|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Кафедра |  | O7 |  | Информационные системы и программная инженерия |
|  |  | шифр |  | наименование кафедры, по которой выполняется работа |
| Дисциплина |  | Компьютерная геометрия и графика | | |
|  |  | наименование дисциплины | | |

|  |  |  |
| --- | --- | --- |
| ЛАБОРАТОРНАЯ РАБОТА | 2 |  |
|  | номер (при наличии) |  |
| Примитивы OpenGL. | | |
| Основные приемы построения двумерных объектов. | | |
| Основы шейдеров. | | |

при наличии указать тему лабораторной работы и (или) номер варианта

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ОБУЧАЮЩИЙСЯ** | | | | | | |
| группы | | | |  | | И913Б |
|  |  | Кульга Н.И | | | | |
| подпись |  | фамилия и инициалы | | | | |
|  | | | | |
| дата сдачи | | | | |
| **ПРОВЕРИЛ** | | | | | | |
|  | | | | | | | |
| ученая степень, ученое звание, должность | | | | | | | |
|  |  | Мажайцев Е.А | | | | | |
| подпись |  | фамилия и инициалы | | | | | |
| Оценка / балльная оценка | | |  | | | |
|  | | | | |
| дата проверки | | | | |

Задание:

Задание 1. Построить точки, расположенные в вершинах правильного n-угольника. Установить режим сглаживания для точек. Экспериментально определить максимальный размер точки, при котором возможно сглаживание.

Задание 2. Используя примитив для вывода линий нарисовать правильный n-угольник. Изменить тип и ширину линий.

Задание 3. Используя примитив для вывода ломаной линии нарисовать фигуру, изображенную на рис.1.

Задание 4. Используя примитив для вывода замкнутой ломаной нарисовать фигуру, изображенную на рис.2

Задание 5. Построить фигуру, изображенную на рис.2, разбив ее на треугольники (каждый треугольник окрашен случайным цветом). Выполните три варианта построений с использованием примитивов:

А) треугольник;

Б) лента треугольников;

В) веер треугольников.

Чем отличаются результаты при изменении способа тонирования?

Задание 6. Используя примитив для вывода веера треугольников построить правильный n-угольник.

Задание 7. Построить невыпуклый многоугольник, изображенный на рис.3, представив его в виде совокупности отдельных треугольников, назначив каждому треугольнику свой цвет. Посмотреть результат работы программы для различных способов тонирования.

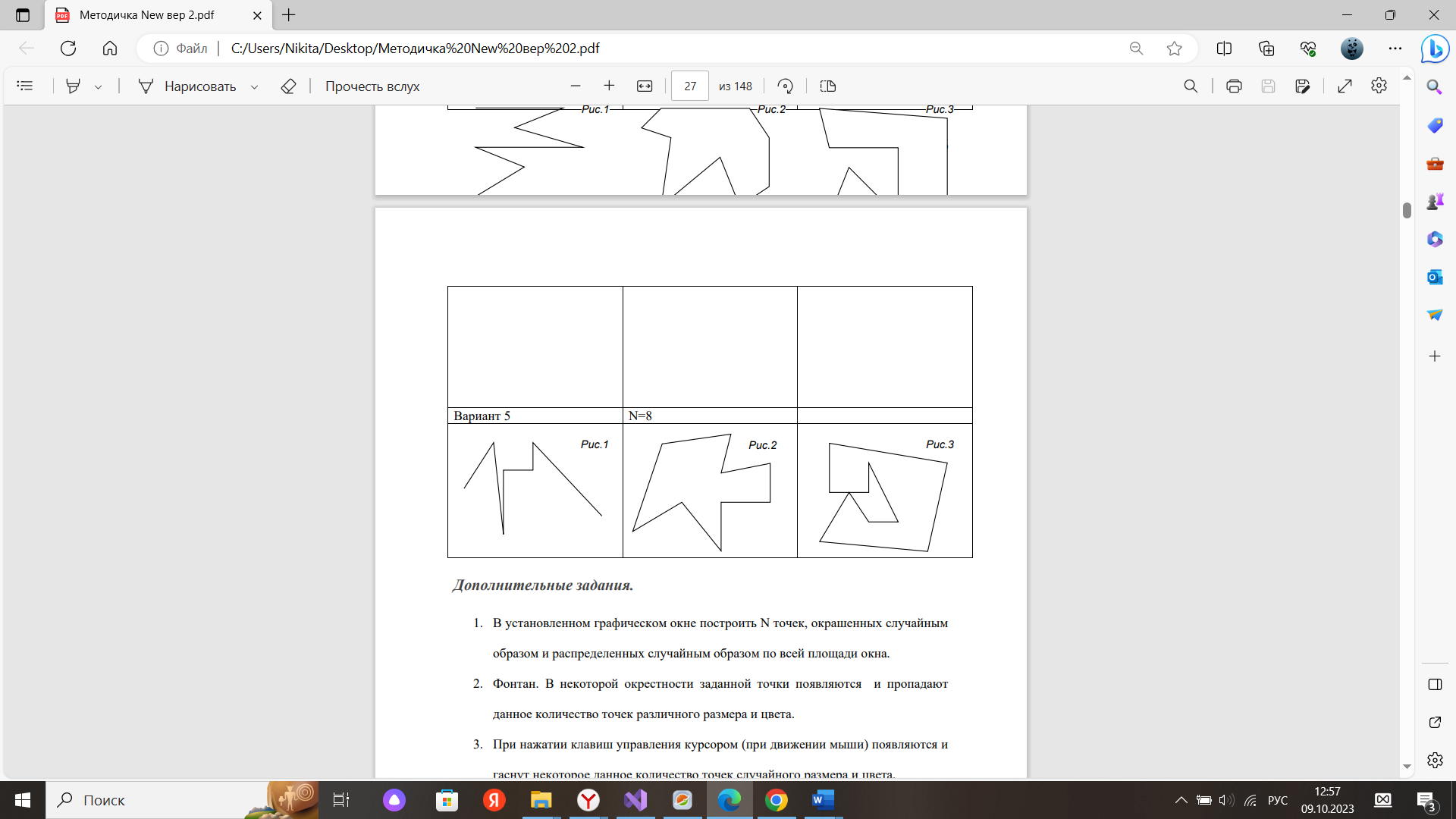
Задание 8. Изменить программу предыдущей задачи таким образом, чтобы

