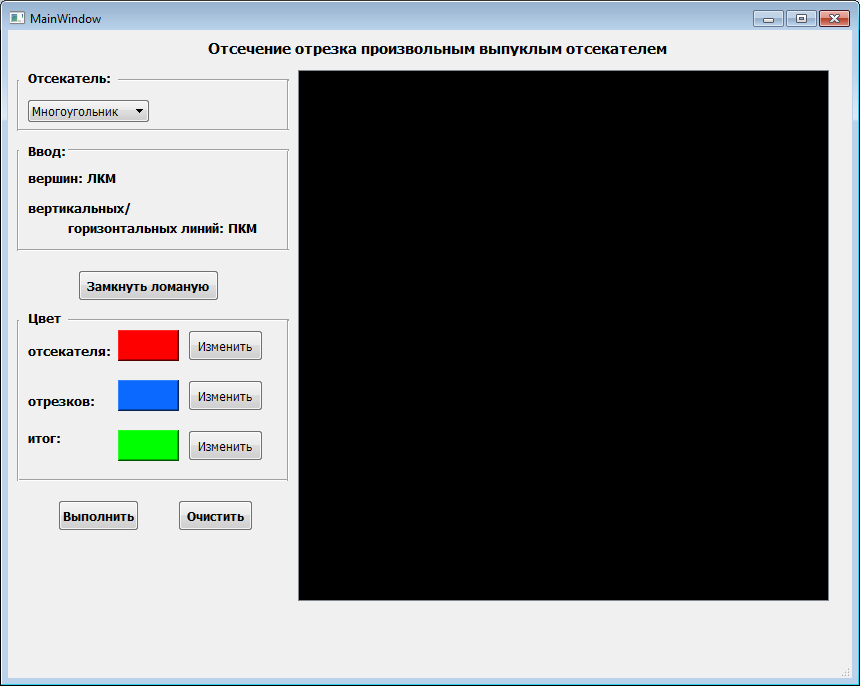
1. Необходимо обеспечить ввод отсекателя – произвольного многоугольника. Высветить его первым цветом.
2. Обеспечить ввод отсекаемого многоугольника (высветить вторым цветом).
3. Ввод осуществлять с помощью мыши и нажатия других клавиш.
4. Выполнить отсечение многоугольника, показав результат третьим цветом. Исходный многоугольник не удалять.

**Интерфейс**

Пользователю предоставляется следующий интерфейс для работы:

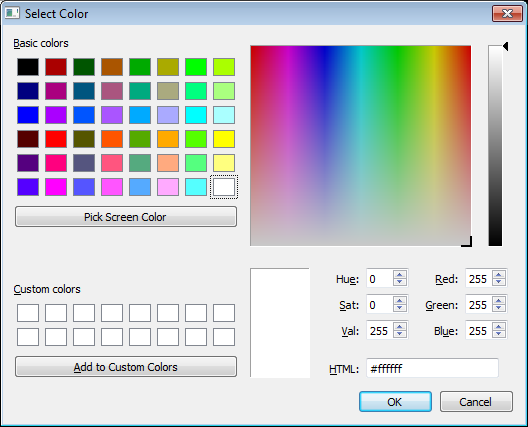


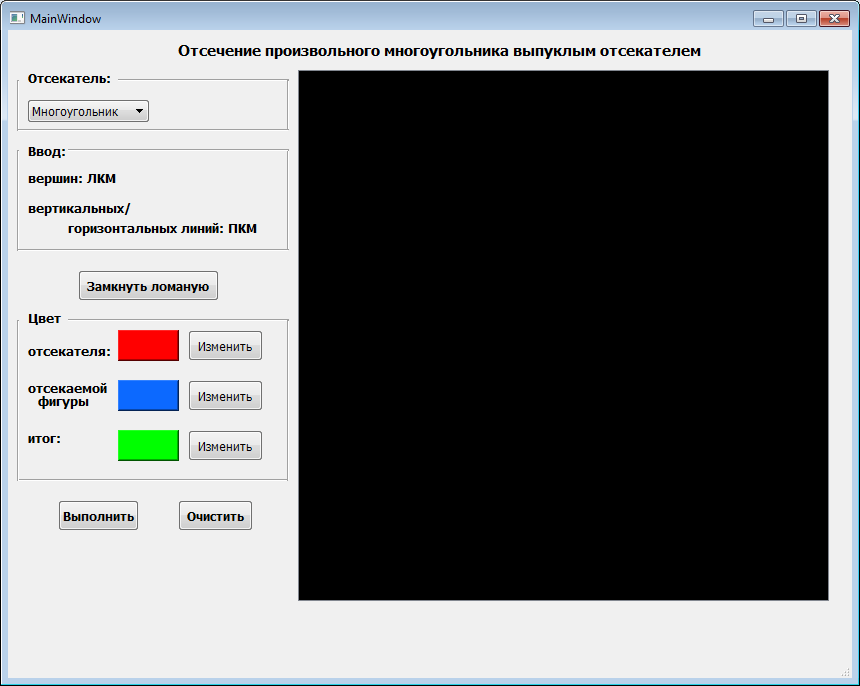
Пользователь сначала должен ввести отсекатель. Ввод вершин осуществляется с помощью мыши, путём нажатия на черный холст. Если нажимается левая кнопка мыши, то рисуется соответствующее ребро, если пользователю нужно изобразить вертикальную или горизонтальную прямую, то он должен сместить курсор на необходимое расстояние и нажать правую кнопку мыши, программа сама скорректирует координаты и выполнит задачу.

