**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №1**

**по дисциплине «Компьютерная графика»**

Тема: Примитивы OpenGL

|  |  |  |
| --- | --- | --- |
| Студент гр. 8383 |  | Киреев К.А. |
| Преподаватель |  | Герасимова Т.В. |

Санкт-Петербург

2021

**Цель работы.**

Изучить основные примитивы библиотеки OpenGL, разработать программу, реализующую представление определенного набора примитивов из имеющихся в библиотеке OpenGL (GL\_POINT, GL\_LINES, GL\_LINE\_STRIP, GL\_LINE\_LOOP, GL\_TRIANGLES, GL\_TRIANGLE\_STRIP, GL\_TRIANGLE\_FAN, GL\_QUADS, GL\_QUAD\_STRIP, GL\_POLYGON).

**Выполнение работы.**

Работа была выполнена в среде разработке Qt Creator.

**Подключение библиотеки OpenGL в Qt Creator.**

Для подключения библиотеки OpenGL в среде разработки Qt Creator необходимо в файле Lab1.pro прописать следующий код:

QT += core gui opengl

LIBS += -lglu32 -lopengl32

**Реализация примитивов.**

Были реализованы следующие примитивы:

**GL\_POINTS –** каждая вершина рассматривается как отдельная точка, параметры которой не зависят от параметров остальных заданных точек. Рисуется 10 точек.

Листинг программы:

glPointSize(6); // данная функция задает размер точки на экране

glBegin(GL\_POINTS); // указываем примитив, который будем создавать

// 1

glColor3f(1, 0, 0); // указываем цвет вершины

glVertex3f(80, 50, 0); // указываем координаты вершины

// 2

glColor3f(1, 0, 0);

glVertex3f(65, 175, 0);

// 3

glColor3f(0, 0, 1);

glVertex3f(160, 250, 0);

// 4

glColor3f(0, 0, 1);

glVertex3f(240, 270, 0);

// 5

glColor3f(0, 1, 1);

glVertex3f(355, 230, 0);

// 6

glColor3f(0, 1, 1);

glVertex3f(370, 50, 0);

// 7

glColor3f(1, 1, 0);

glVertex3f(260, 25, 0);

// 8

glColor3f(1, 1, 0);

glVertex3f(240, 150, 0);

// 9

glColor3f(0, 1, 0);

glVertex3f(180, 95, 0);

// 10

glColor3f(0, 1, 0);

glVertex3f(130, 15, 0);

glEnd();

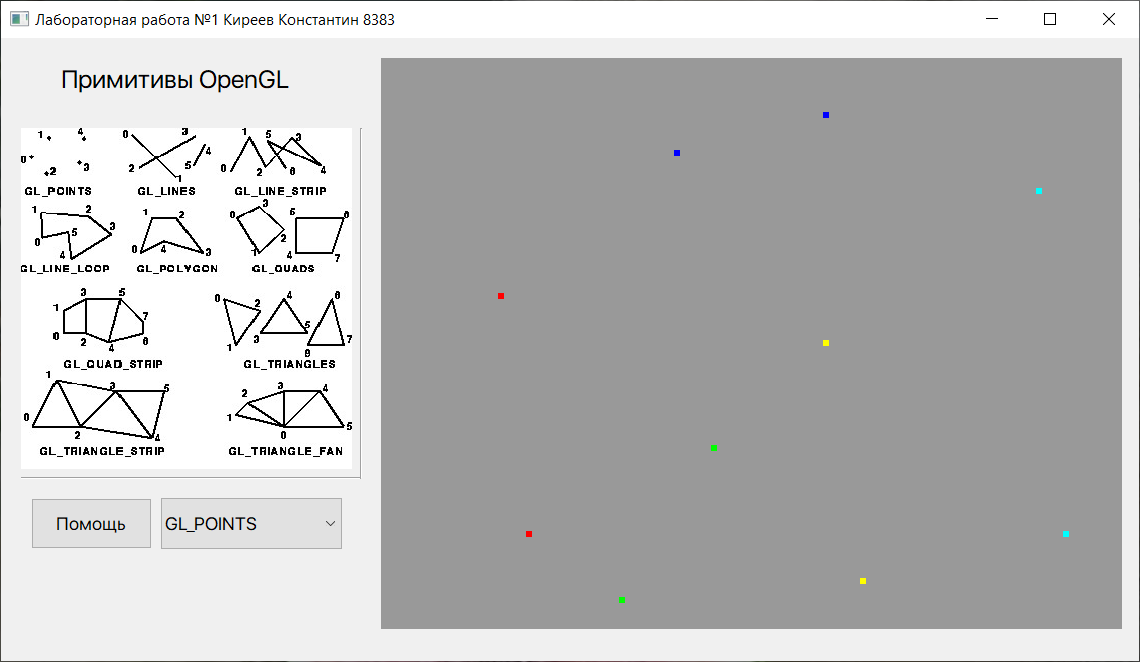


Рис. 1 – Примитив GL\_POINTS

**GL\_LINES** – рисование линий.Вершины без изменений. Рисуется 5 линий.

Листинг программы:

glLineWidth(4);

glBegin(GL\_LINES); // указываем примитив, который будем создавать

*//* *1*

glColor3f(1, 0, 0); // указываем цвет вершины

glVertex3f(80, 50, 0); // указываем координаты вершины

*//* *2*

glColor3f(1, 0, 0);

glVertex3f(65, 175, 0);