А) лицевые грани изображались только вершинами;

Б) лицевые грани изображались закрашенными, а обратные – линиями;

В) лицевые и обратные грани изображались линиями (каркасное

изображение).



Результат работы программы:

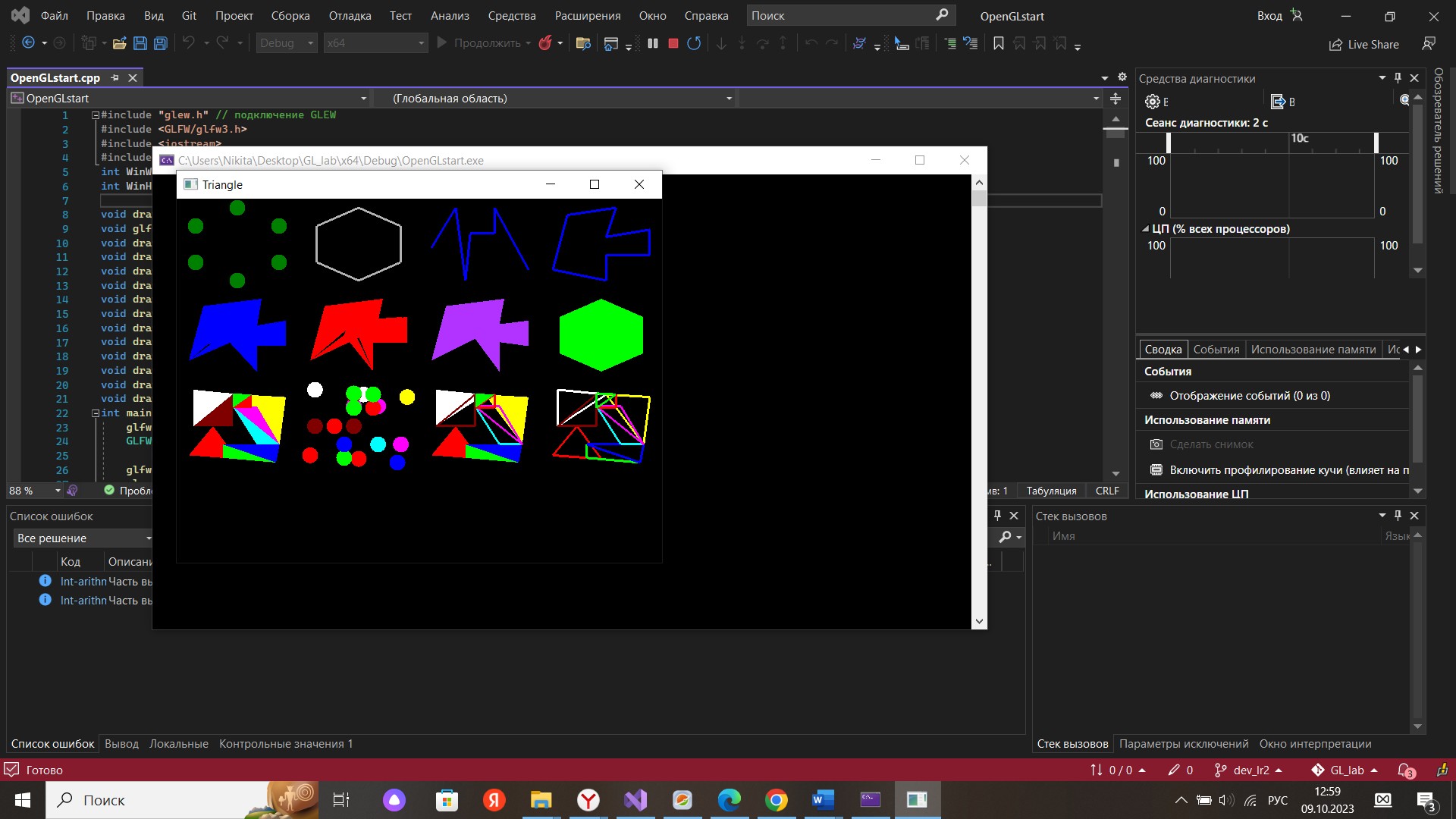


Рисунок 1 -Результат работы.

Код программы:

Main.cpp:

#include "glew.h" // подключение GLEW

#include <GLFW/glfw3.h>

#include <iostream>

#include "ShaderFuncs.h"

int WinWidth = 640;

int WinHeight = 480;

void drawSmth(GLfloat\* colors, GLfloat\* points, GLfloat startx, GLfloat starty, GLfloat size, GLenum type, int n);

void glfw\_window\_size\_callback(GLFWwindow\* window, int width, int height);

void drawHexagonDots(GLfloat startx, GLfloat starty, GLfloat size);

void drawHexagon(GLfloat startx, GLfloat starty, GLfloat size);

void drawLineStrip(GLfloat startx, GLfloat starty, GLfloat size);

void drawLineLoop(GLfloat startx, GLfloat starty, GLfloat size);

void drawTrianglesStrip(GLfloat startx, GLfloat starty, GLfloat size);

void drawTrianglesFan(GLfloat startx, GLfloat starty, GLfloat size);

void drawTriangles(GLfloat startx, GLfloat starty, GLfloat size);

void drawHexagonFan(GLfloat startx, GLfloat starty, GLfloat size);

void drawPolygon(GLfloat startx, GLfloat starty, GLfloat size);

void drawPolygon2(GLfloat startx, GLfloat starty, GLfloat size);

void drawPolygon3(GLfloat startx, GLfloat starty, GLfloat size);