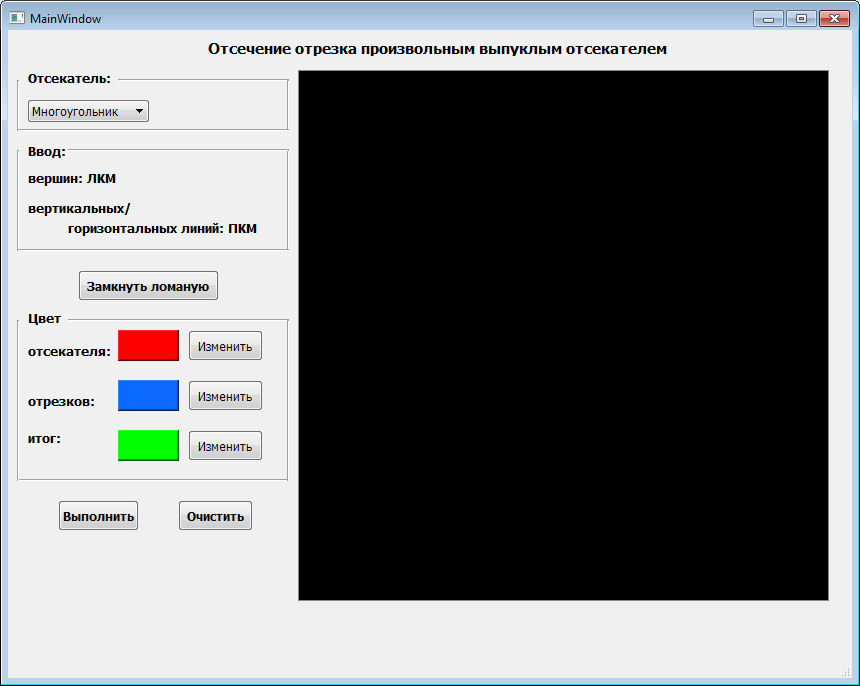
Команда «Замкнуть ломаную» автоматически соединяет начальную точку с конечной.

Пользователь может сделать это и самостоятельно, достаточно относительно близко к начальной вершине нажать мышкой (предусмотрена погрешность, попав в которую координаты программно корректируются) и рисуется соответствующее ребро.

Ввод многоугольника осуществляется аналогичным образом. Ещё предусмотрен ввод вершин многоугольника в произвольных точках ребер отсекателя.

Также пользователь может выбирать цвет отсекателя, отсекаемого многоугольника и итогового результата:



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

**Алгоритм Сазерленда-Ходжмена**

*Об алгоритме*

Этот алгоритм позволяет произвести отсечение произвольного многоугольника (как выпуклого, так и невыпуклого) выпуклым отсекателем. Основная идея алгоритма – многоугольник отсекается последовательно каждой из сторон отсекателя.

Результатом работы алгоритма является список вершин многоугольника, у которого все вершины лежат по видимую сторону от всех ребер отсекателя.

В начале работы алгоритма необходимо проверить, является ли отсекатель выпуклым, если нет, то далее никакие действия производиться не будут.

В ходе работы алгоритма осуществляется проход по всем сторонам отсекателя, анализируется видимость каждого из ребер многоугольника относительно рассматриваемой стороны. Причем список вершин многоугольника постоянно корректируется (какие-то вершины могут удаляться из списка, какие-то, наоборот, добавляться). Это зависит от того, является ли ребро видимым по отношению к текущей стороне отсекателя.

В случае, если после очередного отсечения, список вершин многоугольника оказался пустым, анализ фигуры можно прекратить, так как она считается полностью невидимой.

Отдельно обрабатывается первая вершина многоугольника. Если она видима, то заносится в результат, если нет, то в результат не попадает, но в любом случае становится начальной точкой.

Каждая точка из списка вершин принимается за конечную точку ребра многоугольника (а начальной считается предыдущая). И далее анализируется расположение этого ребра относительно отсекателя.

Видимость точки определяется по векторному произведению вектора стороны отсекателя и вектора, соединяющего какую-либо точку на этой стороне и рассматриваемую точку. Если знак произведения совпадает со знаком, полученном при анализе выпуклости, значит, вектор лежит по одну сторону с внутренней нормалью (следовательно, точка видима).

Факт пересечения ребра многоугольника со стороной отсекателя можно определить по видимости концов ребра. Если видимость различна, то пересечение есть. Саму точку можно найти из системы двух уравнений прямых, проходящих через соответствующие отрезки.

Если оно полностью видимо, то в результат заносится конечная вершина (начальная не заносится, так как если она видима, то уже была занесена на предыдущем шаге).

