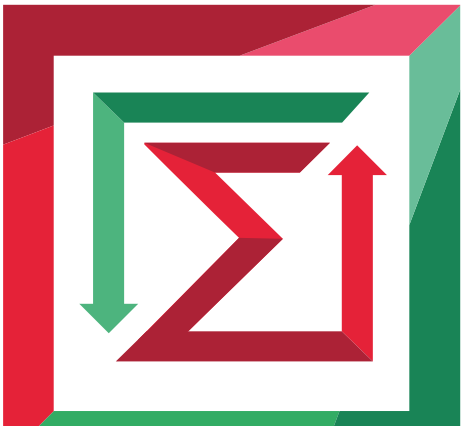
Министерство науки и высшего образования
Российской Федерации

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Новосибирский государственный технический университет»

Кафедра прикладной математики

Дополнительное задание № 1
по дисциплине «Компьютерная графика»

Место для ввода текста.



Факультет: ПМИ
Группа: ПМ-71
Студенты: Востриков Вячеслав
Бурдуков Вадим
Баштовой Павел
Преподаватель: Задорожный Александр Геннадьевич

Новосибирск
2020

Код программы:

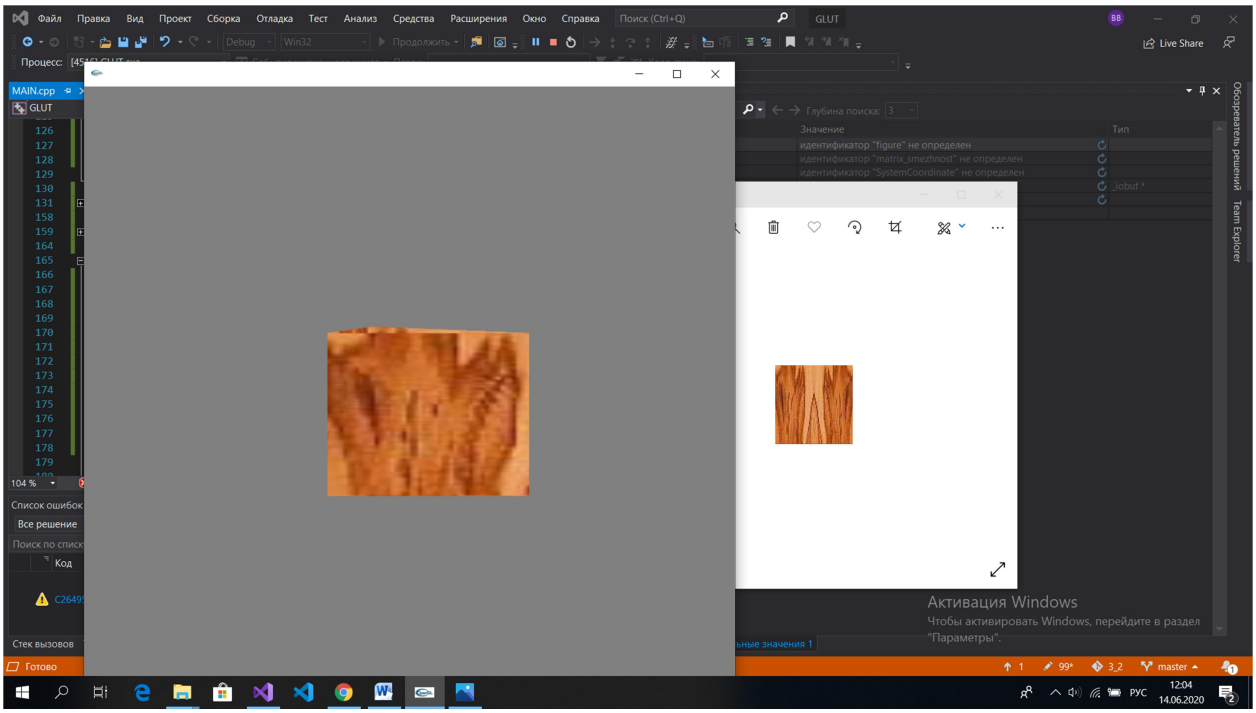
```
#define _CRT_SECURE_NO_WARNINGS
#include<stdlib.h>
#include <vector>
#include <fstream>
#include <stack>
#include <math.h>
#include "glut.h"
#include <soil.h> // Через Nuget Устанавливаем библиотеки для загрузки текстур


int Width = 800, Height = 800;
double time = 0;

#define speed_time 1
#define Pi 3.1415926

using namespace std;

FILE* file = fopen("text.bmp", "rb");

struct point3d {
	double x, y, z;

	point3d() {};
	point3d(double x_, double y_, double z_) {
		this->x = x_;
		this->y = y_;
		this->z = z_;
	}
	point3d operator+(point3d a) {
		this->x = this->x + a.x;
		this->y = this->y + a.y;
		this->z = this->z + a.z;
		return *this;
	}
	point3d operator-(point3d b) {
		this->x -= b.x;
		this->y -= b.y;
		this->z -= b.z;
		return *this;
	}
	point3d operator/(int k) {
		this->x = this->x / k;
		this->y = this->y / k;
		this->z = this->z / k;
		return *this;
	}
};

struct object {
	point3d points[6];
	point3d center_of_mass;

	// Вращение
	void init_object() {

	}

	object() {
		points[0] = point3d(0.5, 0, 0);
		points[1] = point3d(0.5, 0.5, 0);
		points[2] = point3d(0, 0.52, 0);
		points[3] = point3d(0, 0.02, 0);
```