void drawPolygon4(GLfloat startx, GLfloat starty, GLfloat size);

int main() {

glfwInit();

GLFWwindow\* window = glfwCreateWindow(WinWidth, WinHeight, "Triangle",

NULL, NULL);

glfwMakeContextCurrent(window);

glewInit();

glfwSetWindowSizeCallback(window, glfw\_window\_size\_callback);

GLuint vs = LoadShader("VertexShader.glsl", GL\_VERTEX\_SHADER);

GLuint fs = LoadShader("FragmentShader.glsl", GL\_FRAGMENT\_SHADER);

GLuint shader\_programme = glCreateProgram();

glAttachShader(shader\_programme, vs);

glAttachShader(shader\_programme, fs);

glBindAttribLocation(shader\_programme, 0, "vertex\_position");

glBindAttribLocation(shader\_programme, 1, "vertex\_colour");

glLinkProgram(shader\_programme);

while (!glfwWindowShouldClose(window)) {

glfwPollEvents();

if (GLFW\_PRESS == glfwGetKey(window, GLFW\_KEY\_ESCAPE))

{

glfwSetWindowShouldClose(window, 1);

}

glViewport(0, 0, WinWidth, WinHeight);

glClear(GL\_COLOR\_BUFFER\_BIT);

glUseProgram(shader\_programme);

drawHexagonDots(-0.75f, 0.75f, 0.2f);

drawHexagon(-0.25f, 0.75f, 0.2f);

drawLineStrip(0.25f, 0.75f, 0.2f);

drawLineLoop(0.75f, 0.75f, 0.2f);

drawTriangles(-0.75f, 0.25f, 0.2f);

drawTrianglesStrip(-0.25f, 0.25f, 0.2f);

drawTrianglesFan(0.25f, 0.25f, 0.2f);

drawHexagonFan(0.75f, 0.25f, 0.2f);

drawPolygon(-0.75f, -0.25f, 0.2f);

drawPolygon2(-0.25f, -0.25f, 0.2f);

drawPolygon3(0.25f, -0.25f, 0.2f);

drawPolygon4(0.75f, -0.25f, 0.2f);

glfwSwapBuffers(window);

