|  |  |
| --- | --- |
| 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 № \_\_5\_\_**

|  |  |
| --- | --- |
| **Тема РЕАЛИЗАЦИЯ И ИССЛЕДОВАНИЕ АЛГОРИТМОВ РАСТРОВОГО ЗАПОЛНЕНИЯ СПЛОШНЫХ ОБЛАСТЕЙ**  **Вариант 4 Алгоритм заполнения со списком ребер и флагом**  **Студент \_Брянская Е.В.\_\_\_\_\_\_\_\_**  **Группа \_ИУ7-42Б\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Оценка (баллы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель \_Куров А.В.** |  |

Москва.

2020 г.

**Цель работы:** реализация и исследование одного из алгоритмов (в соответствии с вариантом) растрового заполнения области.

Задание:

1. реализовать Алгоритм заполнения со списком ребер и флагом (вариант 4).
2. обеспечить замер времени выполнения алгоритма (без задержки, с выводом на экран только окончательного результата).

Необходимо предусмотреть:

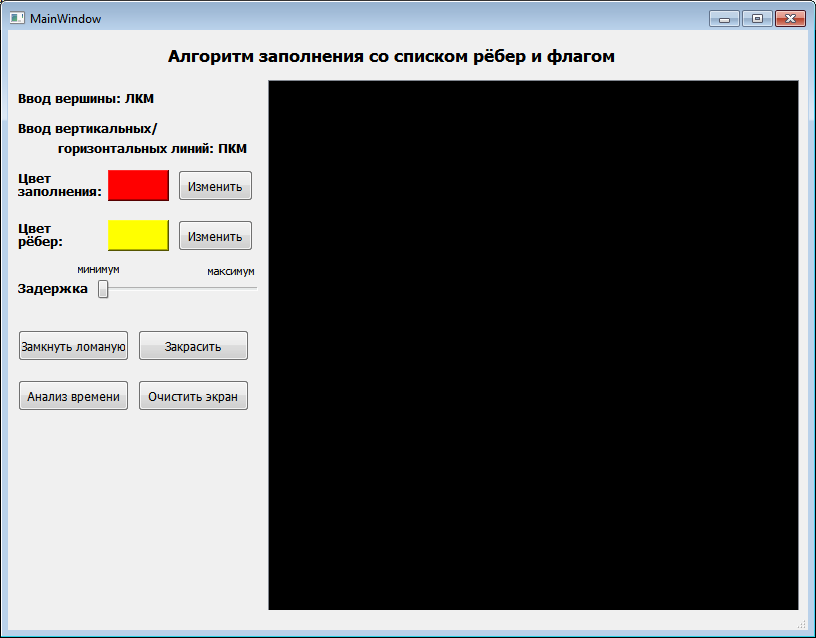
1. Необходимо обеспечить ввод произвольной многоугольной области, содержащей произвольное количество отверстий.

Ввод (вершин многоугольника) производить с помощью мыши, при этом для удобства пользователя должны отображаться ребра, соединяющие вводимые вершины. Предусмотреть ввод горизонтальных и вертикальных ребер.

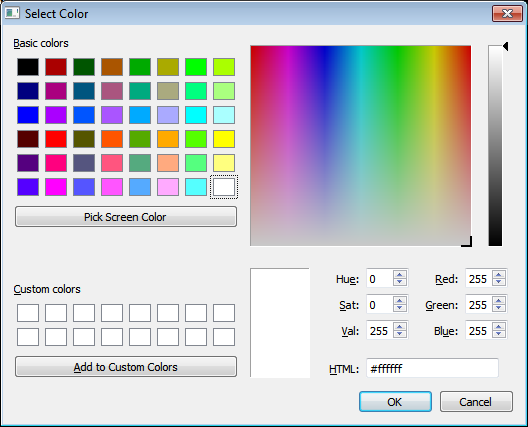
1. Пользователь должен иметь возможность задания цвета заполнения.
2. Работа программы должна предусматривать два режима – с задержкой и без задержки.
3. Режим с задержкой должен позволить проследить выполняемую последовательность действий. (Задержку целесообразно выполнять после обработки очередной строки).

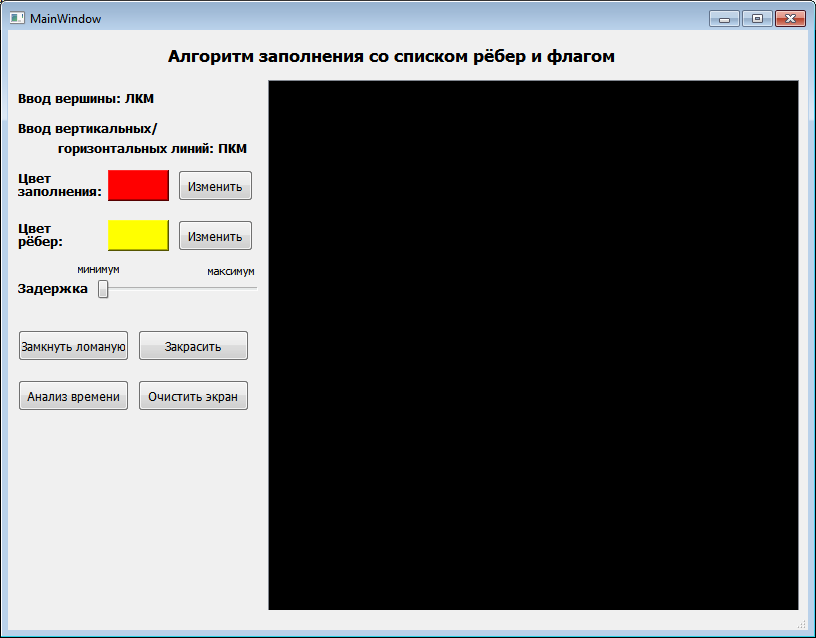
**Интерфейс**

Пользователю предоставляется следующий интерфейс для работы:

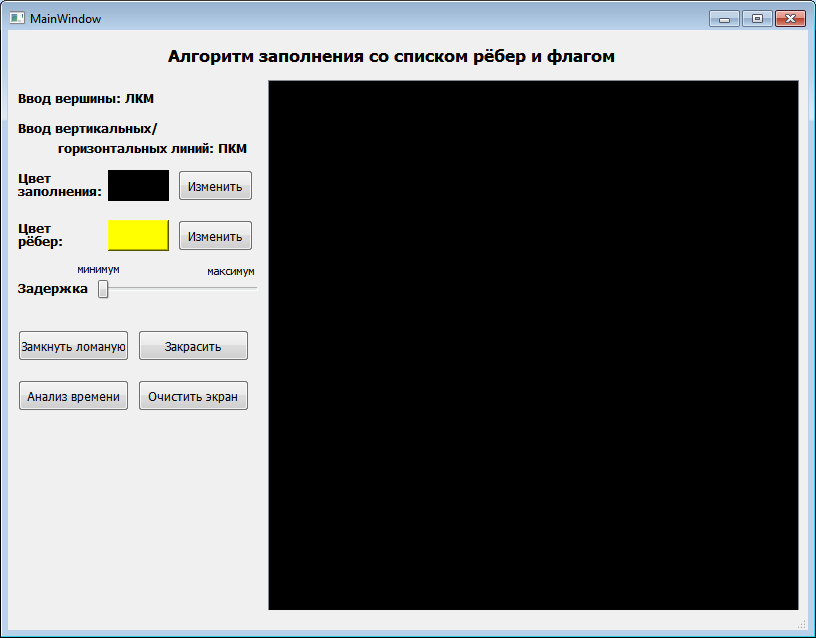


Ввод вершин осуществляется с помощью мыши, путём нажатия на черный холст. Если нажимается левая кнопка мыши, то рисуется соответствующее ребро, если пользователю нужно изобразить вертикальную или горизонтальную прямую, то он должен сместить курсор на необходимое расстояние и нажать правую кнопку мыши,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

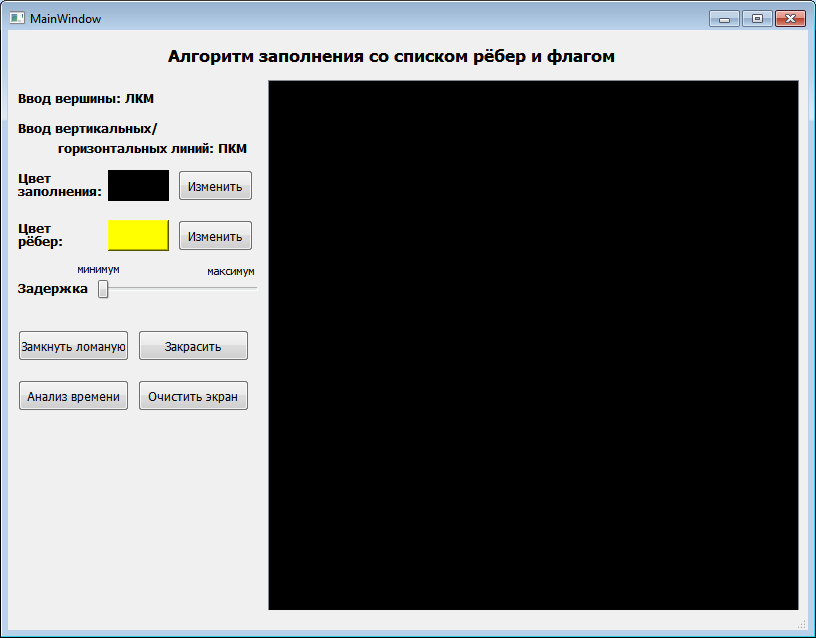
Пользователь может выбрать цвет заполнения и цвет вводимых рёбер:



Программа предоставляет возможность работы алгоритма с задержкой, которую можно регулировать с помощью соответствующего горизонтального ползунка. Если пользователю необходимо использовать режим без задержки нужно выставить значение ползунка на «минимум» (кроме того, изначально при запуске выбран этот параметр).



Пользователь может задавать программе следующие команды:



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.

Команда «Закрасить» запускает процесс заполнения области.

Ещё предусмотрены такие команды, как «Анализ времени» и «Очистить экран».

**Алгоритм заполнения со списком ребер и флагом**

*Об алгоритме*

Одной из отличительных черт данного алгоритма – наличие флага, который является признаком расположения пикселя внутри или вне многоугольника.

Сам алгоритм состоит из двух этапов, шагов. Первый шаг – отрисовывается контур, который ограничивает область. А второй – само заполнение (заполнение областей располагающихся между ограничивающими пикселями).

На первом этапе обрабатываются рёбра многоугольника, путём применения множества сканирующих строк. Производятся вычисления точек пересечения ребра и сканирующей строки (для наибольшей эффективности реализуется итеративный способ, чтобы избежать дополнительных затратных вычислений). После этого вычисляется самый левый пиксель, расположенный справа от точки пересечения.

Таким образом, получается контур фигуры.

На втором этапе осуществляется проход по сканирующим строкам, в процессе которого анализируется положение очередного пикселя. Флаг, равный лжи (False), является признаком того, что пиксель не принадлежит области заполнения, а если он равен истине (True), то наоборот, принадлежит. На основе этого признака пикселю присваивается либо цвет фона, либо цвет многоугольника. В начале программы флаг равен значению лжи (False).

Сам признак меняется, когда очередной пиксель имеет граничное значение.