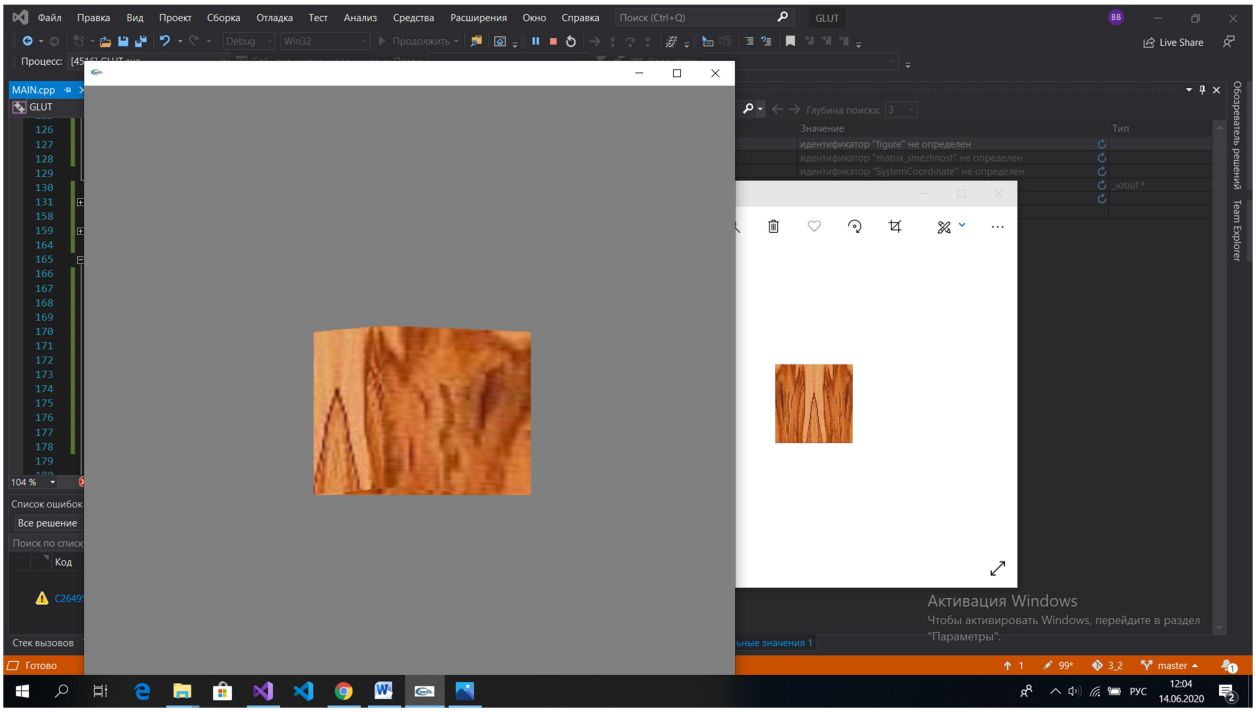
```
		points[4] = point3d(0, 0, 0.5);
		points[5] = point3d(0, 0.5, 0.5);

		point3d k(0, 0, 0);
		for (int i = 0; i < 6; i++)
			k = k + points[i];
		k = k / 6;
		center_of_mass = k;

		for (int i = 0; i < 6; i++)
			points[i] = points[i] - center_of_mass;
	}
} main_object;

void Display(void)
{
	glClearColor(0.5, 0.5, 0.5, 1);
	glClear(GL_COLOR_BUFFER_BIT);

	glEnable(GL_DEPTH_TEST); // включили Тест глубины
	glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
	glLoadIdentity();

	glRotated(time, 0, 1, 0);
	// Текстурирование
	glBegin(GL_POLYGON);
	glTexCoord2f(0, 0);
	glVertex3f(main_object.points[0].x, main_object.points[0].y,
main_object.points[0].z);
	glTexCoord2f(0, 1);
	glVertex3f(main_object.points[1].x, main_object.points[1].y,
main_object.points[1].z);
	glTexCoord2f(0.3, 1);
	glVertex3f(main_object.points[2].x, main_object.points[2].y,
main_object.points[2].z);
	glTexCoord2f(0.3, 0);
	glVertex3f(main_object.points[3].x, main_object.points[3].y,
main_object.points[3].z);

	glEnd();

	///
	glBegin(GL_POLYGON);
	glTexCoord2f(0.3, 0);
	glVertex3f(main_object.points[3].x, main_object.points[3].y,
main_object.points[3].z);
	glTexCoord2f(0.3, 1);
	glVertex3f(main_object.points[2].x, main_object.points[2].y,
main_object.points[2].z);
	glTexCoord2f(0.6, 1);
	glVertex3f(main_object.points[5].x, main_object.points[5].y,
main_object.points[5].z);
	glTexCoord2f(0.6, 0);
	glVertex3f(main_object.points[4].x, main_object.points[4].y,
main_object.points[4].z);

	glEnd();
	////
	glBegin(GL_POLYGON);
	glTexCoord2f(0.6, 0);
	glVertex3f(main_object.points[4].x, main_object.points[4].y,
main_object.points[4].z);
	glTexCoord2f(0.6, 1);
	glVertex3f(main_object.points[5].x, main_object.points[5].y,
main_object.points[5].z);
```