*//* *3*

glColor3f(0, 0, 1);

glVertex3f(160, 250, 0);

*//* *4*

glColor3f(0, 0, 1);

glVertex3f(240, 270, 0);

*//* *5*

glColor3f(0, 1, 1);

glVertex3f(355, 230, 0);

*//* *6*

glColor3f(0, 1, 1);

glVertex3f(370, 50, 0);

*//* *7*

glColor3f(1, 1, 0);

glVertex3f(260, 25, 0);

*//* *8*

glColor3f(1, 1, 0);

glVertex3f(240, 150, 0);

*//* *9*

glColor3f(0, 1, 0);

glVertex3f(180, 95, 0);

*//* *10*

glColor3f(0, 1, 0);

glVertex3f(130, 15, 0);

glEnd();

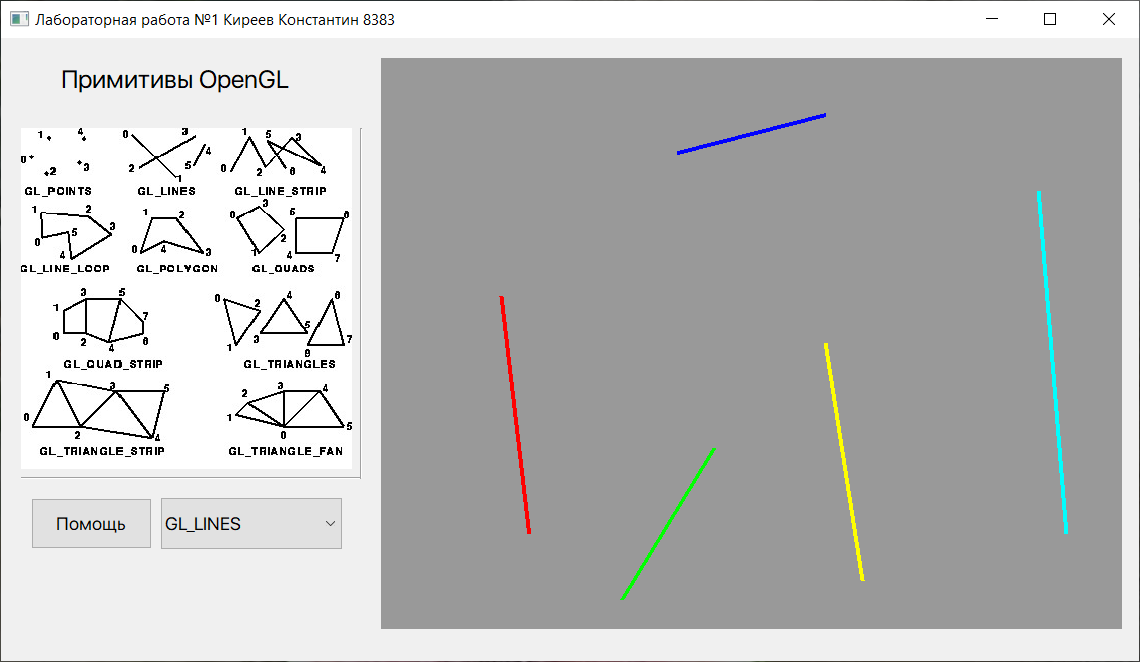


Рис. 2 – Примитив GL\_LINES

**GL\_LINE\_STRIP** – рисование последовательности из нескольких связанных отрезков. Вершины без изменений. Рисуется 9 отрезков.

Листинг программы:

glLineWidth(4);

glBegin(GL\_LINE\_STRIP); // указываем примитив, который будем создавать

*//* *1*

glColor3f(1, 0, 0); // указываем цвет вершины

glVertex3f(80, 50, 0); // указываем координаты вершины

*//* *2*

glColor3f(1, 0, 0);

glVertex3f(65, 175, 0);

*//* *3*

glColor3f(0, 0, 1);

glVertex3f(160, 250, 0);

*//* *4*

glColor3f(0, 0, 1);

glVertex3f(240, 270, 0);

*//* *5*

glColor3f(0, 1, 1);

glVertex3f(355, 230, 0);

*//* *6*

glColor3f(0, 1, 1);

glVertex3f(370, 50, 0);

*//* *7*

glColor3f(1, 1, 0);

glVertex3f(260, 25, 0);

*//* *8*

glColor3f(1, 1, 0);

glVertex3f(240, 150, 0);