*Важный момент*

На момент отрисовки контура может возникнуть следующая ситуация: при достаточно малых углах ребра могут “сливаться”, т.е. иметь 1 (или больше, если угол очень маленький по величине) общий пиксель. Это значит, что признак не изменится в нужный момент. Чтобы избежать этой проблемы, нужно в процессе работы проверить эту ситуацию, и если она возникает, то следует высветить соседний пиксель.

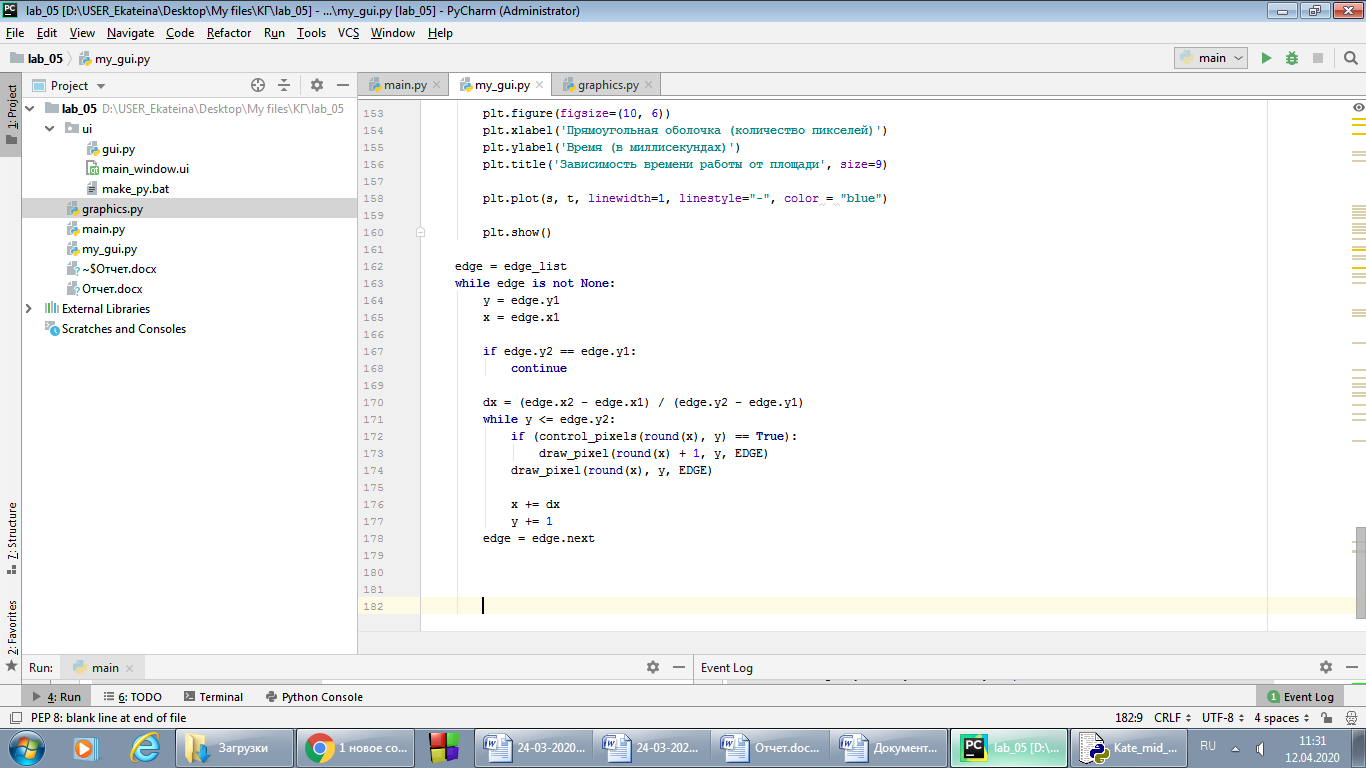
*Преимущества алгоритма*

Одно из главных преимуществ является то, что каждый пиксель обрабатывается только 1 раз (что очень сильно сокращает временные затраты по сравнению с другими алгоритмами).

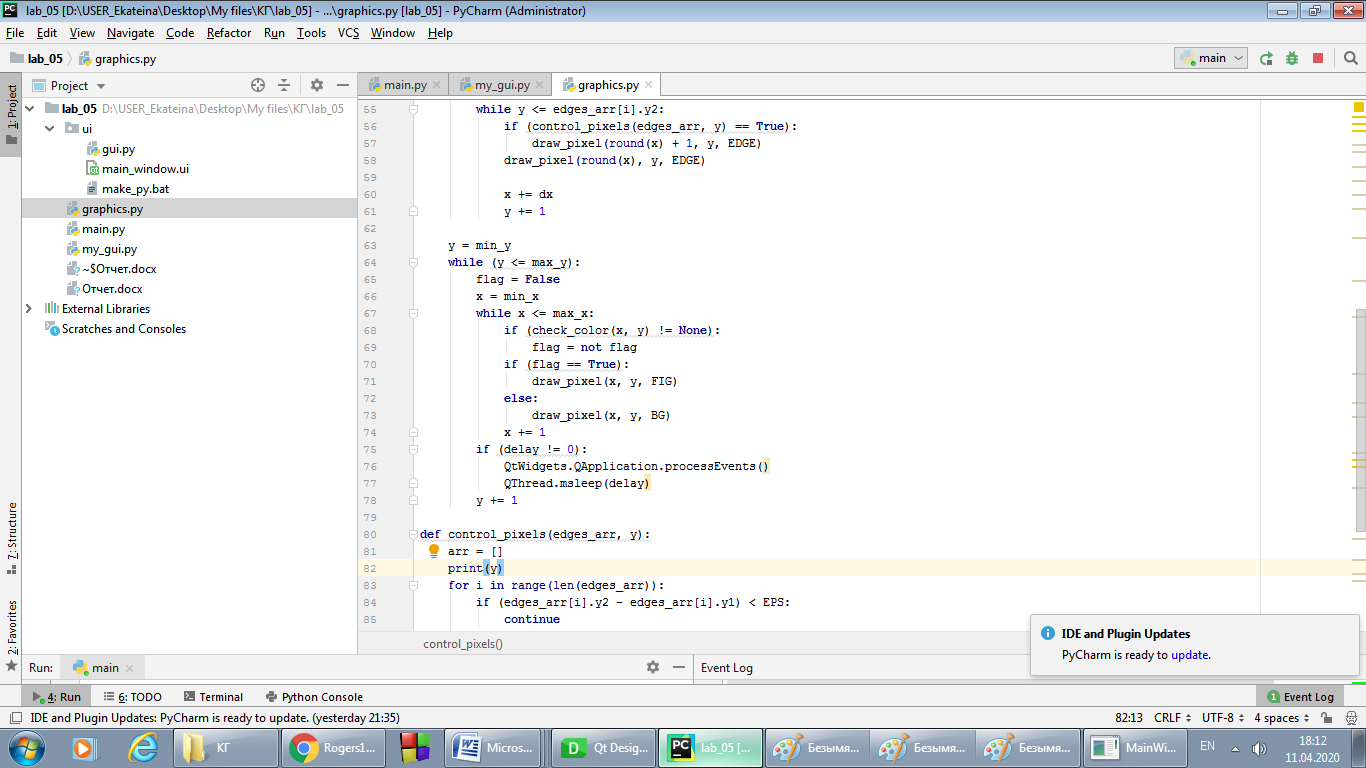
Кроме того, в этом алгоритме не требуется сортировка рёбер (следовательно, не тратится и время на это).

