|  |  |
| --- | --- |
| 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 № 8**

|  |  |
| --- | --- |
| **Тема: Изучение и программная реализация алгоритма отсечения отрезка выпуклым отсекателем (Алгоритм Кируса-Бека).**  **Студент** ВоякинА. Я.  **Группа ИУ7-44Б**  **Оценка (баллы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |  |

Москва.

2020 г.

**Цель работы:**

Изучение и программная реализация алгоритма отсечения отрезка выпуклым отсекателем (Алгоритм Кируса-Бека).

**Условие задания:**

1. Необходимо обеспечить ввод отсекателя – произвольного прямоугольника. Высветить его первым цветом.
2. Необходимо 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. Выполнить отсечение отрезков, показав их результат третьим цветом. Исходные отрезки не удалять.

**Описание алгоритма:**

1. Ввод P1, P2, K, C(K)
2. tн = 0, tв = 1
3. D = P2 – P1
4. Проверка отсекателя на выпуклость
5. Цикл отсечения (по I от 1 до K)
   1. Wi = P1 – fi
   2. Вычисление Dск, Wск
   3. Если Dск = 0, то проверить если Wск < 0, то отрезок невидим
   4. t = -Wск / Dcк
   5. Если Dск > 0, то проверить если t > 1, то отрезок невидим, иначе tн = max(t, tн)
   6. Если Dск <= 0, то проверить если t < 0, то отрезок невидим, иначе tв = min(t, tв)
   7. Конец цикла
6. Если tн < tв, то изобразить отрезок (tн, tв)
7. Конец.

**Инструкция к программе:**

Рисование отсекателя происходит с помощью правой кнопки мыши.

Соединение первой точки отсекателя и последней происходит при нажатии клавиши Shift.

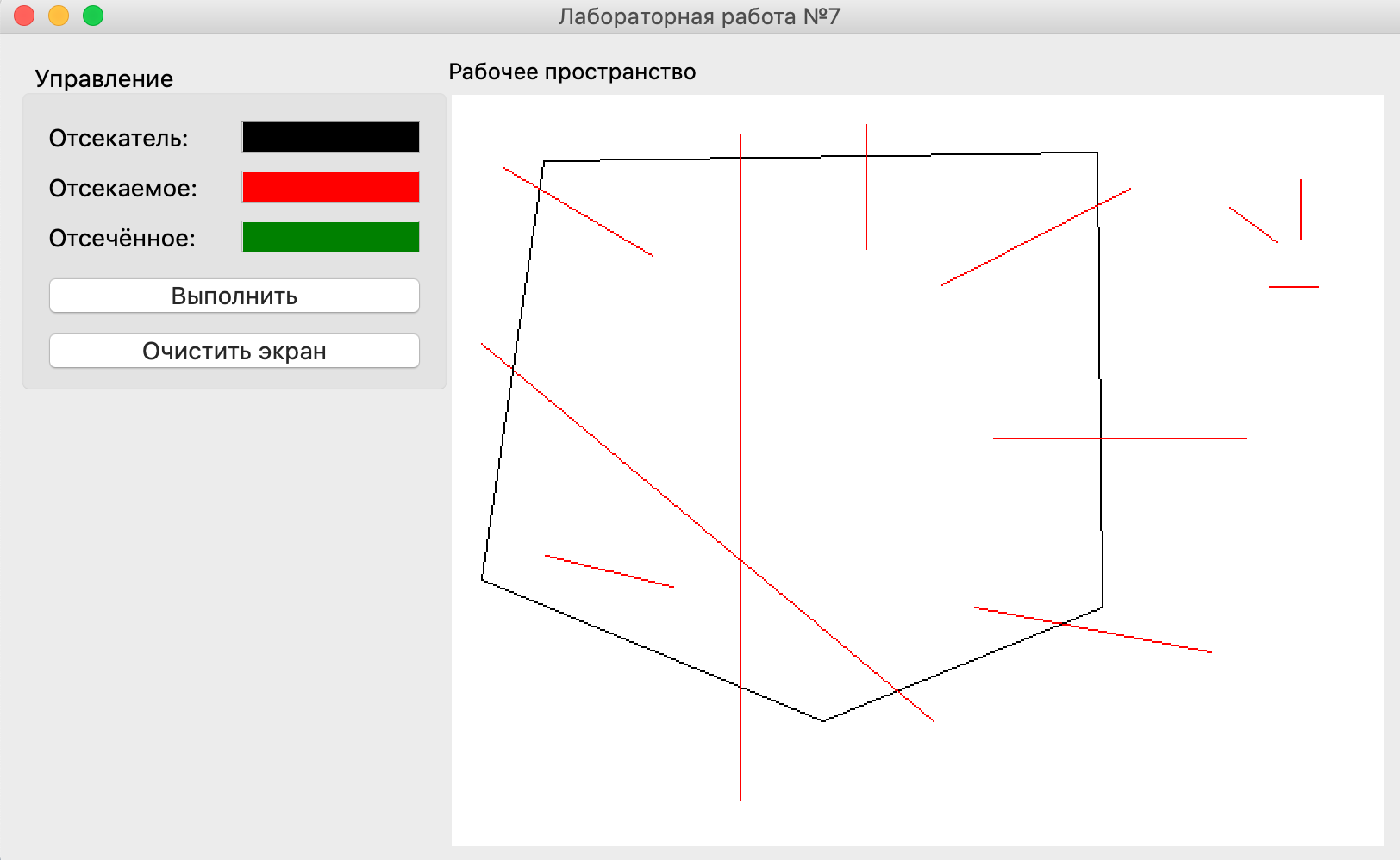
Рисование линий происходит с помощью левой кнопки мыши.