}

glfwTerminate();

return 0;

}

void glfw\_window\_size\_callback(GLFWwindow\* window, int width, int

height) {

WinWidth = width;

WinHeight = height;

}

void drawSmth(GLfloat\* colors, GLfloat\* points, GLfloat startx, GLfloat starty, GLfloat size, GLenum type, int n) {

for (int i = 0; i < n \* 3; i++)

points[i] \*= size;

for (int i = 0; i < n \* 3; i += 3)

points[i] += startx;

for (int i = 1; i < n \* 3; i += 3)

points[i] += starty;

GLuint coords\_vbo = 0;

glGenBuffers(1, &coords\_vbo);

glBindBuffer(GL\_ARRAY\_BUFFER, coords\_vbo);

glBufferData(GL\_ARRAY\_BUFFER, n \* 3 \* sizeof(float), points,

GL\_STATIC\_DRAW);

GLuint colors\_vbo = 0;

glGenBuffers(1, &colors\_vbo);

glBindBuffer(GL\_ARRAY\_BUFFER, colors\_vbo);

glBufferData(GL\_ARRAY\_BUFFER, n \* 3 \* sizeof(float), colors,

GL\_STATIC\_DRAW);

GLuint vao = 0;

glGenVertexArrays(1, &vao);

glBindVertexArray(vao);

glBindBuffer(GL\_ARRAY\_BUFFER, coords\_vbo);

glVertexAttribPointer(0, 3, GL\_FLOAT, GL\_FALSE, 0, NULL);

glBindBuffer(GL\_ARRAY\_BUFFER, colors\_vbo);

glVertexAttribPointer(1, 3, GL\_FLOAT, GL\_FALSE, 0, NULL);

glEnableVertexAttribArray(0);

glEnableVertexAttribArray(1);

glBindVertexArray(vao);

glDrawArrays(type, 0, n);

glBindVertexArray(vao);

glDeleteVertexArrays(1, &vao);

glBindBuffer(GL\_ARRAY\_BUFFER, coords\_vbo);

glDeleteBuffers(1, &coords\_vbo);

glBindBuffer(GL\_ARRAY\_BUFFER, colors\_vbo);

glDeleteBuffers(1, &colors\_vbo);

}

void drawHexagonDots(GLfloat startx, GLfloat starty, GLfloat size) {

GLfloat colors[] = {

0.0f, 0.5f, 0.0f,

0.0f, 0.5f, 0.0f,

0.0f, 0.5f, 0.0f,

0.0f, 0.5f, 0.0f,

0.0f, 0.5f, 0.0f,

0.0f, 0.5f, 0.0f,

};

GLfloat points[] = {

0.0f, 1.0f, 0.0f,

-0.86f, 0.5f, 0.0f,

0.86f, 0.5f, 0.0f,

0.86f, -0.5f, 0.0f,

-0.86f, -0.5f, 0.0f,

0.0f, -1.0f, 0.0f

};

glEnable(GL\_PROGRAM\_POINT\_SIZE);

glPointSize(20);

glEnable(GL\_POINT\_SMOOTH);

drawSmth(colors, points, startx, starty, size, GL\_POINTS, 6);

}

void drawHexagon(GLfloat startx, GLfloat starty, GLfloat size) {

GLfloat colors[] = {

0.7f, 0.7f, 0.7f,

0.7f, 0.7f, 0.7f,

0.7f, 0.7f, 0.7f,

0.7f, 0.7f, 0.7f,

0.7f, 0.7f, 0.7f,

0.7f, 0.7f, 0.7f

};

GLfloat points[] = {

0.0f, 1.0f, 0.0f,

-0.86f, 0.5f, 0.0f,

-0.86f, -0.5f, 0.0f,

0.0f, -1.0f, 0.0f,

0.86f, -0.5f, 0.0f,

0.86f, 0.5f, 0.0f,

};

glLineWidth(3);

drawSmth(colors, points, startx, starty, size, GL\_LINE\_LOOP, 6);

}

void drawLineStrip(GLfloat startx, GLfloat starty, GLfloat size) {

GLfloat colors[] = {

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

};

GLfloat points[] = {

-1.0f, -0.1f, 0.0f,

-0.5f, 1.0f, 0.0f,

-0.3f, -1.0f, 0.0f,

-0.2f, 0.3f, 0.0f,

0.3f, 0.3f, 0.0f,

0.3f, 1.0f, 0.0f,

1.0f, -0.7f, 0.0f,

};

glLineWidth(3);

drawSmth(colors, points, startx, starty, size, GL\_LINE\_STRIP, 7);

}

void drawLineLoop(GLfloat startx, GLfloat starty, GLfloat size) {

GLfloat colors[] = {

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

};