*//* *9*

glColor3f(0, 1, 0);

glVertex3f(180, 95, 0);

*//* *10*

glColor3f(0, 1, 0);

glVertex3f(130, 15, 0);

glEnd();

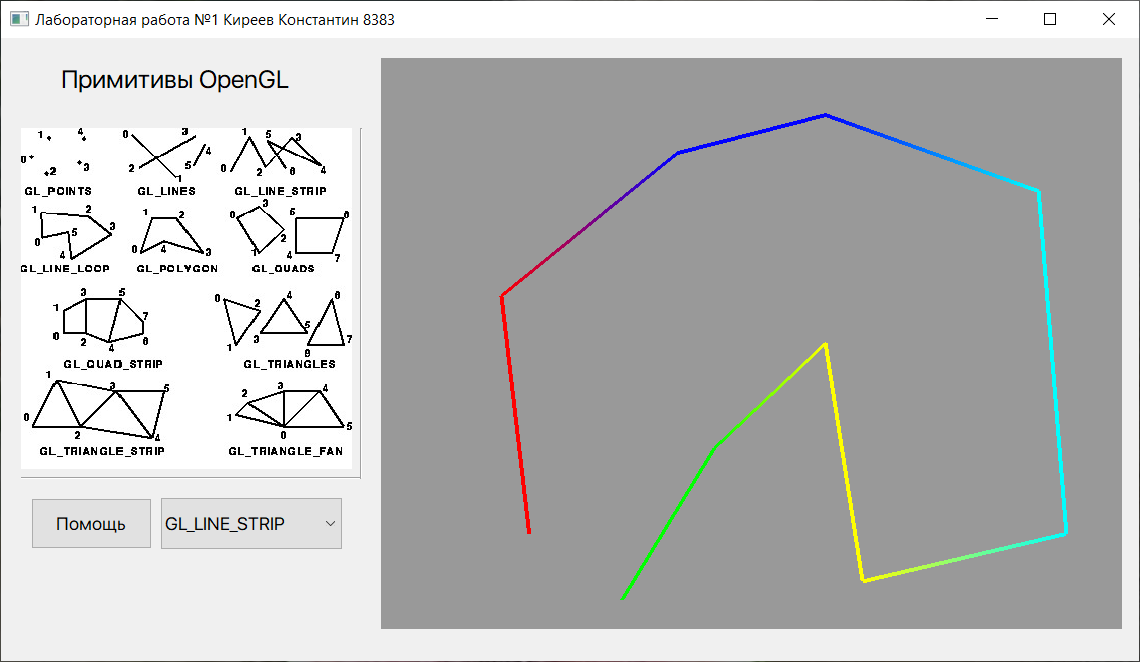


Рис. 3 – Примитив GL\_LINE\_STRIP

**GL\_LINE\_LOOP** – рисование замкнутой кривой линии. Вершины без изменений. Всего рисуется 10 отрезков.

Листинг программы:

glLineWidth(4);

glBegin(GL\_LINE\_LOOP); // указываем примитив, который будем создавать

*//* *1*

glColor3f(1, 0, 0); // указываем цвет вершины

glVertex3f(80, 50, 0); // указываем координаты вершины

*//* *2*

glColor3f(1, 0, 0);

glVertex3f(65, 175, 0);

*//* *3*

glColor3f(0, 0, 1);

glVertex3f(160, 250, 0);

*//* *4*

glColor3f(0, 0, 1);

glVertex3f(240, 270, 0);

*//* *5*

glColor3f(0, 1, 1);

glVertex3f(355, 230, 0);

*//* *6*

glColor3f(0, 1, 1);

glVertex3f(370, 50, 0);

*//* *7*

glColor3f(1, 1, 0);

glVertex3f(260, 25, 0);

*//* *8*

glColor3f(1, 1, 0);

glVertex3f(240, 150, 0);

*//* *9*

glColor3f(0, 1, 0);

glVertex3f(180, 95, 0);

*//* *10*

glColor3f(0, 1, 0);

glVertex3f(130, 15, 0);

glEnd();