*Реализация:*

Отрисовка контура фигуры:



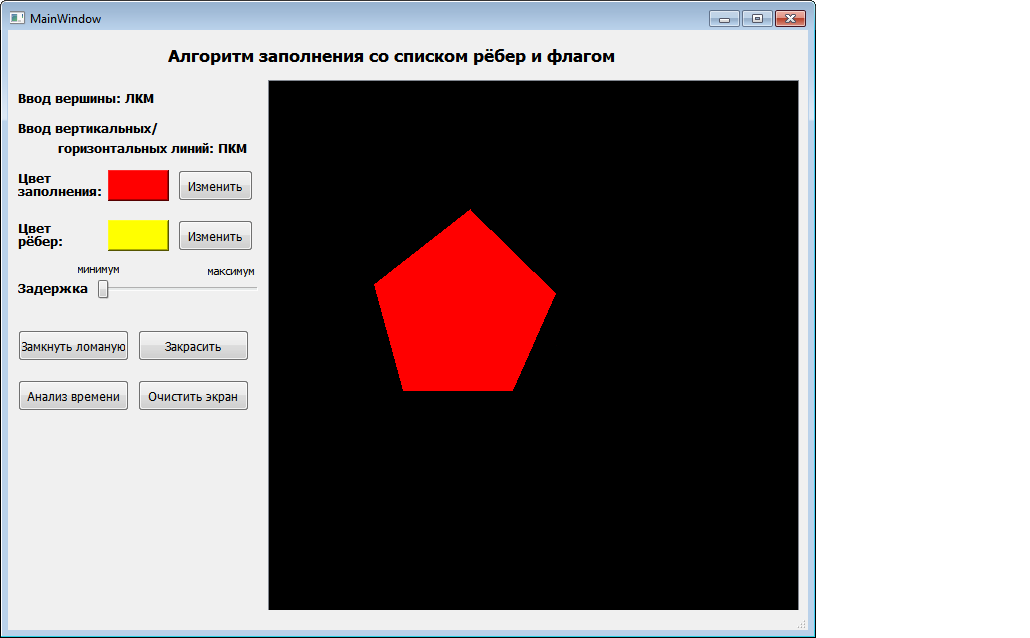
Заполнение (перед этим были вычислены значения min\_x/min\_y, max\_x/max\_y – параметры прямоугольной оболочки):



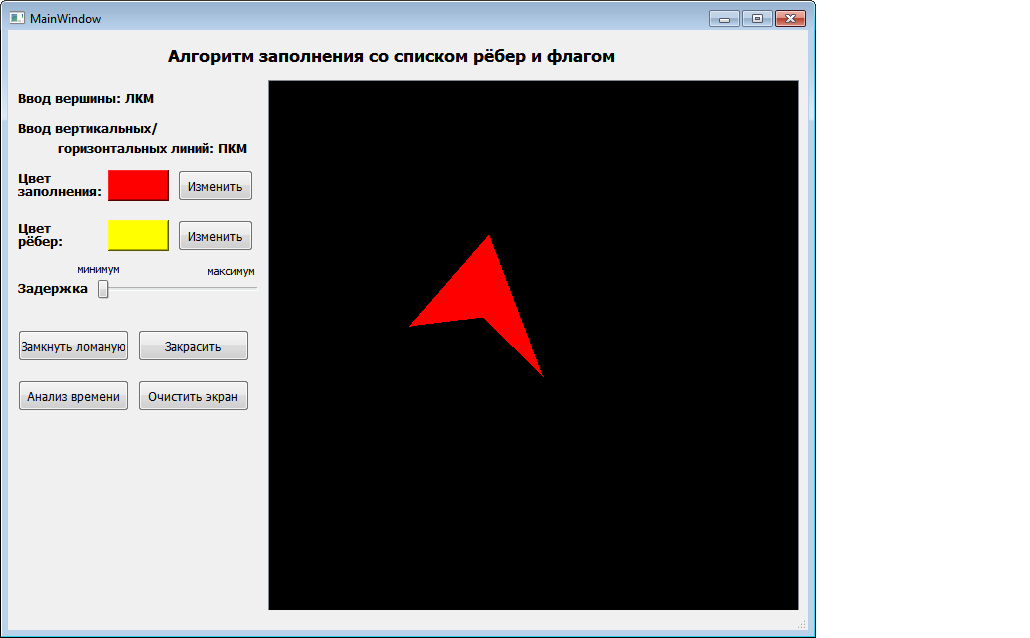
В конце условие – условие для задержек, которые можно задать в начале работы; при задержке, равной нулю, фигура будет отрисовываться в обычном режиме.

