МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №2
по дисциплине «Компьютерная графика»
Тема: «Примитивы OpenGL»

Студент гр. 6381 _____ Фиалковский М.С.

Студент гр. 6381 _____ Афийчук И.И.

Преподаватель _____ Герасимова Т.В.

Санкт-Петербург
2019

Задание.

Разработать программу, реализующую представление тестов отсечения (glScissor), прозрачности (glAlphaFunc), смешения цветов (glBlendFunc) в библиотеке OpenGL на базе разработанных вами в предыдущей работе примитивов.

Общие сведения.

Включение/выключение различных возможностей, изменение состояния OpenGL осуществляется при помощи двух команд - glEnable и glDisable, одна из которых включает, а вторая выключает некоторый режим.

```
void glEnable(GLenum cap)
void glDisable(GLenum cap)
```

Обе команды имеют один аргумент – cap, который может принимать значения, определяющие ту или иную настройку, например, GL_ALPHA_TEST, GL_BLEND, GL_SCISSOR_TEST и многие другие.

Тест отсечения

Режим **GL_SCISSOR_TEST** отсекает ту часть, которая находятся вне прямоугольника "вырезки". Положение и размеры данного прямоугольника устанавливаются функцией glScissor:

```
void glScissor( GLint x, GLint y, GLsizei width, GLsizei height );
```

Тест прозрачности

Режим **GL_ALPHA_TEST** задает тестирование по альфа-каналу(который определяет непрозрачность).

```
void glAlphaFunc( GLenum func, GLclampf ref )
```

где func может принимать следующие значения:

GL_NEVER – никогда не пропускает
GL_LESS – пропускает, если входное значение альфа меньше, чем значение ref
GL_EQUAL – пропускает, если входное значение альфа равно значению ref
GL_LEQUAL – пропускает, если входное значение альфа меньше или равно значения ref
GL_GREATER – пропускает, если входное значение альфа больше, чем значение ref
GL_NOTEQUAL – пропускает, если входное значение альфа не равно значению ref
GL_GEQUAL – пропускает, если входное значение альфа больше или равно значения ref
GL_ALWAYS – всегда пропускается, по умолчанию,

а ref определяет значение, с которым сравнивается входное значение альфа.

Однако в версиях OpenGL 3.1 и выше данный режим считается устаревшим и не используется, т.к. пользователю OpenGL необходимо самому писать фрагментный шейдер, с помощью оператора discard которого реализовать вручную один из необходимых режимов работы альфа-тестирования не так и сложно.

Тест смешения цветов

Режим GL_BLEND разрешает смешивание поступающих значений цветов RGBA со значениями, находящимися в буфере цветов.
Функция glBlendFunc устанавливает пиксельную арифметику.
void glBlendFunc(GLenum sfactor, GLenum dfactor);
где параметры

sfactor устанавливает способ вычисления входящих факторов смешения RGBA.

dfactor устанавливает способ вычисления факторов смешения RGBA, находящихся в цветовом буфере.

Оба параметра могут принимать следующие значения:

GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA, GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, GL_ONE_MINUS_CONSTANT_ALPHA, GL_SRC_ALPHA_SATURATE, GL_SRC1_COLOR, GL_ONE_MINUS_SRC1_COLOR, GL_SRC1_ALPHA, and GL_ONE_MINUS_SRC1_ALPHA.

Прозрачность лучше организовывать, используя команду glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA). Такой же вызов применяют для устранения ступенчатости линий и точек. Для устранения ступенчатости многоугольников применяют вызов команды glBlendFunc(GL_SRC_ALPHA_SATURATE, GL_ONE).

Ход работы.

За основу была взята и доработана программа из первой работы.

Сборка и компиляция программы осуществляется с помощью утилиты cmake и компилятора gcc из пакета MinGW. Графический интерфейс выполнен с помощью библиотеки FLTK. В качестве обертки над библиотекой OpenGL, используется GLEW.

При запуске программы создаётся окно AppWindow – базовое окно приложения, унаследованное от Fl_Window. Внутрь него методом композиции вложены окно GlSubWin (в котором будет отображаться вся выводимая графики), унаследованное от Fl_Gl_Window, и меню конфигурации. Это окно из библиотеки FLTK создано специально для работы с pure OpenGL calls.

Создание требуемого для работы контекста передаётся внутренним механизмам библиотеки. Для вызова команд OpenGL требуется переопределить виртуальный метод draw() в GlSubWin и делать это внутри него.

