МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГБОУ ВО АЛТАЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Институт цифровых технологий, электроники и физики

Кафедра вычислительной техники и электроники (ВТиЭ)

Лабораторная работа № 3

**Простейшая анимация в OpenGL.**

Выполнил студент 595 гр.

\_\_\_\_\_\_\_\_\_\_\_\_\_ Лаптев А.В.

Проверил:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Калачев А.В.

Лабораторная работа защищена

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2023 г.

Барнаул 2023

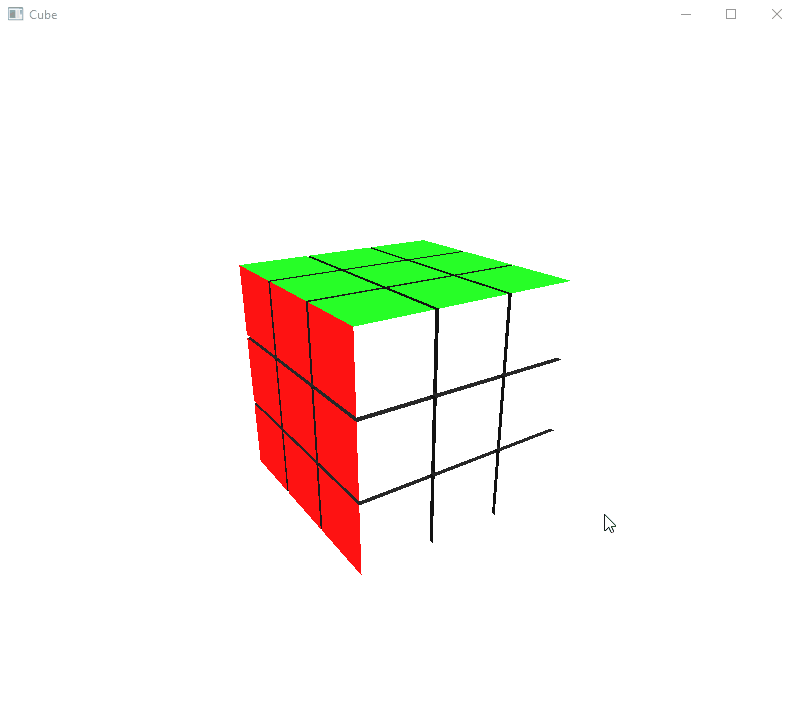
**Вариант 2.**

Разработайте OpenGL-приложение, визуализирующее послойно вращающийся (в различных плоскостях) кубик Рубика.

# Выполнение работы.

Для реализации задания используем библиотеку GLUT. Функции этой библиотеки включают построение простейших геометрических объектов.

Работа программы выглядит следующим образом:



**Вывод:** С использованием библиотек OpenGL была построена модель послойно вращающегося (в различных плоскостях) кубика Рубика.

# Приложение А

// lab3.cpp: определяет точку входа для приложения.

//

#include "stdafx.h"

#include "Cube.h"

#include <time.h>

#include <stdlib.h>

#define CUBE\_SIZE 13

#define TIMER 30

// обозначаем цвета:

// (верх, низ, впереди, сзади, лево, право)

unsigned int c[9] = { 0xFFFFFF, 0xFFFF00, 0x0000FF, 0x00FF00, 0xFF0000, 0xCD853F };

// координаты источника света

GLfloat lightPos[] = { 0, 100, 200, 0 };

// проекции угла поворота на оси

int xRot = 24, yRot = 34, zRot = 0;

// отдаление

double translateZ = -35.0;

// кубик-рубик

Cube cube;

// флаг того, крутится куб сам, или нет (будет переключаться правой кнопкой мыши)

int timerOn = 0;

void display()

{

glPushMatrix();

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glColor3f(1, 0, 0);

glLightfv(GL\_LIGHT0, GL\_POSITION, lightPos);

glTranslatef(0, 0, translateZ);

glRotatef(xRot, 1, 0, 0);

glRotatef(yRot, 0, 1, 0);

glTranslatef(CUBE\_SIZE / -2.0, CUBE\_SIZE / -2.0, CUBE\_SIZE / -2.0);

cube.draw();

glPopMatrix();

glutSwapBuffers();