GLfloat points[] = {

-1.0f, -0.7f, 0.0f,

-0.7f, 0.8f, 0.0f,

0.3f, 1.0f, 0.0f,

0.1f, 0.2f, 0.0f,

1.0f, 0.4f, 0.0f,

1.0f, -0.3f, 0.0f,

0.1f, -0.3f, 0.0f,

0.1f, -1.0f, 0.0f,

};

glLineWidth(3);

drawSmth(colors, points, startx, starty, size, GL\_LINE\_LOOP, 8);

}

void drawTriangles(GLfloat startx, GLfloat starty, GLfloat size) {

GLfloat colors[] = {

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

0.0f, 0.0f, 1.0f,

};

GLfloat points[] = {

-1.0f, -0.7f, 0.0f,

-0.7f, 0.8f, 0.0f,

0.5f, 1.0f, 0.0f,

-1.0f, -0.7f, 0.0f,

-0.3f, 0.0f, 0.0f,

0.2f, 0.0f, 0.0f,

0.5f, 1.0f, 0.0f,

0.4f, 0.2f, 0.0f,

-0.7f, -0.3f, 0.0f,

0.1f, 0.2f, 0.0f,

1.0f, 0.4f, 0.0f,

1.0f, -0.3f, 0.0f,

1.0f, -0.3f, 0.0f,

0.1f, -0.3f, 0.0f,

-0.5f, 0.55f, 0.0f,

0.4f, 0.3f, 0.0f,

0.4f, -1.0f, 0.0f,

-0.3f, 0.0f, 0.0f,

};

drawSmth(colors, points, startx, starty, size, GL\_TRIANGLES, 18);

}

void drawTrianglesStrip(GLfloat startx, GLfloat starty, GLfloat size) {

GLfloat colors[] = {

1.0f, 0.0f, 0.0f,

1.0f, 0.0f,0.0f,

1.0f, 0.0f, 0.0f,

1.0f, 0.0f, 0.0f,

1.0f, 0.0f, 0.0f,

1.0f, 0.0f,0.0f,

1.0f, 0.0f, 0.0f,

1.0f, 0.0f, 0.0f,

1.0f, 0.0f, 0.0f,

1.0f, 0.0f, 0.0f,

1.0f, 0.0f, 0.0f,

1.0f, 0.0f, 0.0f,

1.0f, 0.0f, 0.0f,

1.0f, 0.0f, 0.0f,

1.0f, 0.0f, 0.0f,

1.0f, 0.0f, 0.0f,

1.0f, 0.0f, 0.0f,

};

GLfloat points[] = {

-1.0f, -0.7f, 0.0f,

-0.7f, 0.8f, 0.0f,

0.5f, 1.0f, 0.0f,

0.3f, 0.4f, 0.0f,

0.3f, -0.2f, 0.0f,

1.0f, 0.5f, 0.0f,

1.0f, -0.2f, 0.0f,

0.3f, 0.4f, 0.0f,

-0.2f, 0.6f, 0.0f,

0.3f, -1.0f, 0.0f,

0.3f, -1.0f, 0.0f,

-0.2f, 0.4f, 0.0f,

-0.3f, 0.0f, 0.0f,

0.9f, 0.5f, 0.0f,

-1.0f, -0.7f, 0.0f,

};

drawSmth(colors, points, startx, starty, size, GL\_TRIANGLE\_STRIP, 15);

}

void drawTrianglesFan(GLfloat startx, GLfloat starty, GLfloat size) {

GLfloat colors[] = {

0.7f, 0.2f, 1.0f,

0.7f, 0.2f, 1.0f,

0.7f, 0.2f, 1.0f,

0.7f, 0.2f, 1.0f,

0.7f, 0.2f, 1.0f,

0.7f, 0.2f, 1.0f,

0.7f, 0.2f, 1.0f,

0.7f, 0.2f, 1.0f,

0.7f, 0.2f, 1.0f,

0.7f, 0.2f, 1.0f,

0.7f, 0.2f, 1.0f,

0.7f, 0.2f, 1.0f,

0.7f, 0.2f, 1.0f,

0.7f, 0.2f, 1.0f,

0.7f, 0.2f, 1.0f,

0.7f, 0.2f, 1.0f,

0.7f, 0.2f, 1.0f

};

GLfloat points[] = {

-0.1f, -0.1f, 0.0f,

-1.0f, -0.7f, 0.0f,

-0.7f, 0.8f, 0.0f,

0.5f, 1.0f, 0.0f,

0.4f, 0.3f, 0.0f,

1.0f, 0.4f, 0.0f,

1.0f, -0.3f, 0.0f,

0.4f, 0.3f, 0.0f,

0.4f, -1.0f, 0.0f,

};

drawSmth(colors, points, startx, starty, size, GL\_TRIANGLE\_FAN, 9);