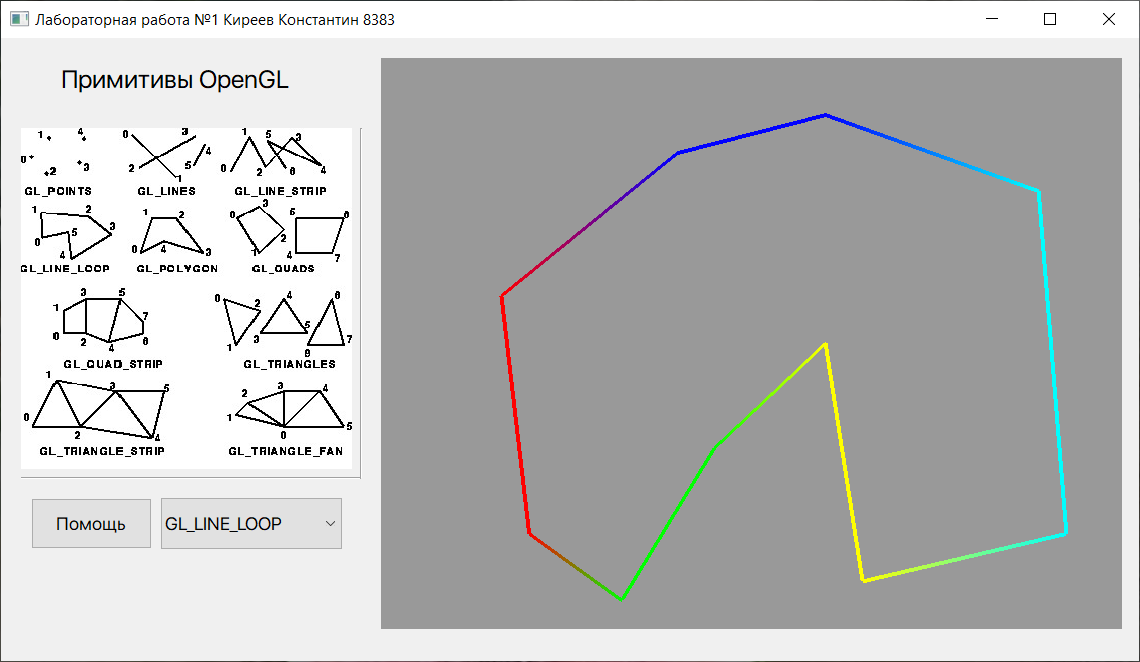


Рис. 4 – Примитив GL\_LINE\_LOOP

**GL\_TRIANGLES –** рисование треугольников. Вершины без ихменений. каждая тройка вершин рассматривается как независимый треугольник. Всего рисуется 3 треугольника, 10 вершина игнорируется.

Листинг программы:

glBegin(GL\_TRIANGLES); // указываем примитив, который будем создавать

*//* *1*

glColor3f(1, 0, 0); // указываем цвет вершины

glVertex3f(80, 50, 0); // указываем координаты вершины

*//* *2*

glColor3f(1, 0, 0);

glVertex3f(65, 175, 0);

*//* *3*

glColor3f(0, 0, 1);

glVertex3f(160, 250, 0);

*//* *4*

glColor3f(0, 0, 1);

glVertex3f(240, 270, 0);

*//* *5*

glColor3f(0, 1, 1);

glVertex3f(355, 230, 0);

*//* *6*

glColor3f(0, 1, 1);

glVertex3f(370, 50, 0);

*//* *7*

glColor3f(1, 1, 0);

glVertex3f(260, 25, 0);

*//* *8*

glColor3f(1, 1, 0);

glVertex3f(240, 150, 0);

*//* *9*

glColor3f(0, 1, 0);

glVertex3f(180, 95, 0);

*//* *10*

glColor3f(0, 1, 0);

glVertex3f(130, 15, 0);

glEnd();

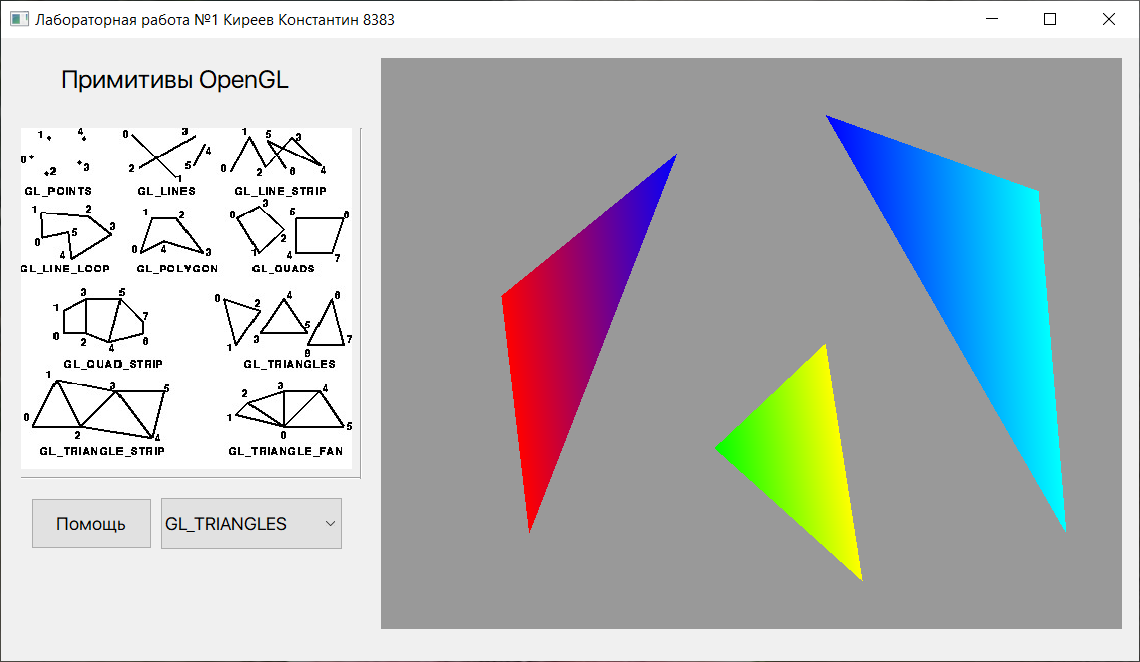


Рис. 5 – Примитив GL\_TRIANGLES

**GL\_TRIANGLE\_STRIP –** рисование группы связанных треугольников, имеющих общую грань. Вершины без изменений. Всего рисуется 8 треугольников.