}

void reshape(int w, int h)

{

glViewport(0, 0, w, h);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

GLfloat fAspect = (GLfloat)w / (GLfloat)h;

gluPerspective(60, fAspect, 1, 1000.0);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

}

void init()

{

glClearColor(1.0, 1.0, 1.0, 0.0);

// инициализируем случайные числа

srand(time(0));

// освещение

float mat\_specular[] = { 0.3, 0.3, 0.3, 0 };

float diffuseLight[] = { 0.2, 0.2, 0.2, 1 };

float ambientLight[] = { 0.9, 0.9, 0.9, 1.0 };

glShadeModel(GL\_SMOOTH);

glMaterialfv(GL\_FRONT, GL\_SPECULAR, mat\_specular);

glMateriali(GL\_FRONT, GL\_SHININESS, 128);

glLightfv(GL\_LIGHT0, GL\_DIFFUSE, diffuseLight);

glLightfv(GL\_LIGHT0, GL\_AMBIENT, ambientLight);

glEnable(GL\_LIGHT0);

glEnable(GL\_COLOR\_MATERIAL);

glColorMaterial(GL\_FRONT, GL\_AMBIENT);

glEnable(GL\_DEPTH\_TEST);

glEnable(GL\_LIGHTING);

// инициализируем куб

cube.clear(CUBE\_SIZE, c);

}

void specialKeys(int key, int, int)

{

// клавиши влево/вправо вращают по Y

// клавиши вверх/вниз вращают по X

// F1 - возвращает в начальное положение

if (key == GLUT\_KEY\_DOWN)

{

xRot += 3;

if (xRot >= 360)

xRot -= 360;

glutPostRedisplay();

}

if (key == GLUT\_KEY\_UP)

{

xRot -= 3;

if (xRot < 0)

xRot += 360;

glutPostRedisplay();

}

if (key == GLUT\_KEY\_RIGHT)

{

yRot += 3;

if (yRot >= 360)

yRot -= 360;

glutPostRedisplay();

}

if (key == GLUT\_KEY\_LEFT)

{

yRot -= 3;

if (yRot < 0)

yRot += 360;

glutPostRedisplay();

}

if (key == GLUT\_KEY\_HOME)

{

translateZ += 5;

glutPostRedisplay();

}

if (key == GLUT\_KEY\_END)

{

translateZ -= 5;

glutPostRedisplay();

}

if (key == GLUT\_KEY\_F1)

{

cube.clear(CUBE\_SIZE, c);

glutPostRedisplay();

}

}

void keys(unsigned char key, int, int)

{

// если нажали клавишу от 0 до 5 - начинаем поворот на 3 градуса

if (cube.current == -1 && key >= '0' && key < '6')

{

cube.Rotate(key - '0', 3);

display();

}

}

void mouse(int key, int state, int, int)

{

if (key == GLUT\_RIGHT\_BUTTON && state == GLUT\_DOWN)

{

// переключаем флаг

timerOn = 1 - timerOn;

}

}

void timer(int)

{

glutTimerFunc(TIMER, timer, 0);

if (timerOn)

{

// если включен автоматический поворот, и смотрим

// если сейчас никакая грань не крутится, то начинаем крутить случайную,

// иначе крутим текущую

if (cube.current == -1)

keys(rand() % 6 + '0', 0, 0);

else

cube.Rotate(cube.current, 3);

}

else

{

if (cube.current != -1)

cube.Rotate(cube.current, 3);

}

display();

}

int main(int argc, char\*\* argv)

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB | GLUT\_DEPTH);

glutInitWindowSize(800, 700);

glutInitWindowPosition(1, 1);

glutCreateWindow("Cube");

init();

glutDisplayFunc(display);

glutReshapeFunc(reshape);

glutKeyboardFunc(keys);

glutMouseFunc(mouse);

glutTimerFunc(TIMER, timer, 0);

glutSpecialFunc(specialKeys);

glutMainLoop();

return 0;

}