Передача конфигурационных параметров изображения реализовано с помощью вспомогательного класса State. Для каждого отдельного изображения (типа теста) создаётся свой независимый от других класс наследник State, который затем помещается в общий контейнер состояний. В созданном классе определяются передаваемые из меню параметры и логика их изменения. Также в нём определяются параметры и наполнение самого меню конфигурации, видимого пользователю.

Отрисовка текущего изображения (типа теста) происходит с помощью вызова требуемой функции из вектора функторов. Вызовы этих функторов происходят внутри метода draw окна GlSubWin. Для добавления нового такого функтора следует пронаследоваться от абстрактного класса IPainter и переопределить оператор() с определённым набором передаваемых аргументов и затем добавить его в список функторов в правильном порядке.

Точки для дальнейшей отрисовки изображений не хранятся в отдельном месте, а создаются генератором случайных чисел каждый раз при обновлении какого-либо параметра изображения или при переключении этих самых изображений.

Во всех примитивах используется похожая схема рисования:

```
glBegin(GL_НАЗВАНИЕ_ПРИМИТИВА);
# цикл по количеству точек
	applyColor(…); // вызов функции glColor(r,g,b,alpha) для установки цвета
	… // генерация x и y координаты точки по определённому для примитива алгоритму
	glVertex2f(x, y); // определение точки
glEnd();
```

OpenGL распознает точки между «операторными скобками» и строит примитив в зависимости от переданного аргумента в glBegin(…).

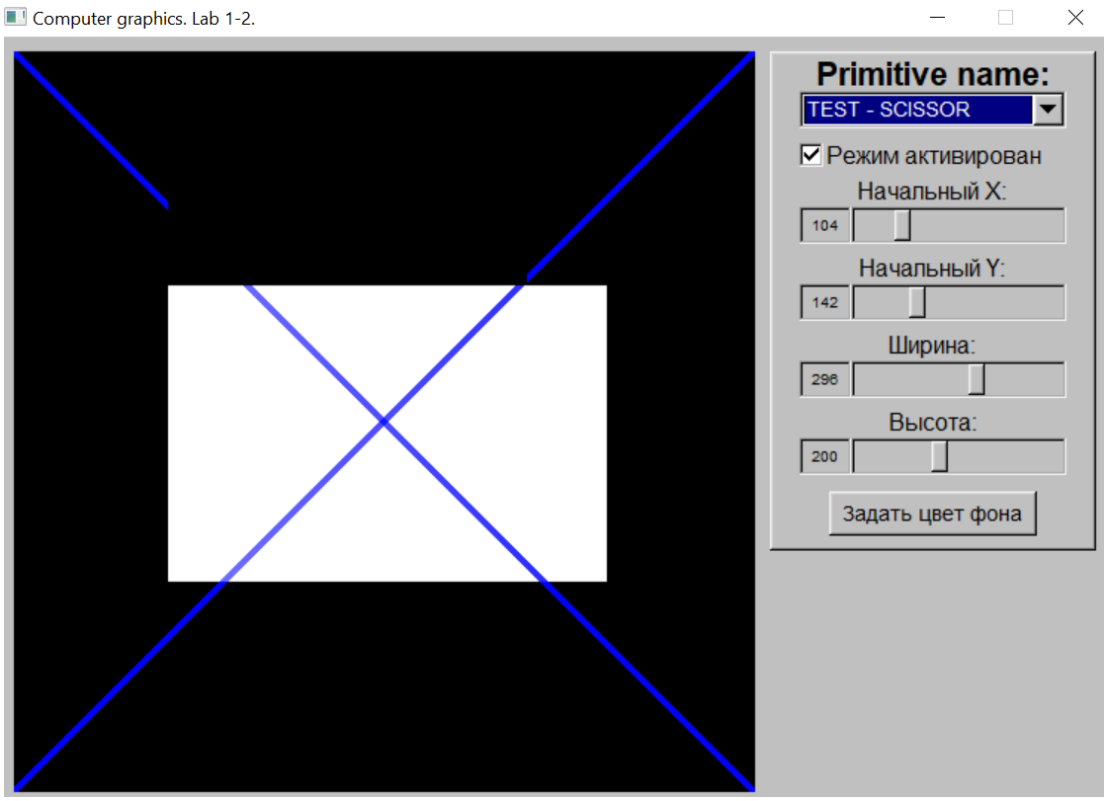
Переключаться между примитивами можно с помощью букв.