}

void drawHexagonFan(GLfloat startx, GLfloat starty, GLfloat size) {

GLfloat colors[] = {

0.0f, 1.0f, 0.0f,

0.0f, 1.0f, 0.0f,

0.0f, 1.0f, 0.0f,

0.0f, 1.0f, 0.0f,

0.0f, 1.0f, 0.0f,

0.0f, 1.0f, 0.0f,

0.0f, 1.0f, 0.0f,

0.0f, 1.0f, 0.0f,

};

GLfloat points[] = {

0.0f,0.0f,0.0f,

0.0f, 1.0f, 0.0f,

-0.86f, 0.5f, 0.0f,

-0.86f, -0.5f, 0.0f,

0.0f, -1.0f, 0.0f,

0.86f, -0.5f, 0.0f,

0.86f, 0.5f, 0.0f,

0.0f, 1.0f, 0.0f,

};

drawSmth(colors, points, startx, starty, size, GL\_TRIANGLE\_FAN, 8);

}

void drawPolygon(GLfloat startx, GLfloat starty, GLfloat size) {

GLfloat colors[] = {

1.0f,0.0f,0.0f,

1.0f,0.0f,0.0f,

1.0f,0.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,0.0f,1.0f,

0.0f,0.0f,1.0f,

0.0f,0.0f,1.0f,

1.0f,1.0f,0.0f,

1.0f,1.0f,0.0f,

1.0f,1.0f,0.0f,

0.0f,1.0f,1.0f,

0.0f,1.0f,1.0f,

0.0f,1.0f,1.0f,

1.0f,0.0f,1.0f,

1.0f,0.0f,1.0f,

1.0f,0.0f,1.0f,

1.0f,1.0f,1.0f,

1.0f,1.0f,1.0f,

1.0f,1.0f,1.0f,

0.5f,0.0f,0.0f,

0.5f,0.0f,0.0f,

0.5f,0.0f,0.0f,

1.0f,0.0f,0.0f,

1.0f,0.0f,0.0f,

1.0f,0.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,0.0f,1.0f,

0.0f,0.0f,1.0f,

0.0f,0.0f,1.0f,

};

GLfloat points[] = {

-1.0f,-0.8f,0.0f,

0.0f,-0.9f,0.0f,

-0.5f,0.0f,0.0f,

-0.3f,-0.5f,0.0f,

-0.3f,-0.87f,0.0f,

0.8f,-1.0f,0.0f,

0.8f,-1.0f,0.0f,

0.87f,-0.5f,0.0f,

-0.3f,-0.5f,0.0f,

0.87f,-0.5f,0.0f,

1.0f,0.8f,0.0f,

0.1f,0.87f,0.0f,

0.87f,-0.5f,0.0f,

0.4f,-0.5f,0.0f,

-0.1f,0.55f,0.0f,

0.87f,-0.5f,0.0f,

-0.1f,0.55f,0.0f,

0.4f,0.55f,0.0f,

0.1f,0.87f,0.0f,

-0.9f,1.0f,0.0f,

-0.9f,0.0f,0.0f,

-0.9f,0.0f,0.0f,

-0.1f,0.87f,0.0f,

-0.1f,0.0f,0.0f,

0.3f,0.87f,0.0f,

0.3f,0.5f,0.0f,

-0.1f,0.5f,0.0f,

0.3f,0.87f,0.0f,

-0.1f,0.9f,0.0f,

-0.1f,0.5f,0.0f,

};

drawSmth(colors, points, startx, starty, size, GL\_TRIANGLES, 30);

}

void drawPolygon2(GLfloat startx, GLfloat starty, GLfloat size) {

GLfloat colors[] = {

1.0f,0.0f,0.0f,

1.0f,0.0f,0.0f,

1.0f,0.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,0.0f,1.0f,

0.0f,0.0f,1.0f,

0.0f,0.0f,1.0f,

1.0f,1.0f,0.0f,

1.0f,1.0f,0.0f,

1.0f,1.0f,0.0f,

0.0f,1.0f,1.0f,

0.0f,1.0f,1.0f,

0.0f,1.0f,1.0f,

1.0f,0.0f,1.0f,

1.0f,0.0f,1.0f,

1.0f,0.0f,1.0f,

1.0f,1.0f,1.0f,

1.0f,1.0f,1.0f,

1.0f,1.0f,1.0f,

0.5f,0.0f,0.0f,

0.5f,0.0f,0.0f,

0.5f,0.0f,0.0f,

1.0f,0.0f,0.0f,

1.0f,0.0f,0.0f,

1.0f,0.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,0.0f,1.0f,

0.0f,0.0f,1.0f,

0.0f,0.0f,1.0f,

1.0f,1.0f,0.0f,

1.0f,1.0f,0.0f,

1.0f,1.0f,0.0f,

0.0f,1.0f,1.0f,

0.0f,1.0f,1.0f,

0.0f,1.0f,1.0f,

};