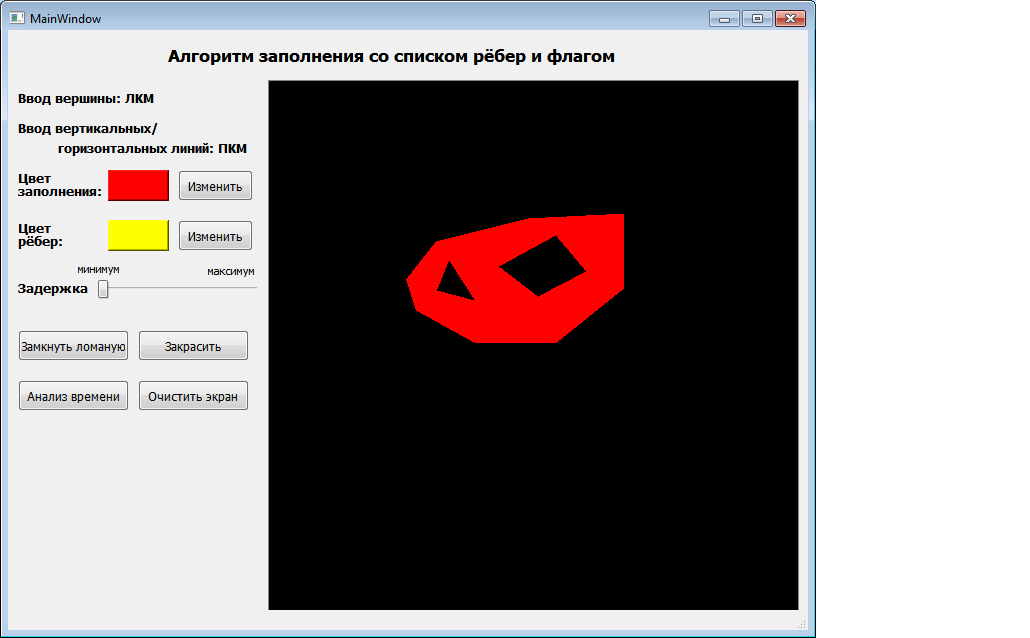
*Примеры работы:*

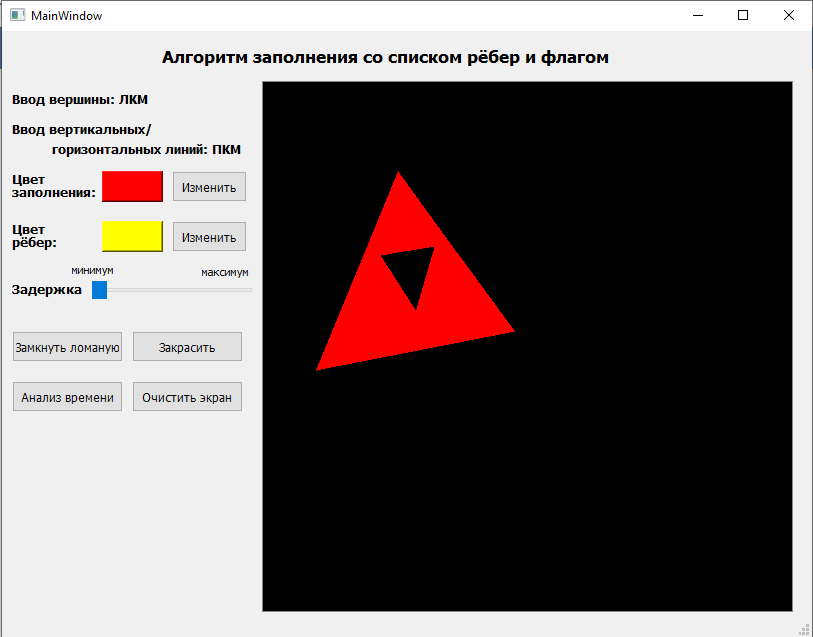
1. Выпуклая фигура

**

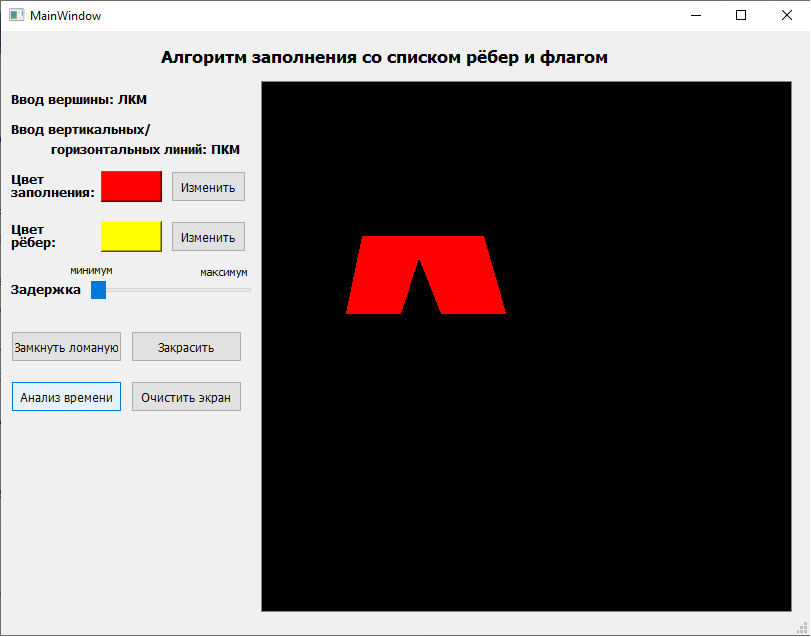
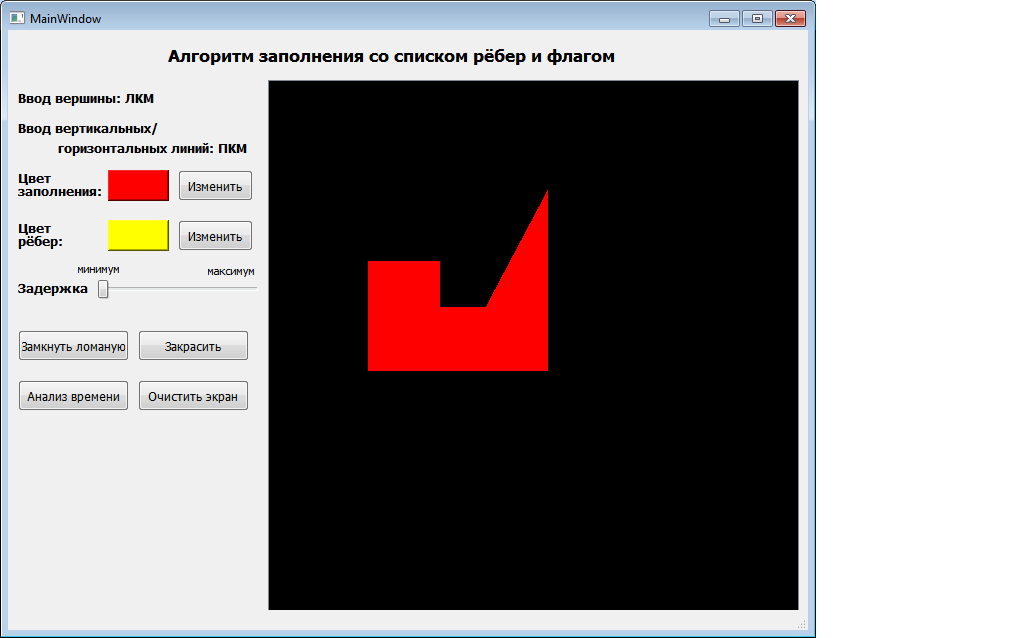
1. Невыпуклая фигура

