**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное бюджетное образовательное учреждение высшего**

**профессионального образования**

**«Казанский национальный исследовательский технический университет**

**им. А.Н. Туполева-КАИ»**

Институт компьютерных технологий и защиты информации

Лабораторная работа № 3

по дисциплине «Компьютерная графика»

**Выполнил**

студент группы 4211

Абдуллаев Л.Х.

Казань

2020

# Задание 1

Цель : Выполнить программу из раздела 5.

Листинг:

#include <math.h>

#include <Gl\glut.h>

GLfloat angle = 0;

void init(void)

{

glClearColor(1.0, 1.0, 0.0, 0.0);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluPerspective(60, 1, 1, 10);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

}

void myDisplay()

{

glPushMatrix(); //Сохраняем VM = 1

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

glEnable(GL\_DEPTH\_TEST);

gluLookAt(3, 2, 1, 0, 0, 0, 0, 1, 0); // VM=Fwe

GLfloat myLightPosition[] = { 3.0, 3.0, 3.0, 1.0 }; // Источник света

glLightfv(GL\_LIGHT0, GL\_POSITION, myLightPosition); /\*Позиция

источника света будет преобразована в CKe\*/

glEnable(GL\_LIGHTING);

glEnable(GL\_LIGHT0);

glPushMatrix(); //Сохраняем VM=Fwe

glRotatef(angle, 0, 1, 0); // VM=Fwe\*R

glRotatef(angle, 0, 1, 0); // VM=Fwe\*R\*R

glRotatef(angle, 0, 0, 1); // VM=Fwe\*R\*R\*R

glutSolidCube(0.95);

//glutWireCube(0.95);

glPopMatrix(); // Восстанавливаем VM=Fwe

glDisable(GL\_LIGHTING); //Выключаем освещение

glBegin(GL\_LINES);

glColor3f(1, 0, 0); glVertex3f(0, 0, 0); glVertex3f(1, 0, 0);

glColor3f(0, 1, 0); glVertex3f(0, 0, 0); glVertex3f(0, 1, 0);

glColor3f(0, 0, 1); glVertex3f(0, 0, 0); glVertex3f(0, 0, 1);

glEnd();

glBegin(GL\_LINES);

glColor3f(1, 1, 1); glVertex3f(3, 3, 3); glVertex3f(0.0, 0.0, 0.0);

glEnd();

glPopMatrix();

glutSwapBuffers();

}

void myReshape(int width, int height)

{

if (width / height < 1) glViewport(0, 0, width, width);

else glViewport(0, 0, height, height);

}

void myIdle()

{

angle += 0.5;

if (angle > 360.0) angle = 0;

glutPostRedisplay();

}

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

{

glutInit(&argc, argv);

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

glutInitWindowSize(800, 800);

glutInitWindowPosition(0, 0);

glutCreateWindow("my window");

glutDisplayFunc(myDisplay);

glutReshapeFunc(myReshape);