Хранение состояний тестов реализовано с помощью статических переменных класса State, поэтому их работа будет видна и на примитивах из первой лабораторной.

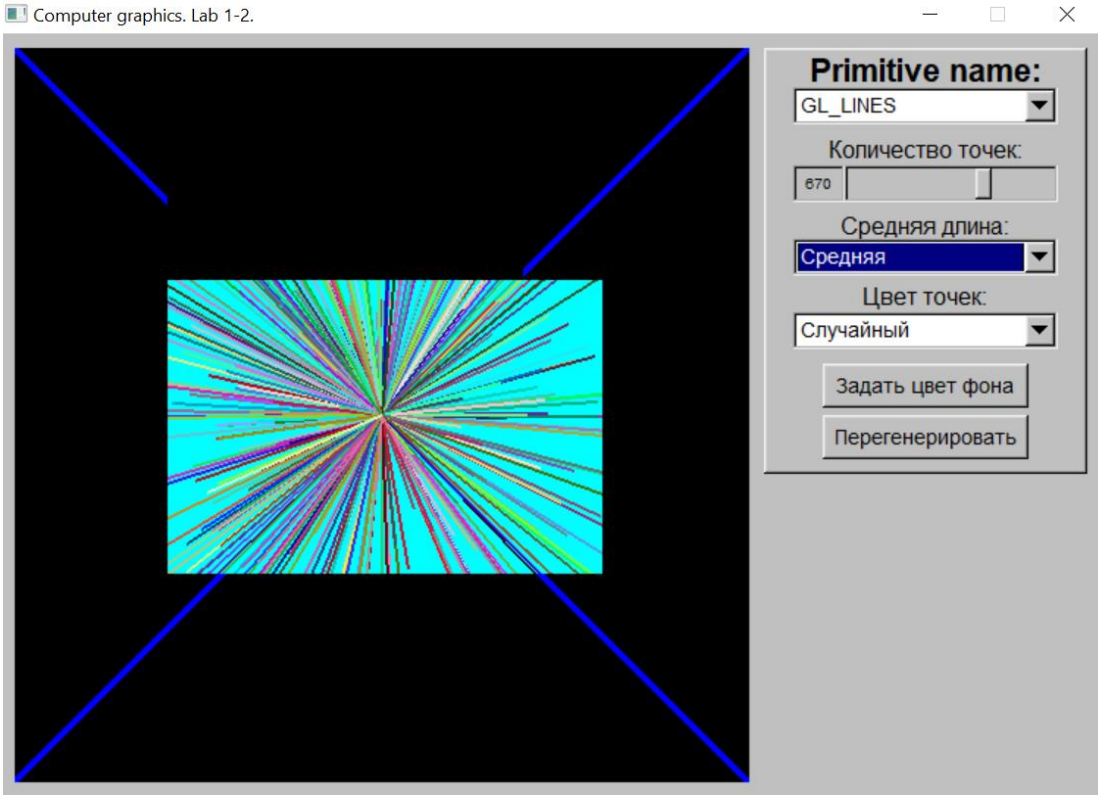
Тестирование.

Результаты тестирования представлены на изображениях:

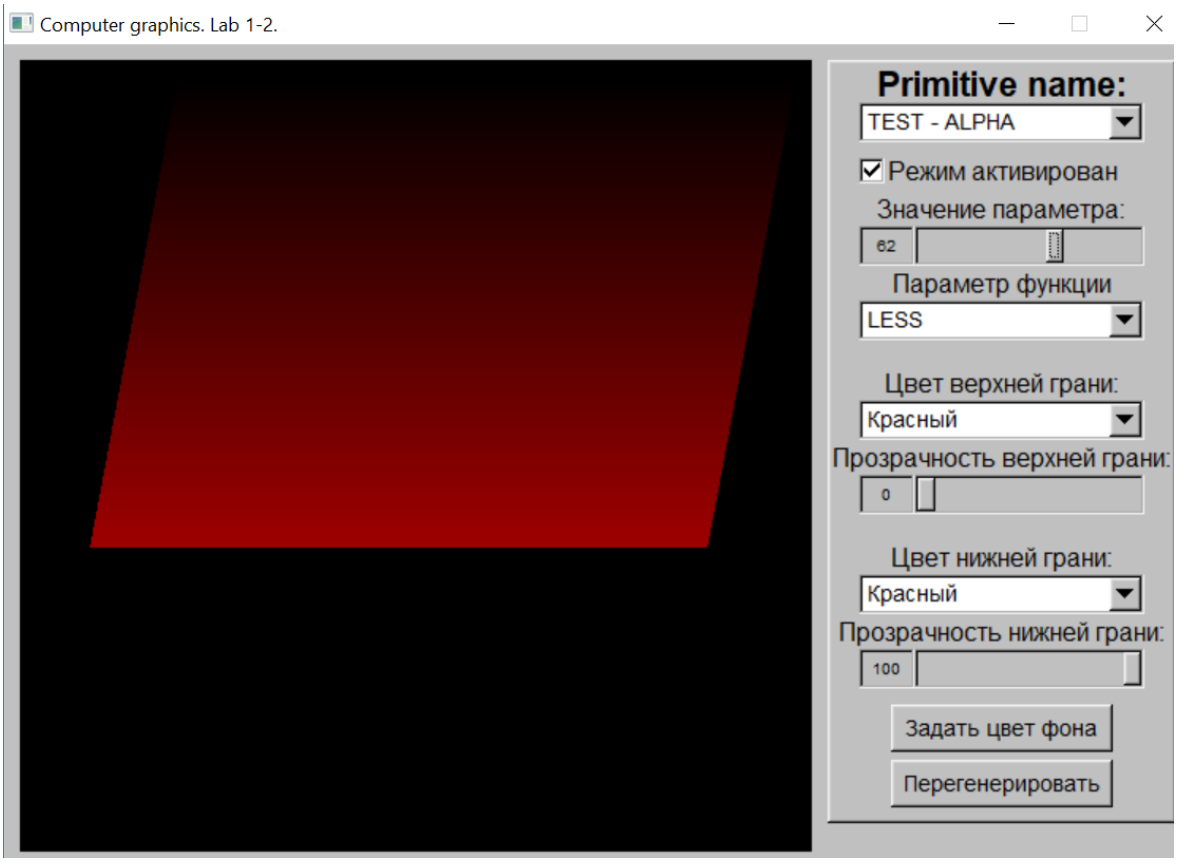
SCISSOR



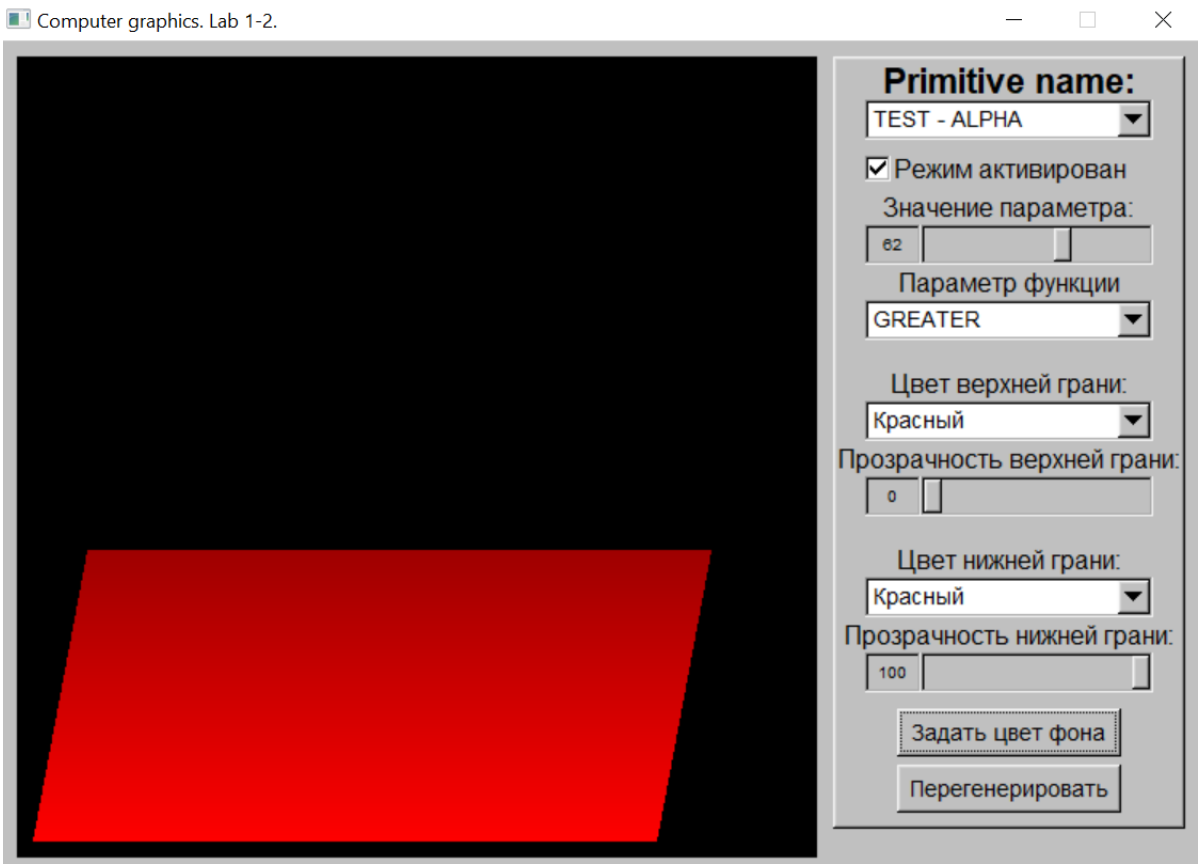
SCISSOR with LINES:



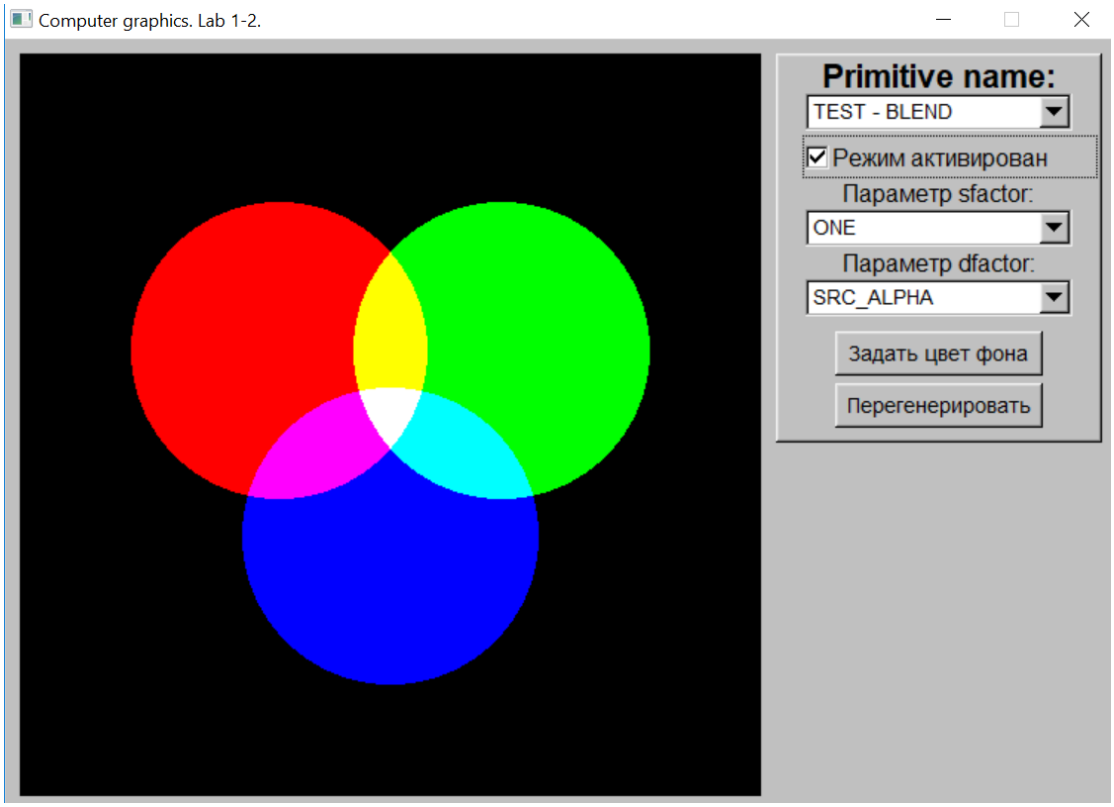
ALPHA:


Computer graphics. Lab 1-2.
Primitive name:
TEST - ALPHA
Режим активирован
Значение параметра:
62
Параметр функции
LESS
Цвет верхней грани:
Красный
Прозрачность верхней грани:
0
Цвет нижней грани:
Красный
Прозрачность нижней грани:
100
Задать цвет фона
Перегенерировать