GLfloat points[] = {

-1.0f,-0.8f,0.0f,

0.0f,-0.9f,0.0f,

-0.5f,0.0f,0.0f,

-0.3f,-0.5f,0.0f,

-0.3f,-0.87f,0.0f,

0.8f,-1.0f,0.0f,

0.8f,-1.0f,0.0f,

0.87f,-0.5f,0.0f,

-0.3f,-0.5f,0.0f,

0.87f,-0.5f,0.0f,

1.0f,0.8f,0.0f,

0.1f,0.87f,0.0f,

0.87f,-0.5f,0.0f,

0.4f,-0.5f,0.0f,

-0.1f,0.55f,0.0f,

0.87f,-0.5f,0.0f,

-0.1f,0.55f,0.0f,

0.4f,0.55f,0.0f,

0.1f,0.87f,0.0f,

-0.9f,1.0f,0.0f,

-0.9f,0.0f,0.0f,

-0.9f,0.0f,0.0f,

-0.1f,0.87f,0.0f,

-0.1f,0.0f,0.0f,

0.3f,0.87f,0.0f,

0.3f,0.5f,0.0f,

-0.1f,0.5f,0.0f,

0.3f,0.87f,0.0f,

-0.1f,0.9f,0.0f,

-0.1f,0.5f,0.0f,

};

glPolygonMode(GL\_FRONT\_AND\_BACK, GL\_POINT);

drawSmth(colors, points, startx, starty, size, GL\_TRIANGLES, 30);

glPolygonMode(GL\_FRONT\_AND\_BACK, GL\_FILL);

}

void drawPolygon3(GLfloat startx, GLfloat starty, GLfloat size) {

GLfloat colors[] = {

1.0f,0.0f,0.0f,

1.0f,0.0f,0.0f,

1.0f,0.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,0.0f,1.0f,

0.0f,0.0f,1.0f,

0.0f,0.0f,1.0f,

1.0f,1.0f,0.0f,

1.0f,1.0f,0.0f,

1.0f,1.0f,0.0f,

0.0f,1.0f,1.0f,

0.0f,1.0f,1.0f,

0.0f,1.0f,1.0f,

1.0f,0.0f,1.0f,

1.0f,0.0f,1.0f,

1.0f,0.0f,1.0f,

1.0f,1.0f,1.0f,

1.0f,1.0f,1.0f,

1.0f,1.0f,1.0f,

0.5f,0.0f,0.0f,

0.5f,0.0f,0.0f,

0.5f,0.0f,0.0f,

1.0f,0.0f,0.0f,

1.0f,0.0f,0.0f,

1.0f,0.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,0.0f,1.0f,

0.0f,0.0f,1.0f,

0.0f,0.0f,1.0f,

};

GLfloat points[] = {

-1.0f,-0.8f,0.0f,

0.0f,-0.9f,0.0f,

-0.5f,0.0f,0.0f,

-0.3f,-0.5f,0.0f,

-0.3f,-0.87f,0.0f,

0.8f,-1.0f,0.0f,

0.8f,-1.0f,0.0f,

0.87f,-0.5f,0.0f,

-0.3f,-0.5f,0.0f,

0.87f,-0.5f,0.0f,

1.0f,0.8f,0.0f,

0.1f,0.87f,0.0f,

0.87f,-0.5f,0.0f,

0.4f,-0.5f,0.0f,

-0.1f,0.55f,0.0f,

0.87f,-0.5f,0.0f,

-0.1f,0.55f,0.0f,

0.4f,0.55f,0.0f,

0.1f,0.87f,0.0f,

-0.9f,1.0f,0.0f,

-0.9f,0.0f,0.0f,

-0.9f,0.0f,0.0f,

-0.1f,0.87f,0.0f,

-0.1f,0.0f,0.0f,

0.3f,0.87f,0.0f,

0.3f,0.5f,0.0f,

-0.1f,0.5f,0.0f,

0.3f,0.87f,0.0f,

-0.1f,0.9f,0.0f,

-0.1f,0.5f,0.0f,

};

glPolygonMode(GL\_FRONT, GL\_FILL);

glPolygonMode(GL\_BACK, GL\_LINE);

drawSmth(colors, points, startx, starty, size, GL\_TRIANGLES, 30);

glPolygonMode(GL\_FRONT\_AND\_BACK, GL\_FILL);