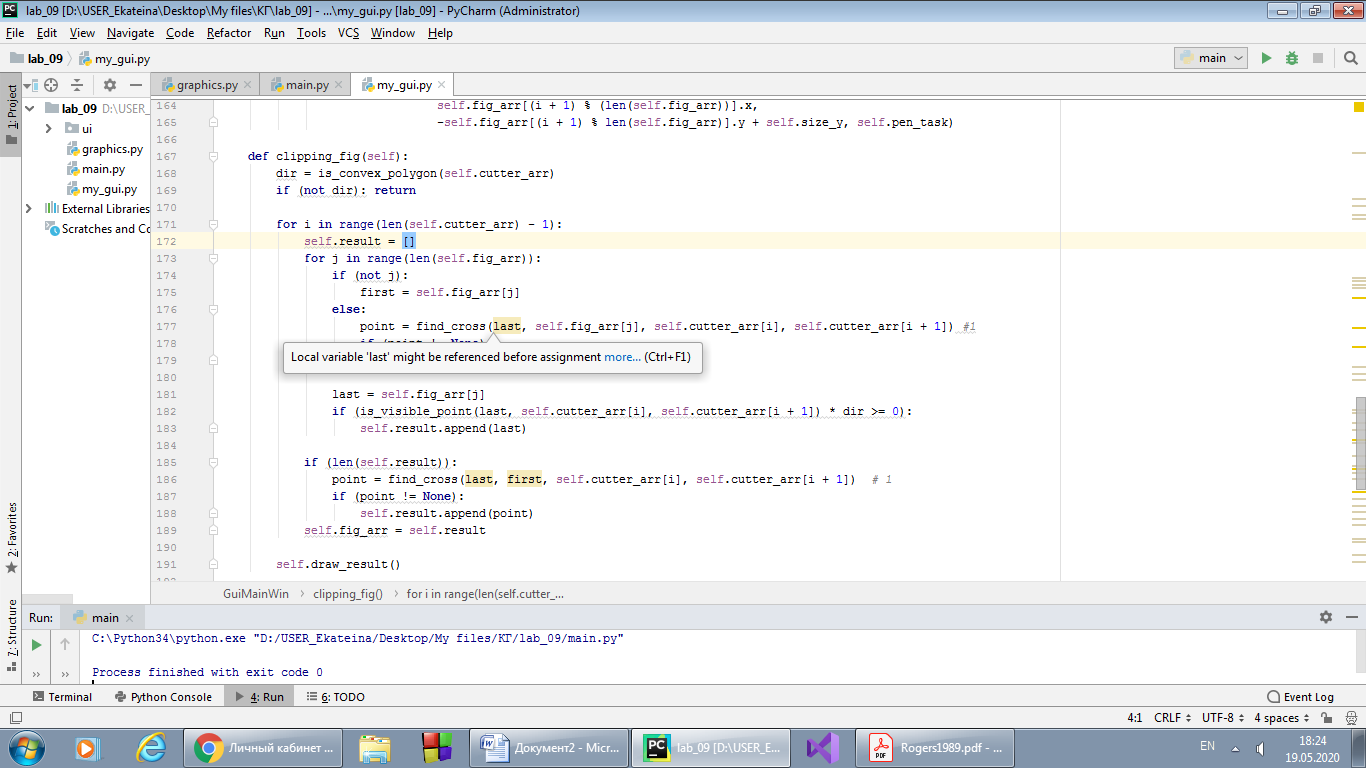
Если ребро полностью невидимо, то ничего не добавляется.

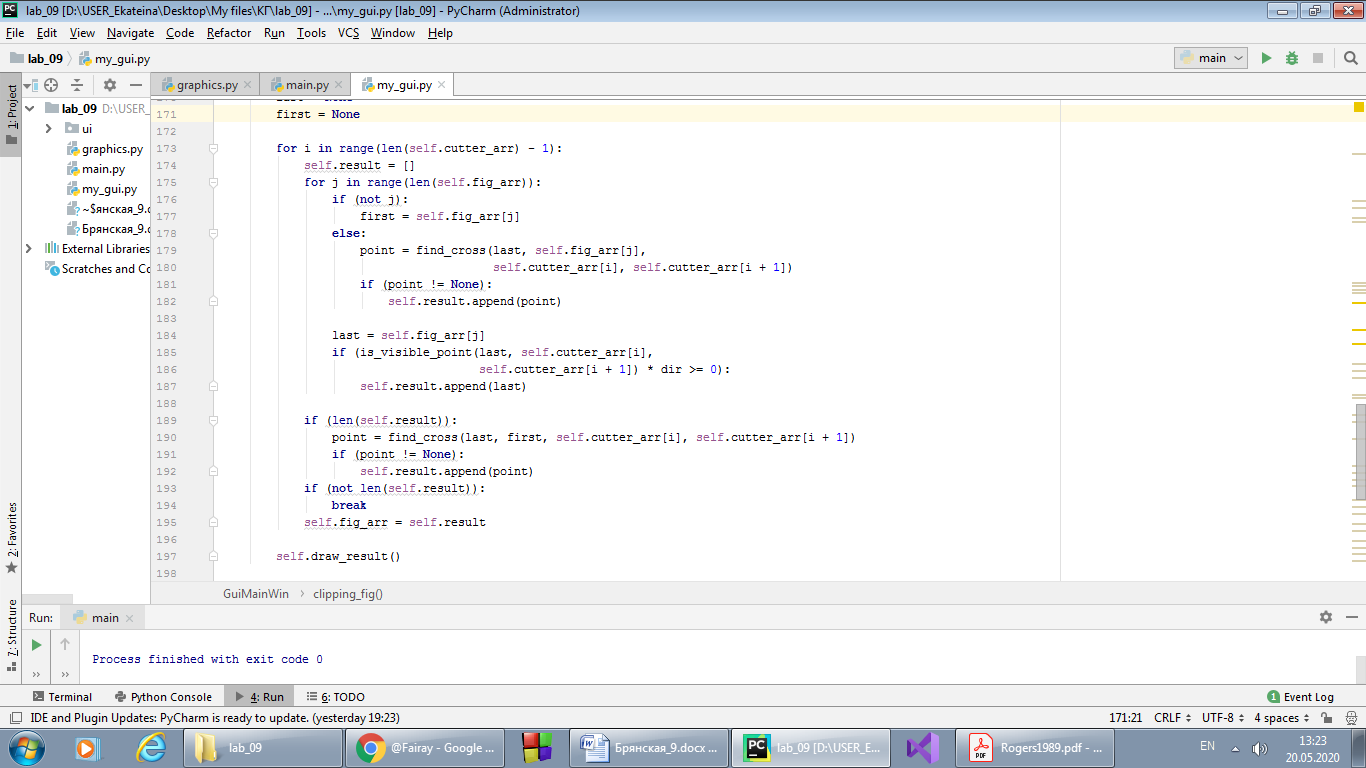
Если же ребро видимо частично, то нужно найти точку пересечения этого ребра и текущей стороны отсекателя и занести её в результат.