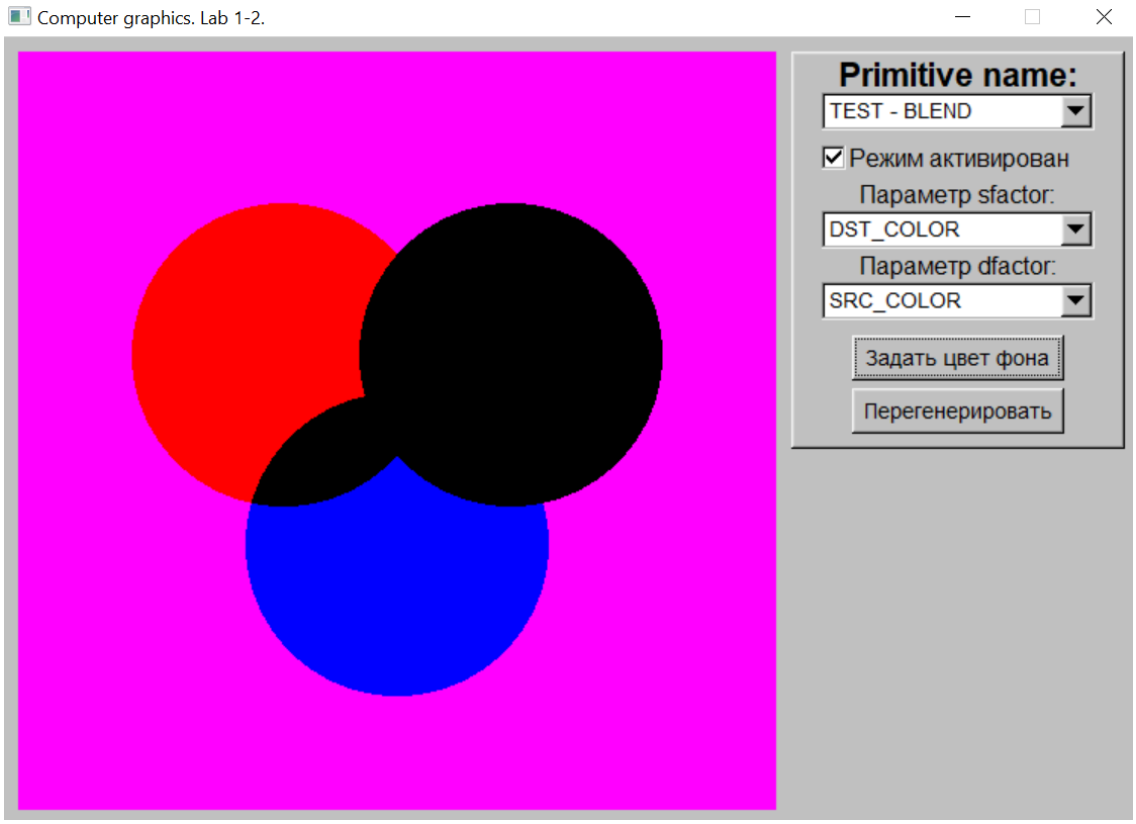
ALPHA:


Computer graphics. Lab 1-2.
Primitive name:
TEST - ALPHA
Режим активирован
Значение параметра:
62
Параметр функции
GREATER
Цвет верхней грани:
Красный
Прозрачность верхней грани:
0
Цвет нижней грани:
Красный
Прозрачность нижней грани:
100
Задать цвет фона
Перегенерировать

BLEND:


Computer graphics. Lab 1-2.
Primitive name:
TEST - BLEND
Режим активирован
Параметр sfactor:
ONE
Параметр dfactor:
SRC_ALPHA
Задать цвет фона
Перегенерировать

BLEND:


Computer graphics. Lab 1-2.
Primitive name:
TEST - BLEND
Режим активирован
Параметр sfactor:
DST_COLOR
Параметр dfactor:
SRC_COLOR
Задать цвет фона
Перегенерировать

Вывод.

В процессе выполнения лабораторной работы была разработана программа, реализующая представление режимов смешивания цветов и отсечения для графических примитивов OpenGL, разработанных в лабораторной работе № 1. Программа работает корректно. При выполнении работы были приобретены навыки работы с графической библиотекой OpenGL.