Листинг программы:

glBegin(GL\_TRIANGLE\_STRIP); // указываем примитив, который будем создавать

*//* *1*

glColor3f(1, 0, 0); // указываем цвет вершины

glVertex3f(80, 50, 0); // указываем координаты вершины

*//* *2*

glColor3f(1, 0, 0);

glVertex3f(65, 175, 0);

*//* *3*

glColor3f(0, 0, 1);

glVertex3f(160, 250, 0);

*//* *4*

glColor3f(0, 0, 1);

glVertex3f(240, 270, 0);

*//* *5*

glColor3f(0, 1, 1);

glVertex3f(355, 230, 0);

*//* *6*

glColor3f(0, 1, 1);

glVertex3f(370, 50, 0);

*//* *7*

glColor3f(1, 1, 0);

glVertex3f(260, 25, 0);

*//* *8*

glColor3f(1, 1, 0);

glVertex3f(240, 150, 0);

*//* *9*

glColor3f(0, 1, 0);

glVertex3f(180, 95, 0);

*//* *10*

glColor3f(0, 1, 0);

glVertex3f(130, 15, 0);

glEnd();

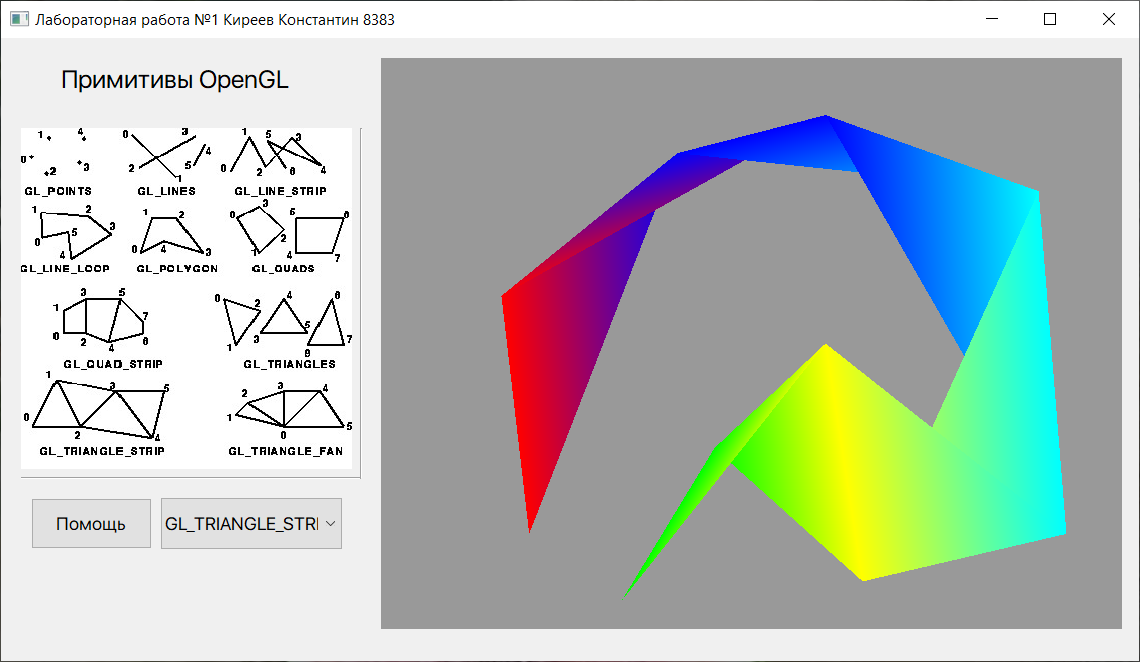


Рис. 6 – Примитив GL\_TRIANGLE\_STRIP

**GL\_TRIANGLE\_FAN –** рисование группы связанных треугольников, имеющих общие грани и одну общую вершину. Вершины без изменений.