Заключительное ребро по-особому обрабатывается. Чтобы его можно было анализировать аналогично другим рёбрам, одним из его концов принимается начальная вершина, которую запомнили в самом начале обхода.

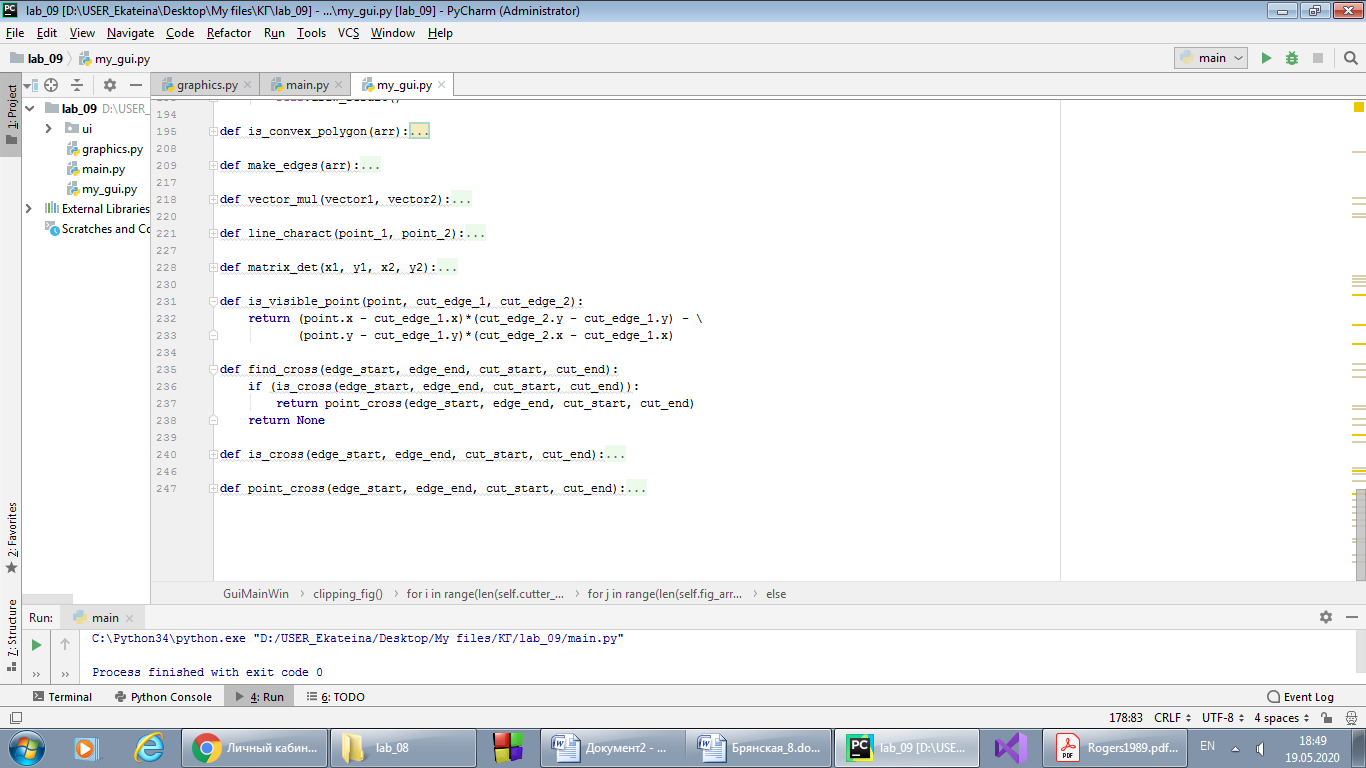
*Код программы:*

Основная функция:

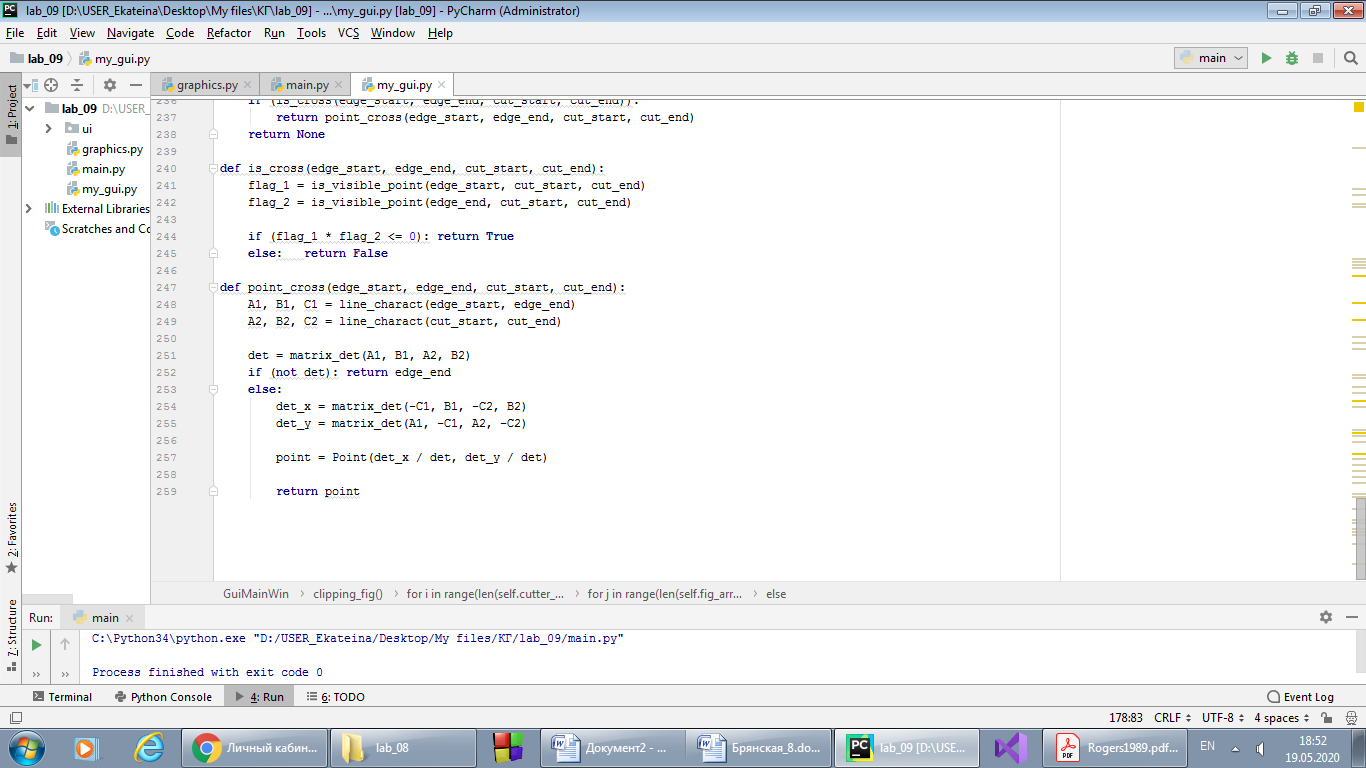
**

**

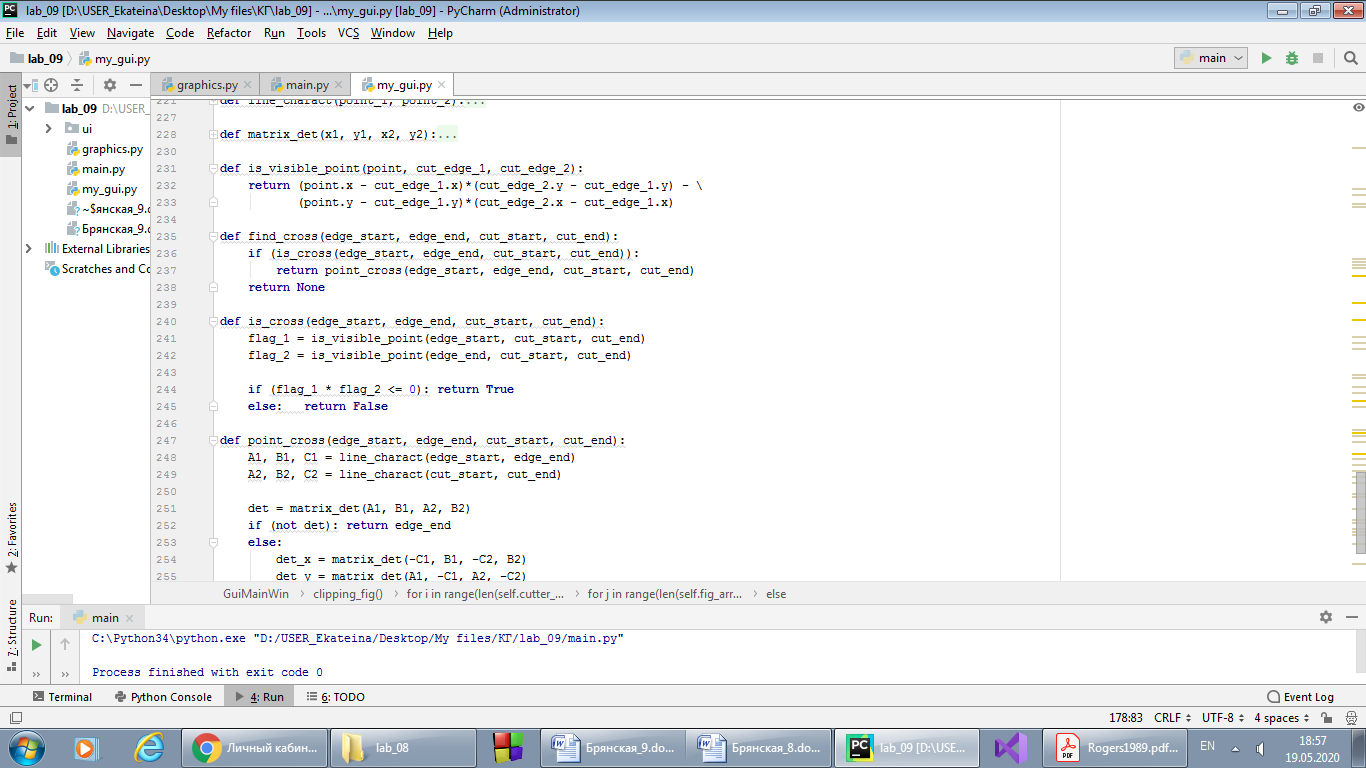
Нахождение пересечения:



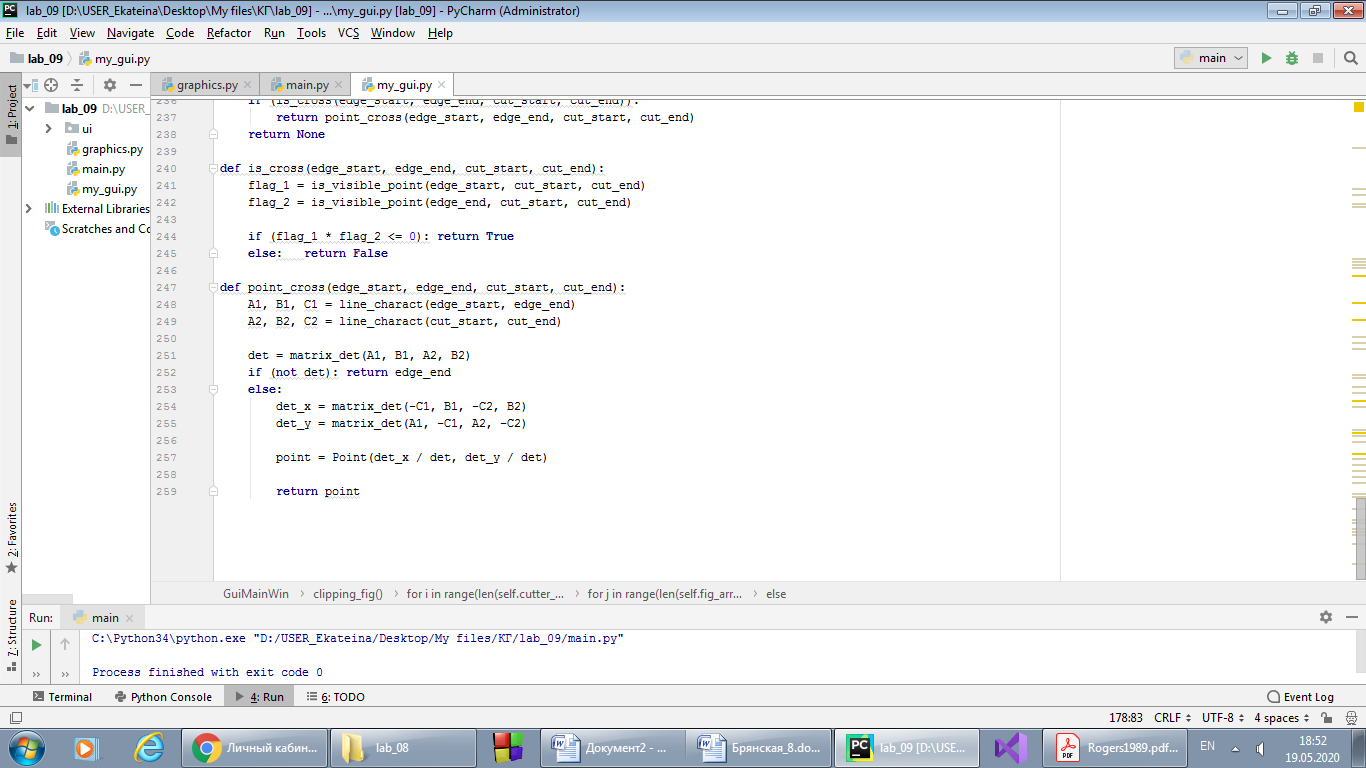
Проверка на факт пересечения:

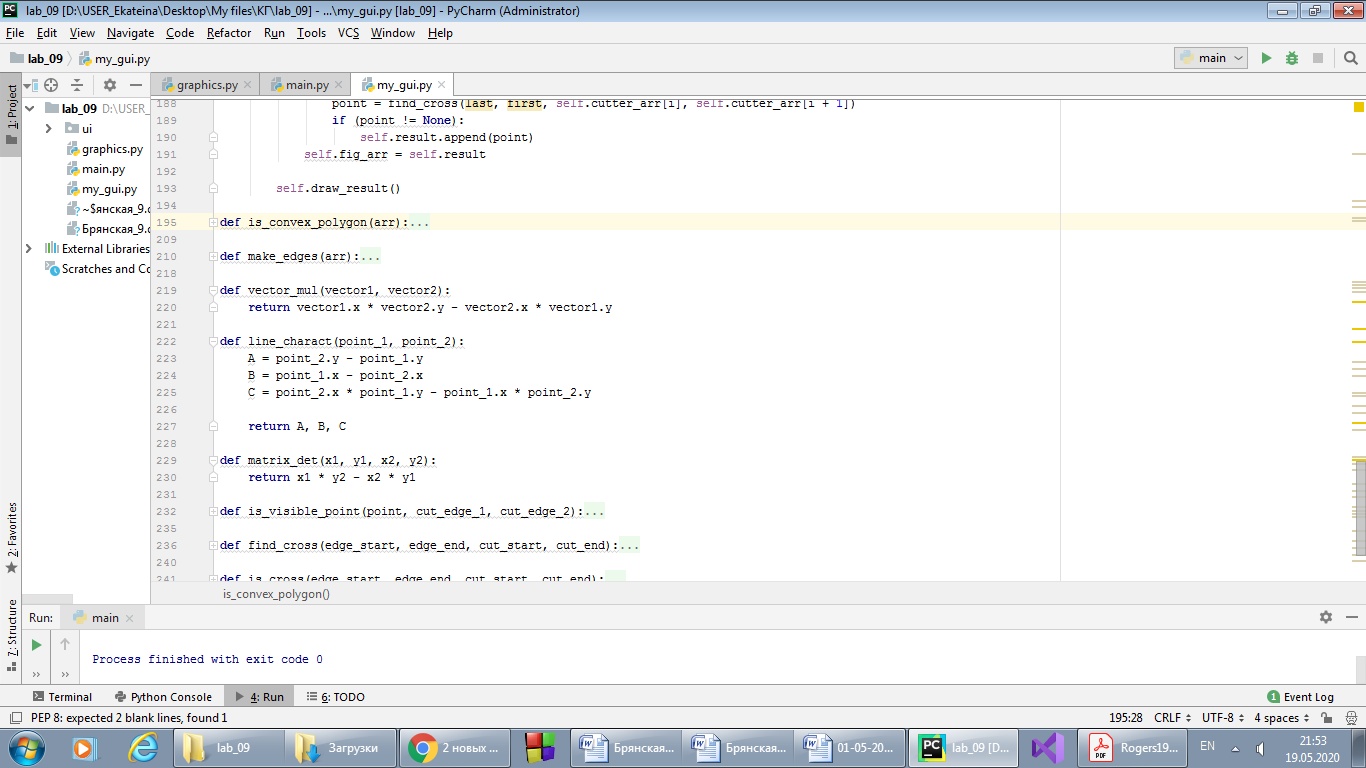


Определение видимости точки:

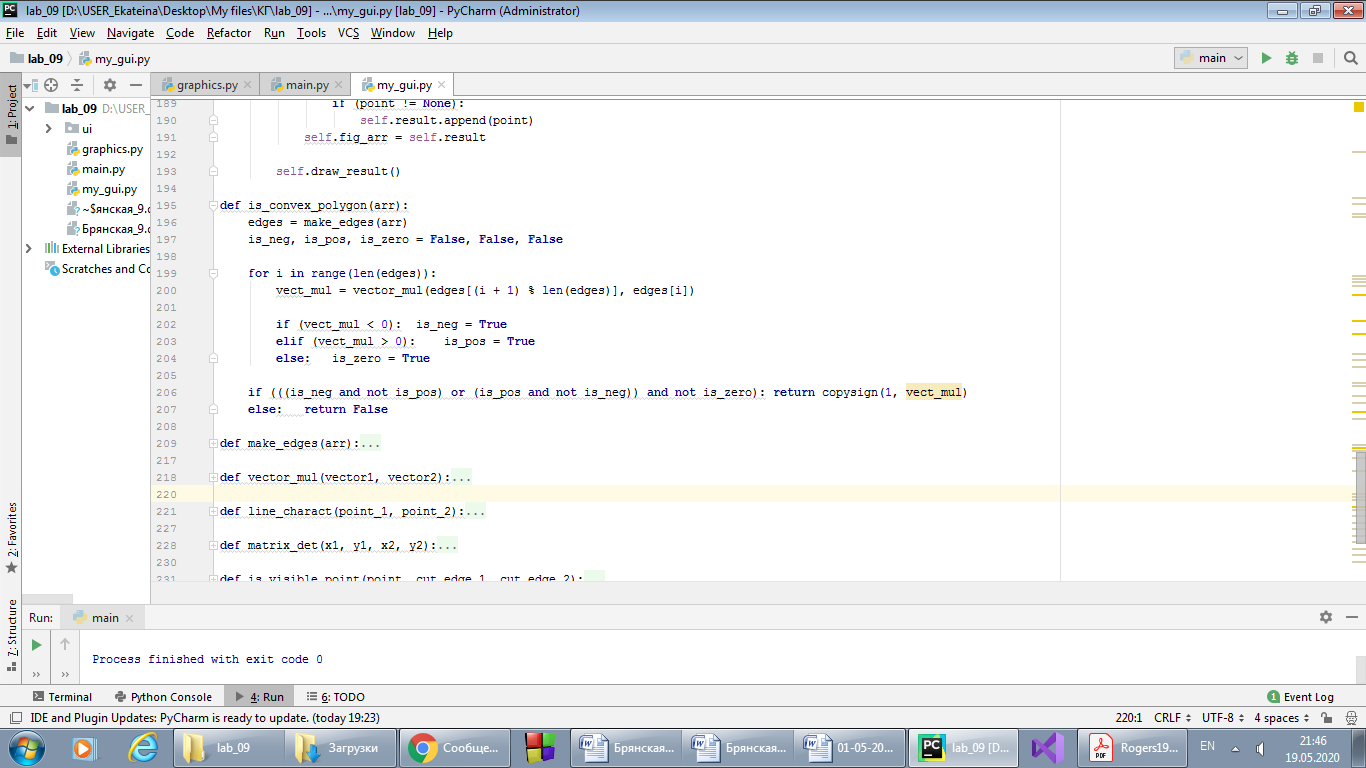


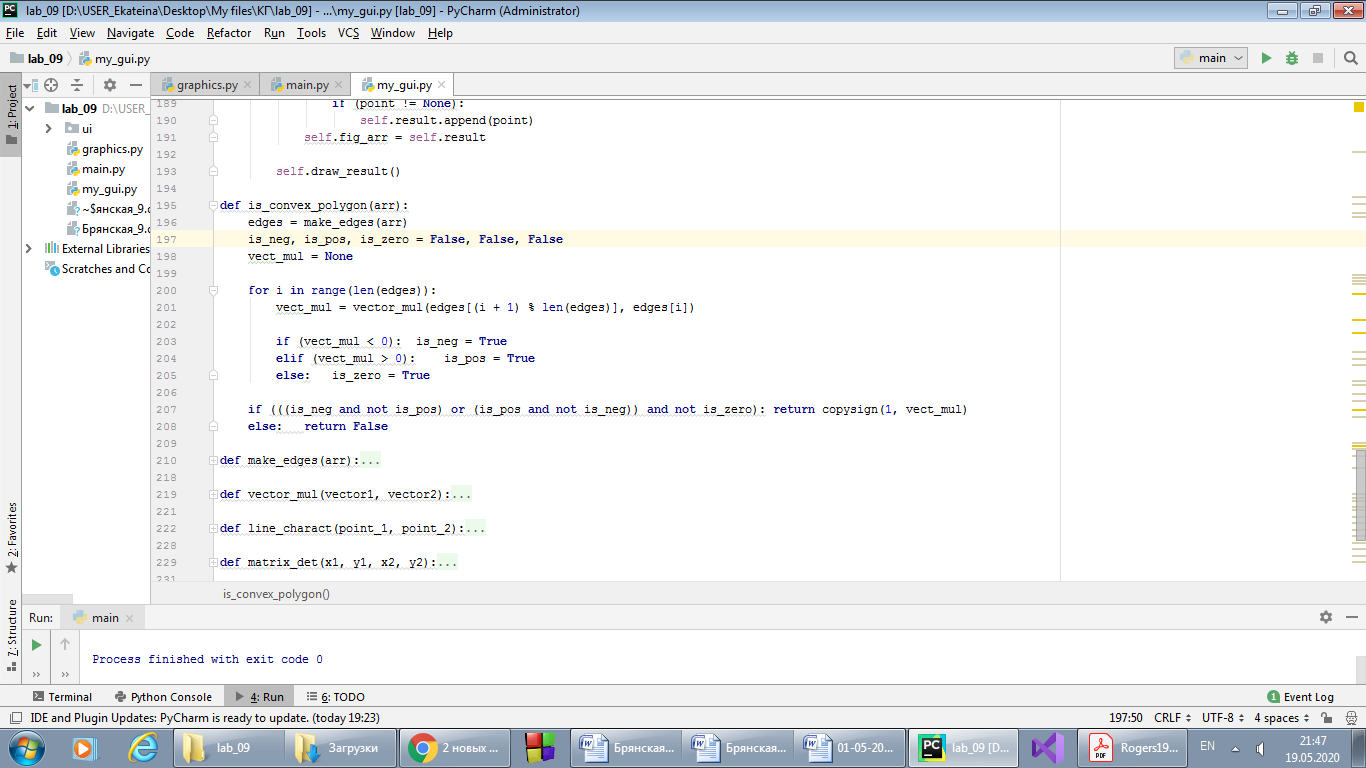
Нахождение точки пересечения:



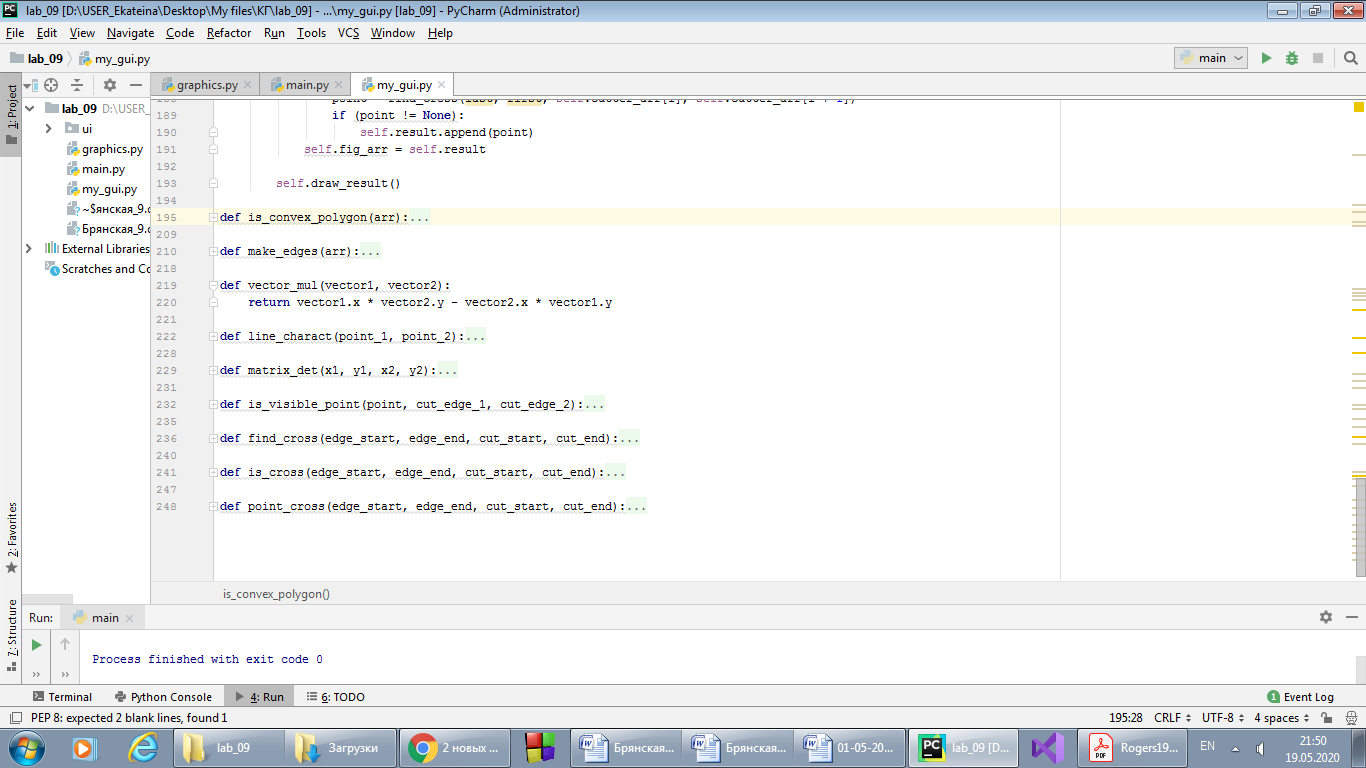


Определение выпуклости:





Векторное произведение:



*Примеры работы:*

|  |  |
| --- | --- |
|  | **91.png** |
|  |  |
|  |  |
| Отсекаемый многоугольник внутри отсекателя | |
|  |  |
| Отсекатель лежит внутри отсекаемого многоугольника | |
|  |  |
|  | **92.png** |
|  |  |
|  |  |
| **93.png** | **94.png** |

**Вывод**

В ходе лабораторной работы был изучен и реализован алгоритм отсечения произвольного многоугольника выпуклым отсекателем (алгоритм Сазерленда-Ходжмена).