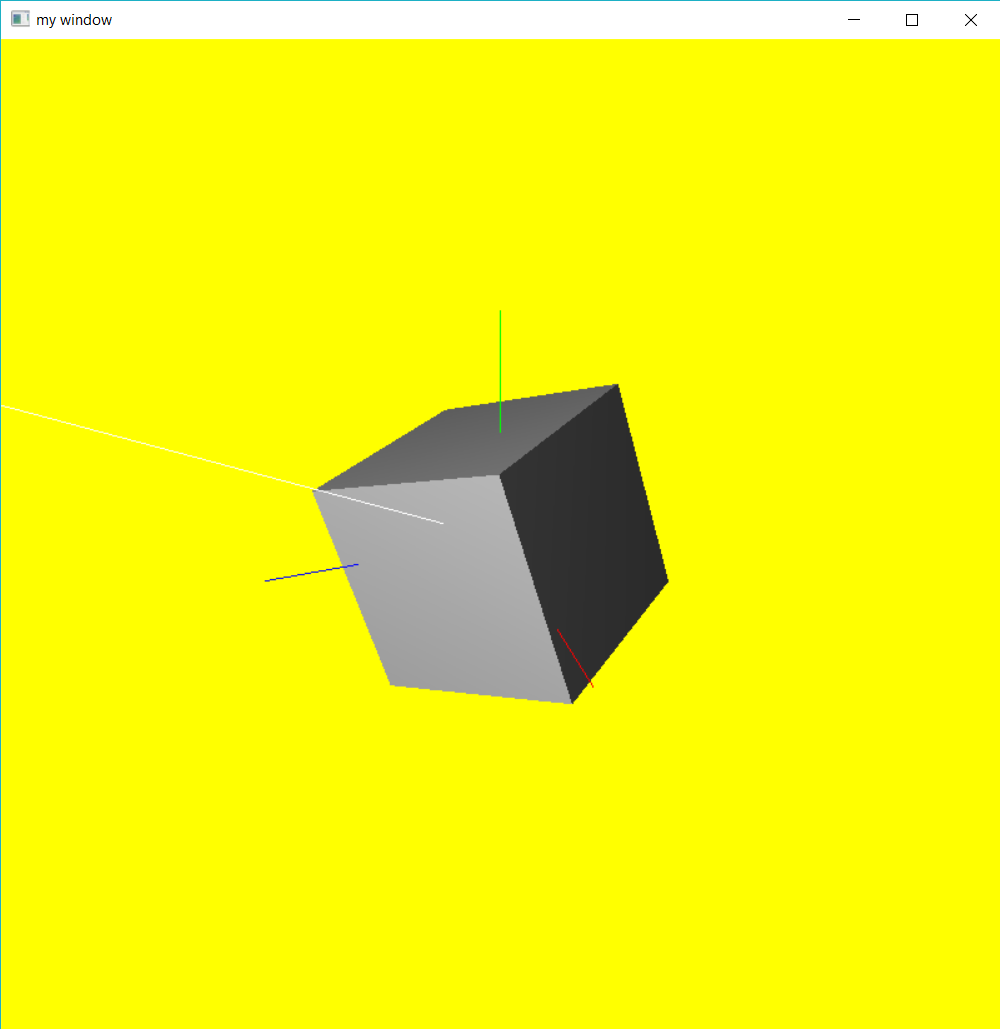
glutIdleFunc(myIdle);

init();

glutMainLoop();

}

Результат работы:



# Задание 2

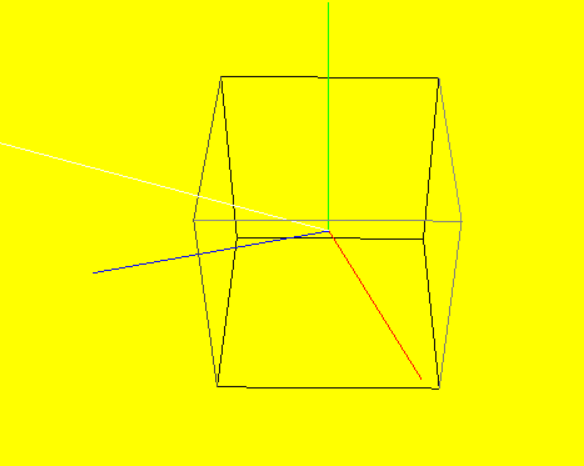
Цель : Внести изменения в программу:

**·** Сделать куб прозрачным.

Закомментировать строку glutSolidCube(0.95);

Раскомментировать строку glutWireCube(0.95);

Результат :



**·** Восстановить первоначальный вариант

**·** Уменьшить скорость вращения куба (добавить в программу вызов функции Sleep)

В функции myDisplay() добавляется строка Sleep(25);

**·** Сделать, чтобы куб вращался только вокруг оси X

Закомментировать строки:

//glRotatef(angle, 0, 1, 0);

//glRotatef(angle, 0, 0, 1);

**·** Сделать, чтобы куб вращался только вокруг оси Y

Раскомментировать строку glRotatef(angle, 0, 1, 0);

Закомментировать строку //glRotatef(angle, 0, 1, 0);

**·** Сделать, чтобы куб вращался в обратную сторону

Изменить строку glRotatef(angle, 0, 1, 0); на glRotatef(-angle, 0, 1, 0);