Листинг программы:

glBegin(GL\_TRIANGLE\_FAN); // указываем примитив, который будем создавать

*//* *1*

glColor3f(1, 0, 0); // указываем цвет вершины

glVertex3f(80, 50, 0); // указываем координаты вершины

*//* *2*

glColor3f(1, 0, 0);

glVertex3f(65, 175, 0);

*//* *3*

glColor3f(0, 0, 1);

glVertex3f(160, 250, 0);

*//* *4*

glColor3f(0, 0, 1);

glVertex3f(240, 270, 0);

*//* *5*

glColor3f(0, 1, 1);

glVertex3f(355, 230, 0);

*//* *6*

glColor3f(0, 1, 1);

glVertex3f(370, 50, 0);

*//* *7*

glColor3f(1, 1, 0);

glVertex3f(260, 25, 0);

*//* *8*

glColor3f(1, 1, 0);

glVertex3f(240, 150, 0);

*//* *9*

glColor3f(0, 1, 0);

glVertex3f(180, 95, 0);

*//* *10*

glColor3f(0, 1, 0);

glVertex3f(130, 15, 0);

glEnd();

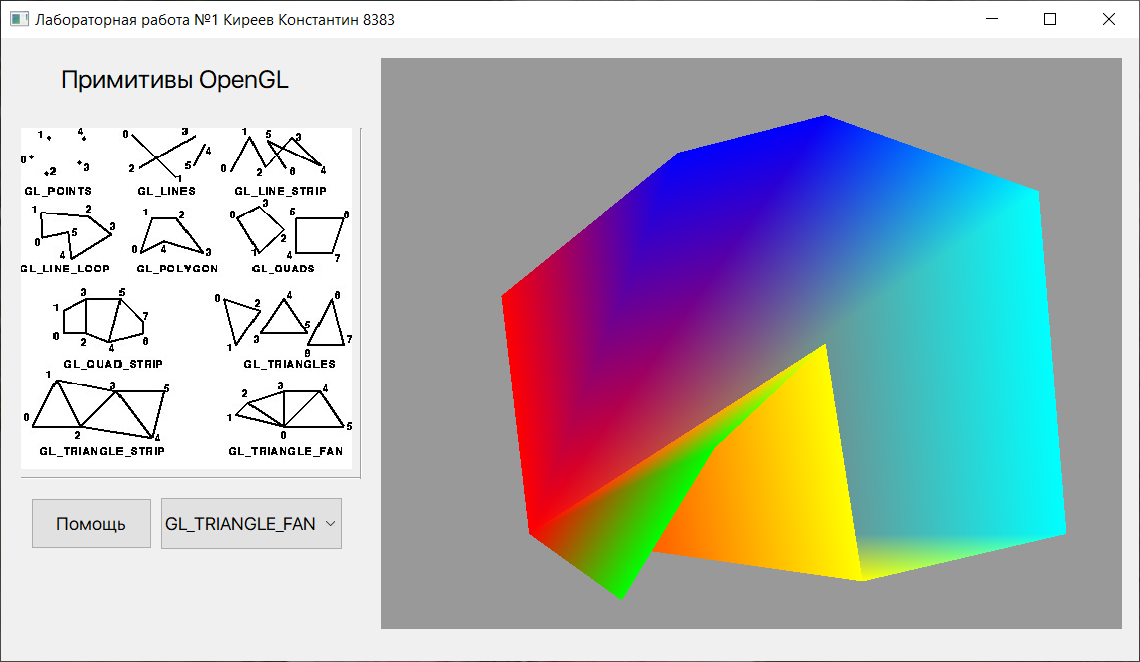


Рис. 7 – Примитив GL\_TRIANGLE\_FAN

**GL\_QUADS –** рисование четырехугольников. Вершины без изменений. Всего рисуется 2 четырехугольника, 9 и 10 вершины игнорируются.

Листинг программы:

glBegin(GL\_QUADS); // указываем примитив, который будем создавать

*//* *1*

glColor3f(1, 0, 0); // указываем цвет вершины

glVertex3f(80, 50, 0); // указываем координаты вершины

*//* *2*

glColor3f(1, 0, 0);

glVertex3f(65, 175, 0);

*//* *3*

glColor3f(0, 0, 1);

glVertex3f(160, 250, 0);

*//* *4*

glColor3f(0, 0, 1);

glVertex3f(240, 270, 0);

*//* *5*

glColor3f(0, 1, 1);

glVertex3f(355, 230, 0);

*//* *6*

glColor3f(0, 1, 1);

glVertex3f(370, 50, 0);

*//* *7*

glColor3f(1, 1, 0);

glVertex3f(260, 25, 0);

*//* *8*

glColor3f(1, 1, 0);

glVertex3f(240, 150, 0);

*//* *9*

glColor3f(0, 1, 0);

glVertex3f(180, 95, 0);

*//* *10*

glColor3f(0, 1, 0);

glVertex3f(130, 15, 0);

glEnd();