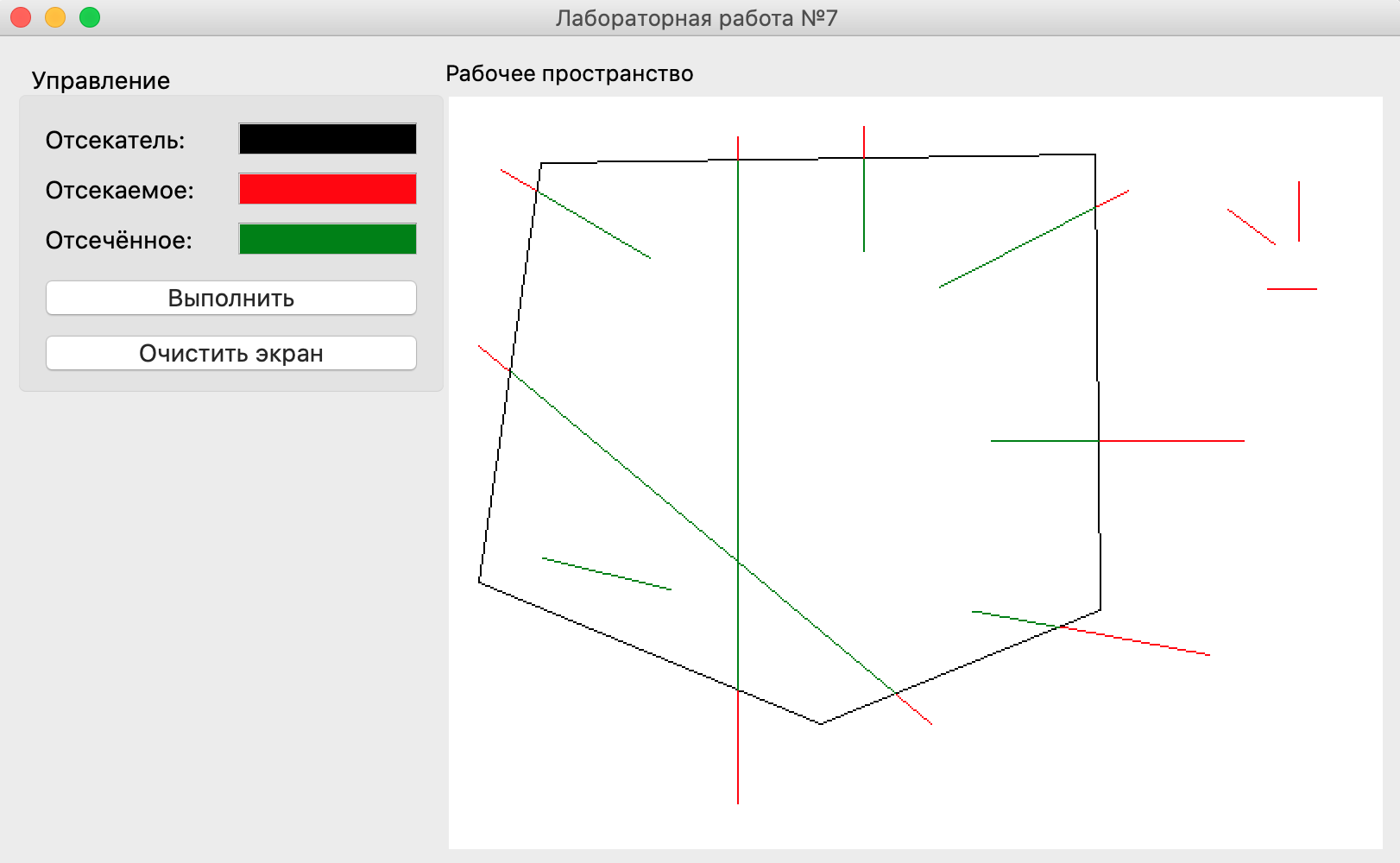
Рисование вертикальных и горизонтальных линий происходит с помощью левой кнопки мыши с зажатыми клавишами «V» и «H» соответственно. Выбор цвета происходит по нажатии на прямоугольник с цветом, который хотите поменять.