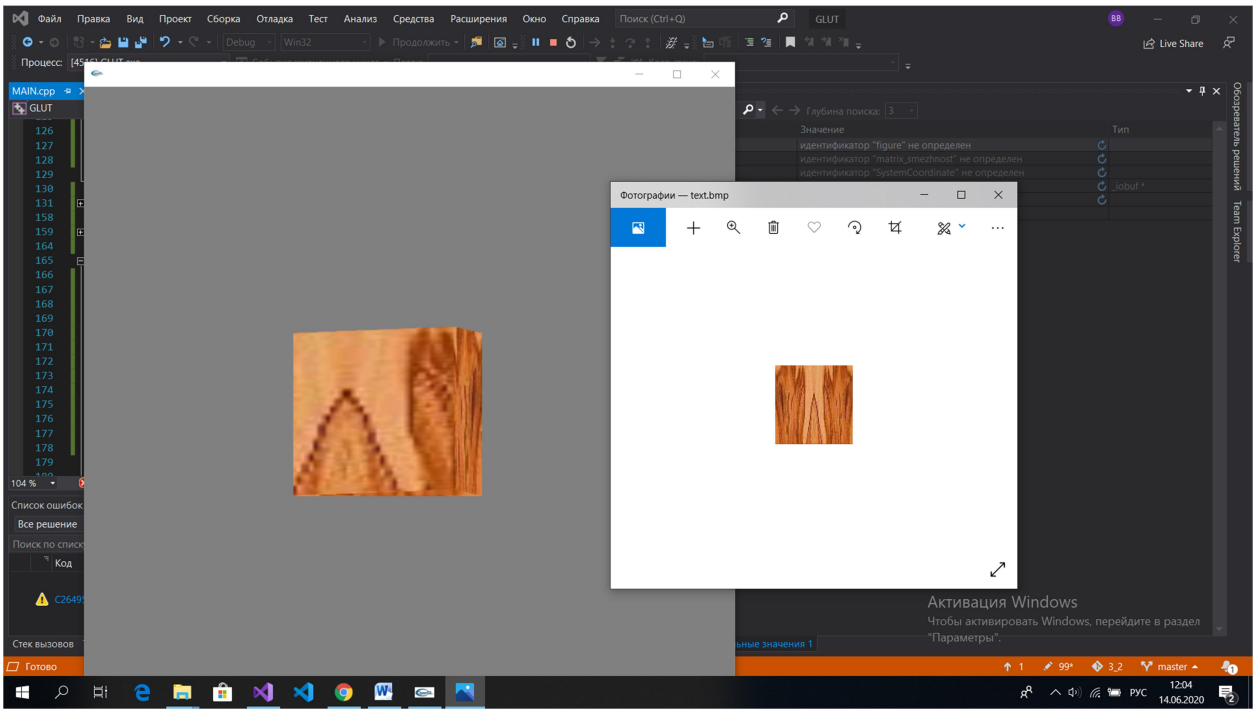
```
	glTexCoord2f(1, 1);
	glVertex3f(main_object.points[1].x, main_object.points[1].y,
main_object.points[1].z);
	glTexCoord2f(1, 0);
	glVertex3f(main_object.points[0].x, main_object.points[0].y,
main_object.points[0].z);

	glEnd();


	glFinish();
	glutSwapBuffers();
}

int read_texture() {
	unsigned char data54[54]; // служебный заголовок 54 байта

	if (file != NULL) {
		fread(data54, 1, 54, file);
		int size = *(data54 + 10);

		int width = *(data54 + 18);
		int height = *(data54 + 22);
		unsigned char* Pixels = new unsigned char[Width * Height * 3];

		fread(Pixels, Width * Height * 3, 1, file);
		fclose(file);
		file = NULL;
		int Type = GL_BGR_EXT;
		GLuint tex;
		glEnable(GL_TEXTURE_2D);			//Разрешение отображения
текстур
		glGenTextures(1, &tex);				//Генерация массива
номеров текстур
		glBindTexture(GL_TEXTURE_2D, tex);	//Выбор текущей текстуры по номеру
		gluBuild2DMipmaps(GL_TEXTURE_2D, 3, width, height, GL_BGR_EXT,
GL_UNSIGNED_BYTE, Pixels); //Задание режима выравнивания
		glTexEnvi(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_MODULATE); //Задание
режима учета параметров материала
	}
	else
		return -1;

	return 1;
}

void Timer(int) {
	time += speed_time;
	glutPostRedisplay();
	glutTimerFunc(10, Timer, 0);
}

int main(int argc, char* argv[])
{
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB);
	glutInitWindowSize(Width, Height);
	glutInitWindowPosition(100, 100);


	glutCreateWindow("");
	glutTimerFunc(10, Timer, 0);

	if (!read_texture()) return -1;
	glutDisplayFunc(Display);
```

```
        glutMainLoop();
        return 1;
}
```

Тестирование:






Фотографии — text.bmp
Активация Windows
Чтобы активировать Windows, перейдите в раздел "Параметры".