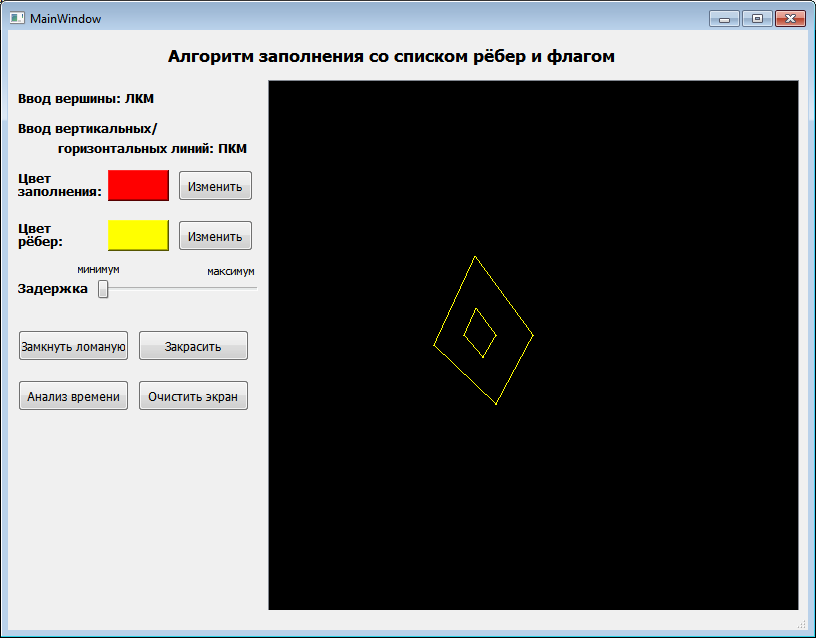


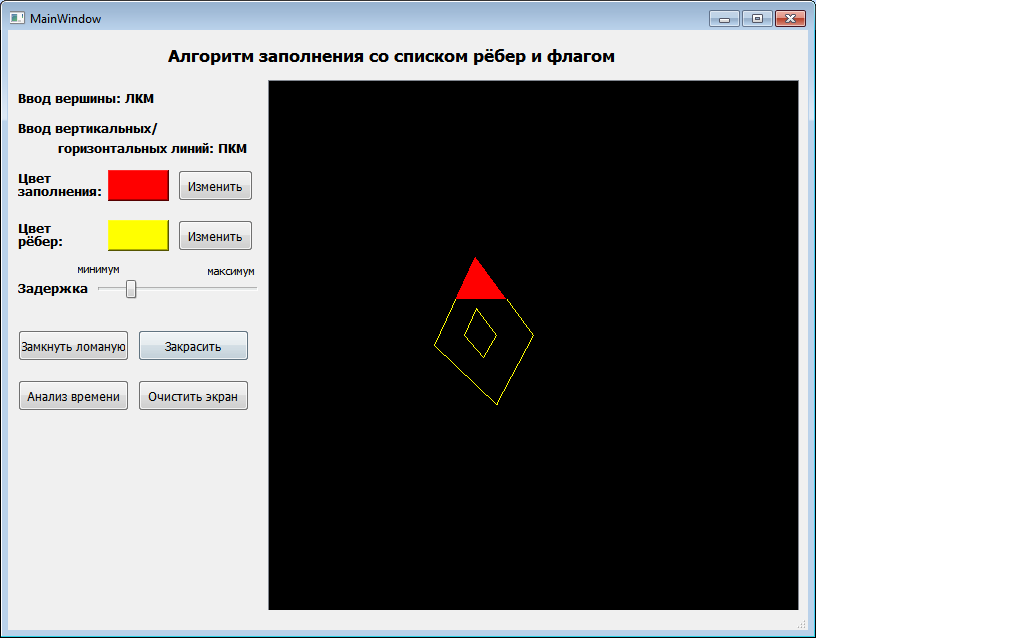