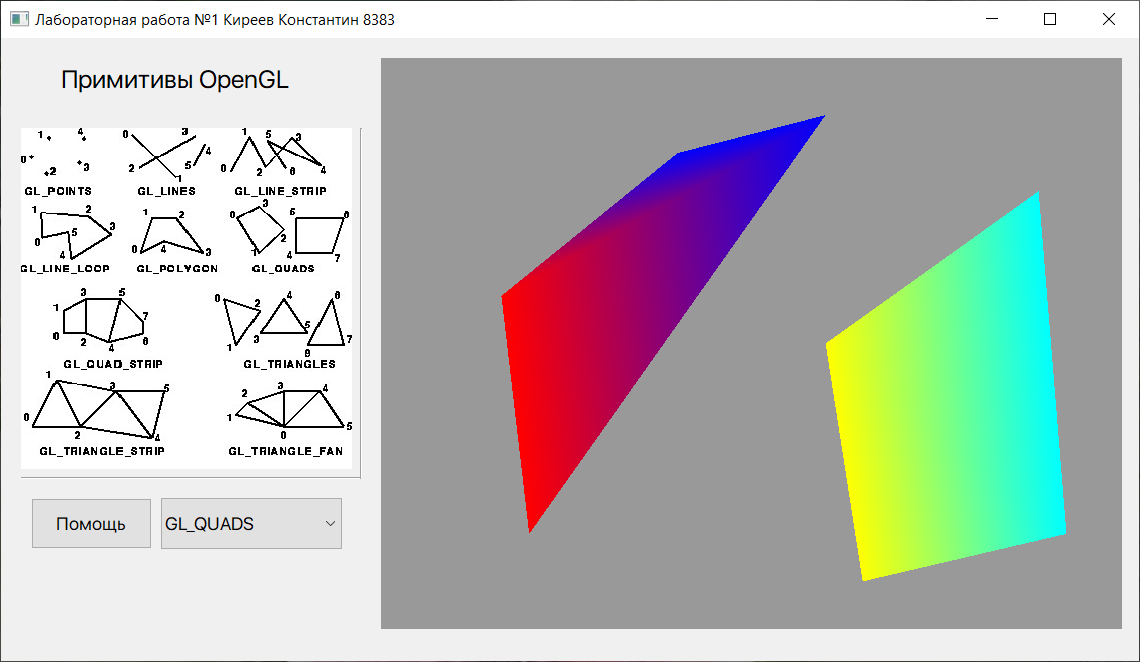


Рис. 8 – Примитив GL\_QUADS

**GL\_QUADS\_STRIP –** рисование четырехугольников, имеющих общую грань. Вершины без изменений.

Листинг программы:

glBegin(GL\_QUADS\_STRIP); // указываем примитив, который будем создавать

*//* *1*

glColor3f(1, 0, 0); // указываем цвет вершины

glVertex3f(80, 50, 0); // указываем координаты вершины

*//* *2*

glColor3f(1, 0, 0);

glVertex3f(65, 175, 0);

*//* *3*

glColor3f(0, 0, 1);

glVertex3f(160, 250, 0);

*//* *4*

glColor3f(0, 0, 1);

glVertex3f(240, 270, 0);

*//* *5*

glColor3f(0, 1, 1);

glVertex3f(355, 230, 0);

*//* *6*

glColor3f(0, 1, 1);

glVertex3f(370, 50, 0);

*//* *7*

glColor3f(1, 1, 0);

glVertex3f(260, 25, 0);

*//* *8*

glColor3f(1, 1, 0);

glVertex3f(240, 150, 0);

*//* *9*

glColor3f(0, 1, 0);

glVertex3f(180, 95, 0);

*//* *10*

glColor3f(0, 1, 0);

glVertex3f(130, 15, 0);

glEnd();