**Код алгоритма.**

def makeCut(*self*, p1\_x, p1\_y, p2\_x, p2\_y, obh):  
 t\_begin = 0  
 t\_end = 1  
 t = 0  
 d\_x = p2\_x - p1\_x  
 d\_y = p2\_y - p1\_y  
 for i in range(0, len(*self*.cutter\_x)):  
 w\_x = p1\_x - *self*.cutter\_x[i]; w\_y = p1\_y - *self*.cutter\_y[i]  
 n\_x = (-*self*.cutter\_y[(i + 1) % len(*self*.cutter\_y)] + *self*.cutter\_y[i]) \* obh  
 n\_y = (*self*.cutter\_x[(i + 1) % len(*self*.cutter\_x)] - *self*.cutter\_x[i]) \* obh  
 Wsc = w\_x \* n\_x + w\_y \* n\_y  
 Dsc = d\_x \* n\_x + d\_y \* n\_y  
 if Dsc == 0:  
 goto .q2  
 t = - Wsc / Dsc  
 if Dsc > 0:  
 goto .q1  
 if t < 0:  
 goto .q4  
 t\_end = min(t, t\_end)  
 goto .q3  
 label .q1  
 if t > 1:  
 goto .q4  
 t\_begin = max(t, t\_begin)  
 goto .q3  
 label .q2  
 if Wsc < 0:  
 goto .q4  
 label .q3  
 if t\_begin > t\_end:  
 goto .q4  
 *self*.bres\_int(p1\_x + (p2\_x - p1\_x) \* t\_end, p1\_y + (p2\_y - p1\_y) \* t\_end,  
 p1\_x + (p2\_x - p1\_x) \* t\_begin, p1\_y + (p2\_y - p1\_y) \* t\_begin, *self*.colour\_line\_in)  
  
 label .q4  
 return

**Демонстрация работы программы.**

****

****