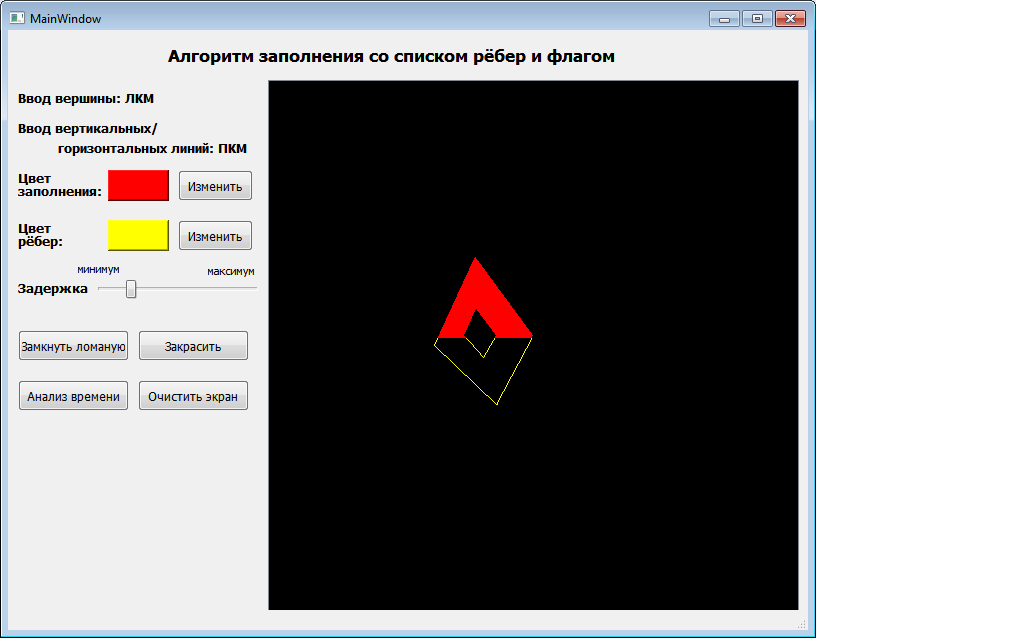
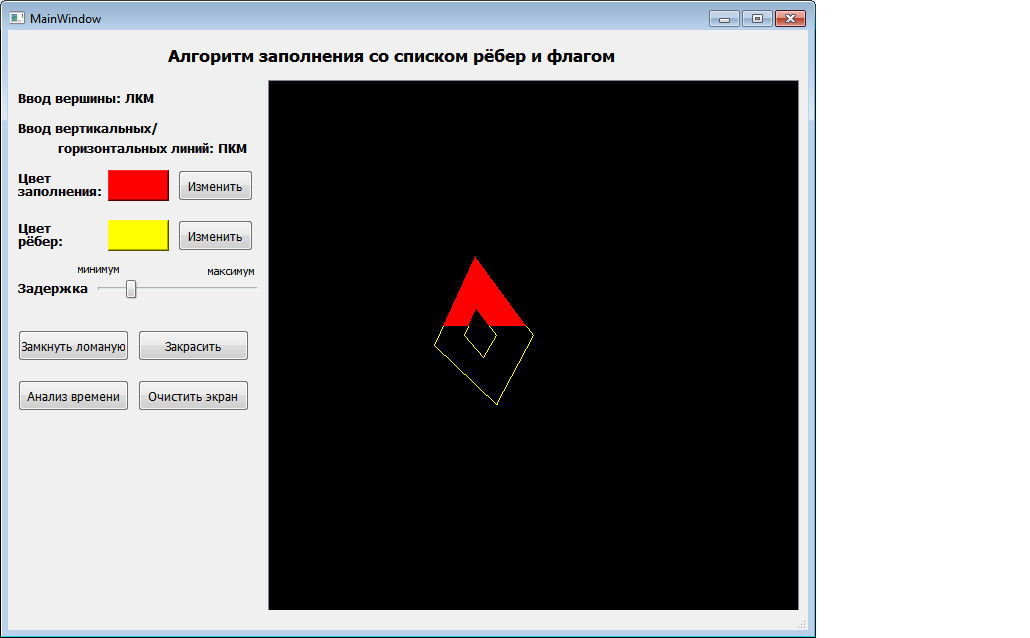
1. Фигура с отверстиями