}

void drawPolygon4(GLfloat startx, GLfloat starty, GLfloat size) {

GLfloat colors[] = {

1.0f,0.0f,0.0f,

1.0f,0.0f,0.0f,

1.0f,0.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,0.0f,1.0f,

0.0f,0.0f,1.0f,

0.0f,0.0f,1.0f,

1.0f,1.0f,0.0f,

1.0f,1.0f,0.0f,

1.0f,1.0f,0.0f,

0.0f,1.0f,1.0f,

0.0f,1.0f,1.0f,

0.0f,1.0f,1.0f,

1.0f,0.0f,1.0f,

1.0f,0.0f,1.0f,

1.0f,0.0f,1.0f,

1.0f,1.0f,1.0f,

1.0f,1.0f,1.0f,

1.0f,1.0f,1.0f,

0.5f,0.0f,0.0f,

0.5f,0.0f,0.0f,

0.5f,0.0f,0.0f,

1.0f,0.0f,0.0f,

1.0f,0.0f,0.0f,

1.0f,0.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,1.0f,0.0f,

0.0f,0.0f,1.0f,

0.0f,0.0f,1.0f,

0.0f,0.0f,1.0f,

};

GLfloat points[] = {

-1.0f,-0.8f,0.0f,

0.0f,-0.9f,0.0f,

-0.5f,0.0f,0.0f,

-0.3f,-0.5f,0.0f,

-0.3f,-0.87f,0.0f,

0.8f,-1.0f,0.0f,

0.8f,-1.0f,0.0f,

0.87f,-0.5f,0.0f,

-0.3f,-0.5f,0.0f,

0.87f,-0.5f,0.0f,

1.0f,0.8f,0.0f,

0.1f,0.87f,0.0f,

0.87f,-0.5f,0.0f,

0.4f,-0.5f,0.0f,

-0.1f,0.55f,0.0f,

0.87f,-0.5f,0.0f,

-0.1f,0.55f,0.0f,

0.4f,0.55f,0.0f,

0.1f,0.87f,0.0f,

-0.9f,1.0f,0.0f,

-0.9f,0.0f,0.0f,

-0.9f,0.0f,0.0f,

-0.1f,0.87f,0.0f,

-0.1f,0.0f,0.0f,

0.3f,0.87f,0.0f,

0.3f,0.5f,0.0f,

-0.1f,0.5f,0.0f,

0.3f,0.87f,0.0f,

-0.1f,0.9f,0.0f,

-0.1f,0.5f,0.0f,

};

glPolygonMode(GL\_FRONT\_AND\_BACK, GL\_LINE);

drawSmth(colors, points, startx, starty, size, GL\_TRIANGLES, 30);

glPolygonMode(GL\_FRONT\_AND\_BACK, GL\_FILL);

}

Shaderfuncs.h:

#pragma once

#include <iostream>

#include <string>

#include <GL/glew.h>

#include <GLFW/glfw3.h>

GLuint LoadShader(std::string filename, GLenum type);

Shaderfuncs.cpp:

#include "ShaderFuncs.h"

#include <fstream>

#include <vector>

GLuint LoadShader(std::string filename, GLenum type)

{

std::ifstream fin;

fin.open(filename);

if (fin) {

std::string textshader;

char c= fin.get();

while (c!=-1) {

textshader += c;

c = fin.get();

}

fin.close();

GLuint s = glCreateShader(type);

char\* buf = new GLchar[textshader.length() + 1];

strcpy\_s(buf, textshader.length()+1 , textshader.c\_str());

glShaderSource(s, 1, &buf, NULL);

glCompileShader(s);

GLint isCompiled = 0;

glGetShaderiv(s, GL\_COMPILE\_STATUS, &isCompiled);

if (isCompiled == GL\_FALSE)

{

GLint maxLength = 0;

glGetShaderiv(s, GL\_INFO\_LOG\_LENGTH, &maxLength);

std::vector<GLchar> errorLog(maxLength);

glGetShaderInfoLog(s, maxLength, &maxLength, &errorLog[0]);

for (GLchar c : errorLog)

std::cout << c;

}

delete[] buf;

return s;

}

else {

std::cout << "ERROR: No such file";

return -1;

}

}

FragmentShader.glsl:

#version 400

in vec3 colour;

out vec4 frag\_colour;

void main () {

frag\_colour = vec4 (colour, 1.0);

}

VertexShaner.glsl:

#version 400

in vec3 vertex\_position;

in vec3 vertex\_colour;

uniform mat4 transform;

out vec3 colour;

void main () {

colour = vertex\_colour;

gl\_Position = transform \* vec4 (vertex\_position, 1.0);

}