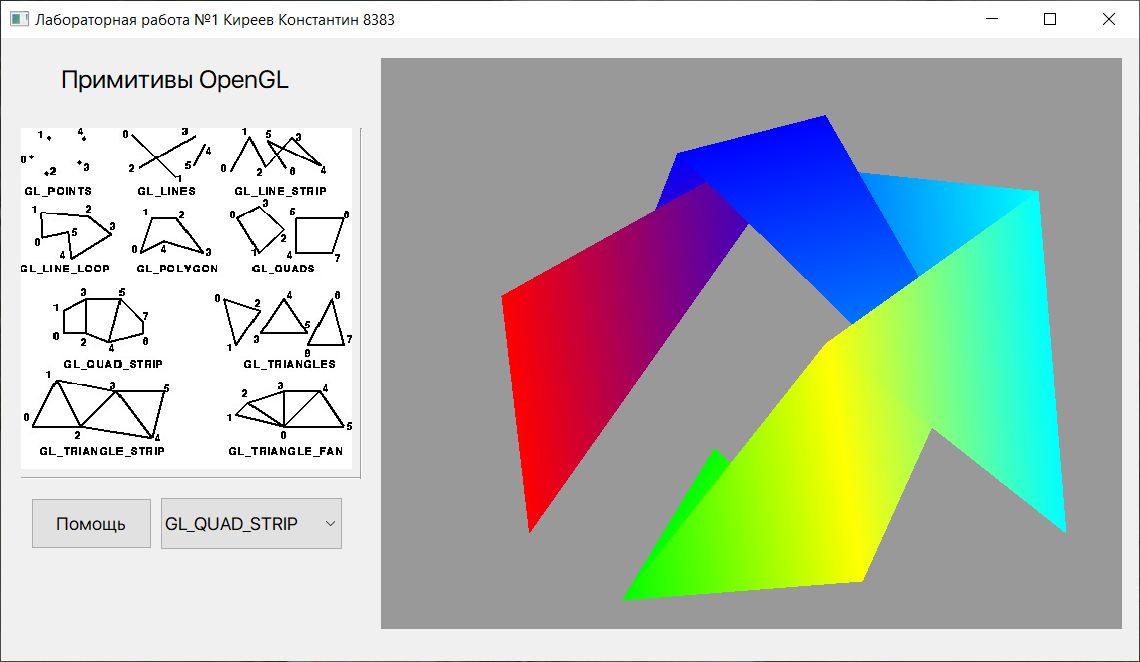


Рис. 9 – Примитив GL\_QUADS\_STRIP

**GL\_POLYGON –** рисование многоугольник. Вершины без изменений.

Листинг программы:

glBegin(GL\_POLYGON); // указываем примитив, который будем создавать

*//* *1*

glColor3f(1, 0, 0); // указываем цвет вершины

glVertex3f(80, 50, 0); // указываем координаты вершины

*//* *2*

glColor3f(1, 0, 0);

glVertex3f(65, 175, 0);

*//* *3*

glColor3f(0, 0, 1);

glVertex3f(160, 250, 0);

*//* *4*

glColor3f(0, 0, 1);

glVertex3f(240, 270, 0);

*//* *5*

glColor3f(0, 1, 1);

glVertex3f(355, 230, 0);

*//* *6*

glColor3f(0, 1, 1);

glVertex3f(370, 50, 0);

*//* *7*

glColor3f(1, 1, 0);

glVertex3f(260, 25, 0);

*//* *8*

glColor3f(1, 1, 0);

glVertex3f(240, 150, 0);

*//* *9*

glColor3f(0, 1, 0);

glVertex3f(180, 95, 0);

*//* *10*

glColor3f(0, 1, 0);

glVertex3f(130, 15, 0);

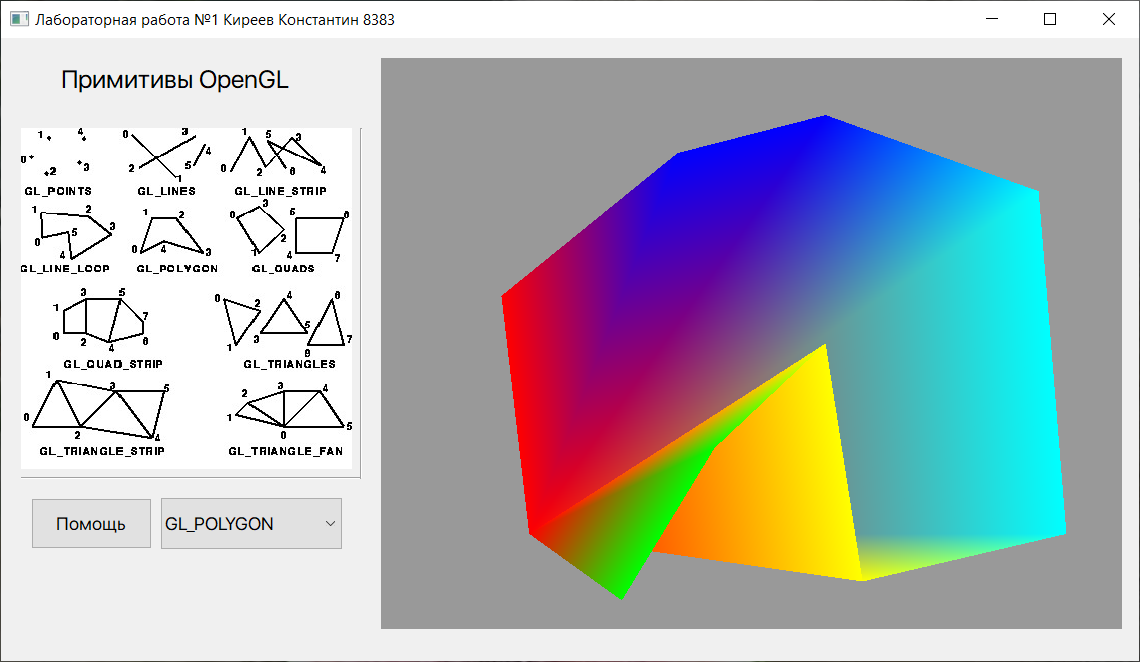
glEnd();

Рис. 10 – Примитив GL\_POLYGON

**Выводы.**

В результате выполнения лабораторной работы была разработана программа, создающая графические примитивы OpenGL. Программа работает корректно. При выполнении работы были приобретены навыки работы с графической библиотекой OpenGL.