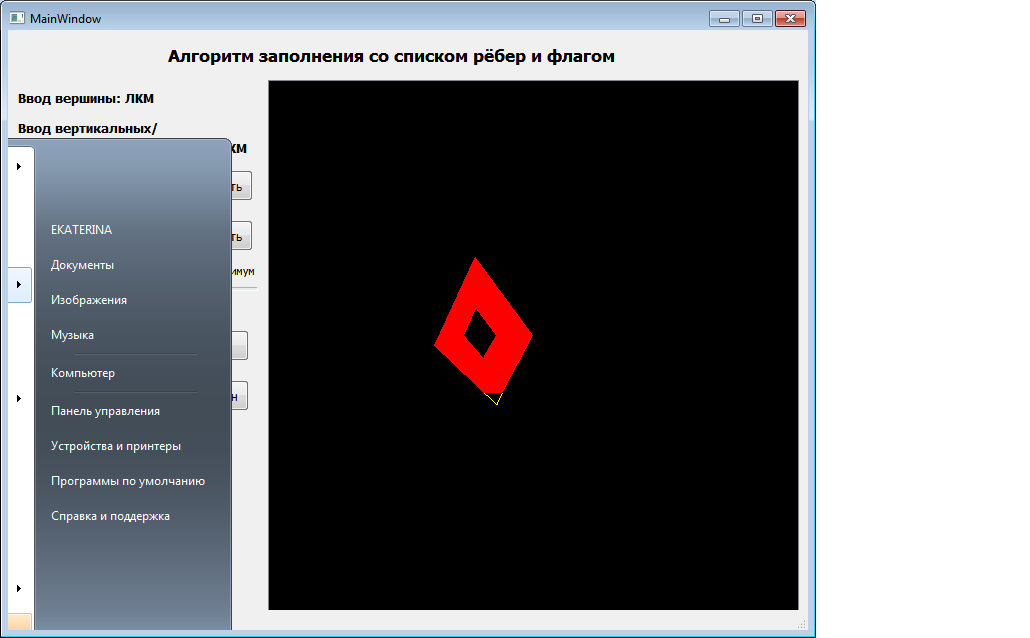


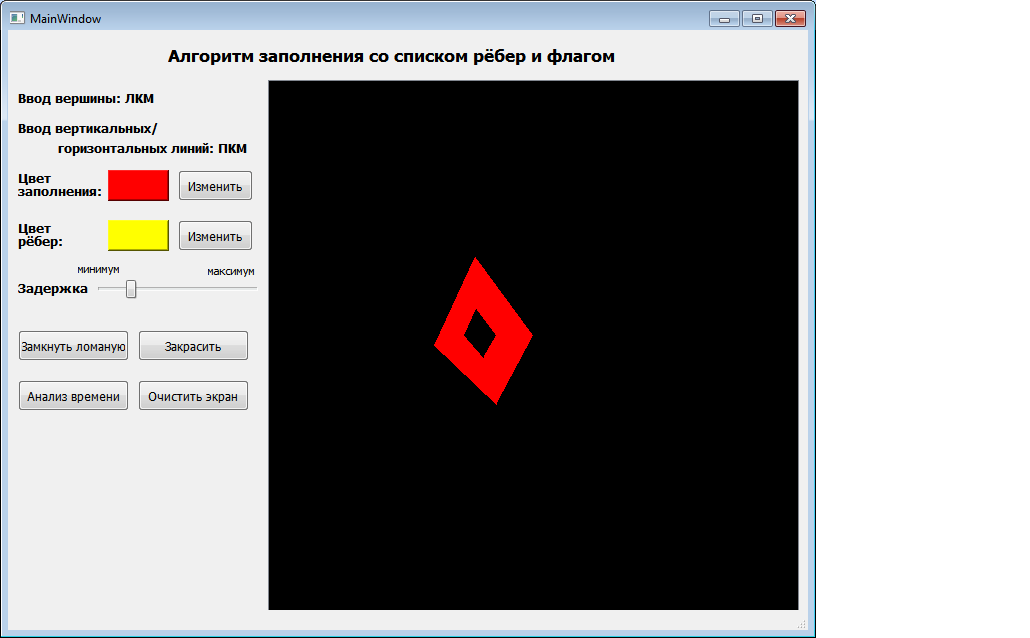
1. Фигура с горизонтальными/вертикальными ребрами



*Работа с задержкой:*

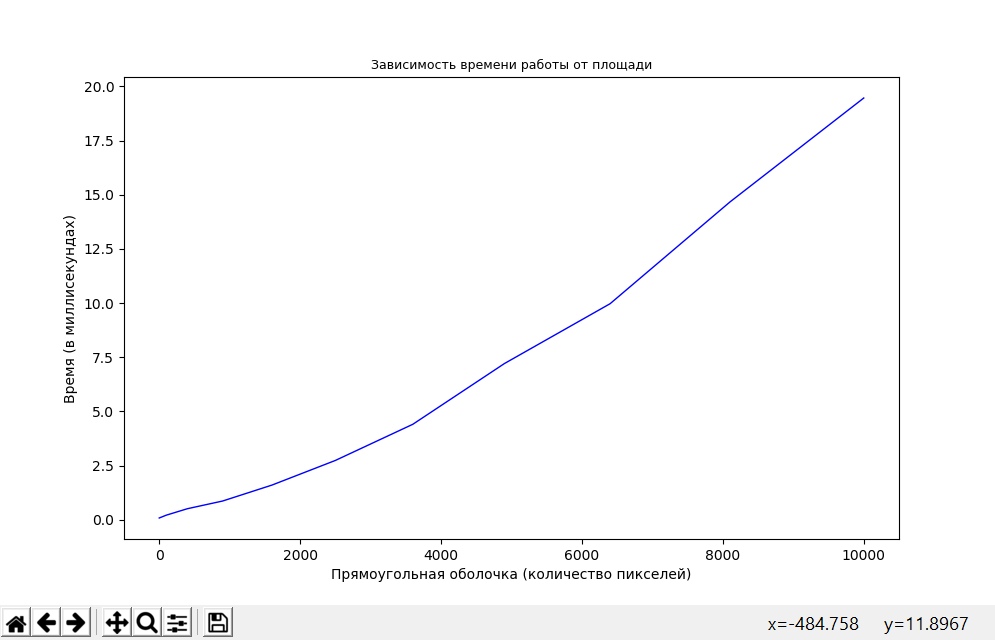
**



**

*Анализ времени:*

Производится анализ зависимости времени работы алгоритма от площади заданной фигуры. В качестве примера был взят квадрат. По оси ОХ указано значение площади.



На основе этого графика можно сделать вывод о том, что время работы линейно зависит от площади фигуры.

**Вывод**

В ходе лабораторной работы были изучены алгоритмы растрового заполнения областей и реализован алгоритм заполнения со списком рёбер и флагом, а также проведен его анализ. Во время работы были подтверждены его преимущества и рассмотрена проблемная ситуация (возникающая при малых углах), и предложено соответствующее её решение.