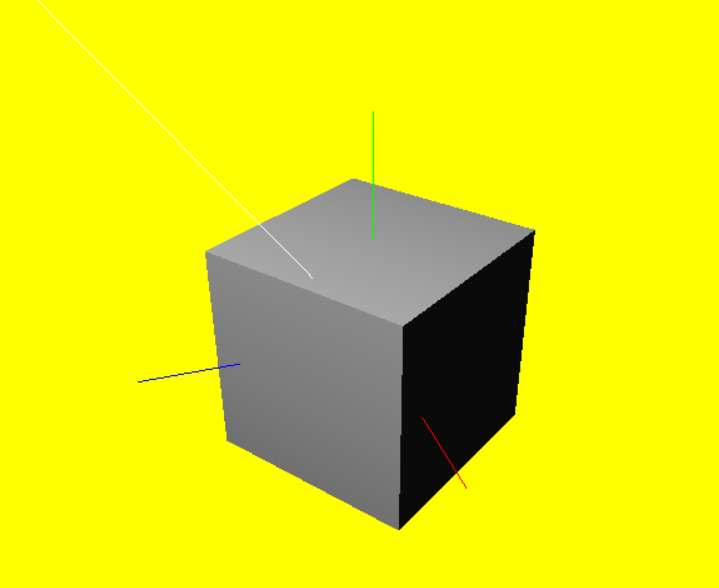
**·** Поменять положение источника света

В строке GLfloat myLightPosition[] = { 3.0, 3.0, 3.0, 1.0 }; изменить параметры: GLfloat myLightPosition[] = { 1.5, 3.0, 2.0, 1.0 };

В строке, рисующей белую линию источника света также изменить параметры:

glColor3f(1, 1, 1); glVertex3f(1.5, 3, 2); glVertex3f(0.0, 0.0, 0.0);

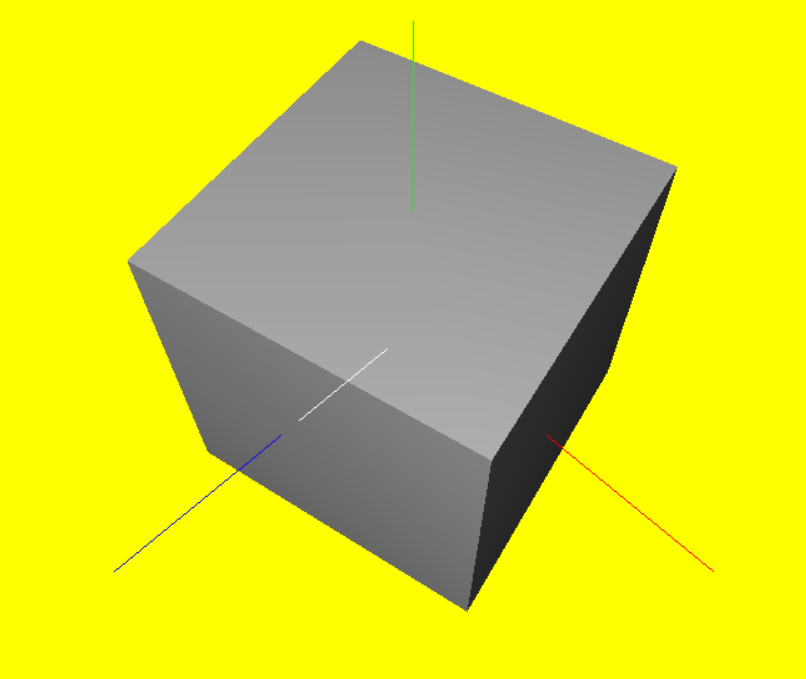
Результат:



**·** Поменять позицию наблюдателя

В строке gluLookAt(3, 2, 1, 0, 0, 0, 0, 1, 0); поменять параметры: gluLookAt(1, 2, 1, 0, 0, 0, 0, 1, 0);

Результат:

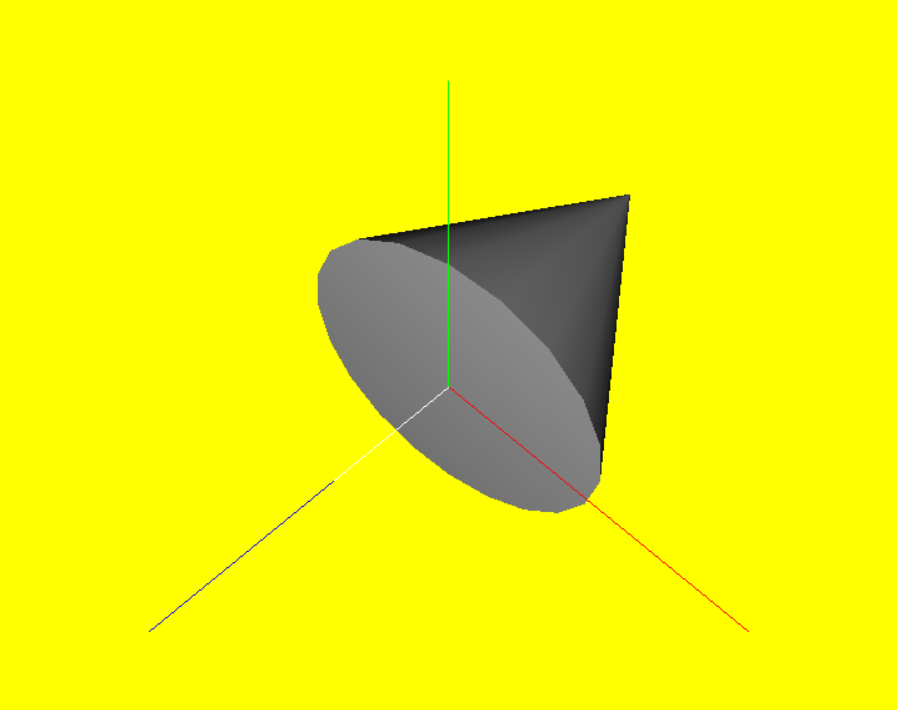


**·** Поменять куб на другую фигуру

Вместо строки glutSolidCube(0.95);

записать glutSolidCone(0.5, 1, 20, 40);

Результат:



Листинг:

#include <math.h>

#include <Gl\glut.h>

GLfloat angle = 0;

void init(void)

{

glClearColor(1.0, 1.0, 0.0, 0.0);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluPerspective(60, 1, 1, 10);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

}

void myDisplay()

{

glPushMatrix(); //Сохраняем VM = 1

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

glEnable(GL\_DEPTH\_TEST);

gluLookAt(1, 2, 1, 0, 0, 0, 0, 1, 0); // VM=Fwe

GLfloat myLightPosition[] = { 1.5, 3.0, 2.0, 1.0 }; // Источник света

glLightfv(GL\_LIGHT0, GL\_POSITION, myLightPosition); /\*Позиция

источника света будет преобразована в CKe\*/

glEnable(GL\_LIGHTING);

glEnable(GL\_LIGHT0);

glPushMatrix(); //Сохраняем VM=Fwe

//glRotatef(angle, 0, 1, 0); // VM=Fwe\*R

glRotatef(-angle, 0, 1, 0); // VM=Fwe\*R\*R

//glRotatef(angle, 0, 0, 1); // VM=Fwe\*R\*R\*R

glutSolidCone(0.5, 1, 20, 40);

//glutWireCube(0.95);

glPopMatrix(); // Восстанавливаем VM=Fwe

glDisable(GL\_LIGHTING); //Выключаем освещение

glBegin(GL\_LINES);

glColor3f(1, 0, 0); glVertex3f(0, 0, 0); glVertex3f(1, 0, 0);

glColor3f(0, 1, 0); glVertex3f(0, 0, 0); glVertex3f(0, 1, 0);

glColor3f(0, 0, 1); glVertex3f(0, 0, 0); glVertex3f(0, 0, 1);

glEnd();

glBegin(GL\_LINES);

glColor3f(1, 1, 1); glVertex3f(3, 3, 3); glVertex3f(0.0, 0.0, 0.0);

glEnd();

glPopMatrix();

glutSwapBuffers();

Sleep(25);

}

void myReshape(int width, int height)

{

if (width / height < 1) glViewport(0, 0, width, width);

else glViewport(0, 0, height, height);

}

void myIdle()

{

angle += 0.5;

if (angle > 360.0) angle = 0;

glutPostRedisplay();

}

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

{

glutInit(&argc, argv);

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

glutInitWindowSize(800, 800);

glutInitWindowPosition(0, 0);

glutCreateWindow("my window");

glutDisplayFunc(myDisplay);

glutReshapeFunc(myReshape);

glutIdleFunc(myIdle);

init();

glutMainLoop();

}

# Задание 3

Цель : Добавить в программу реакцию на нажатие клавиш:

**·** ЛКМ – увеличение размера фигуры

Ввести глобальную переменную GLfloat scale = 1;

Изменить функцию, рисующую фигуру :

glutSolidCone(scale\*0.5, scale\*1, 20, 40);

Написать функцию для обработки нажатия кнопок мыши:

void mouse(int button, int state, int x, int y) {

if (button == GLUT\_LEFT\_BUTTON && state == GLUT\_UP) { scale += 0.1; }

}

В функцию main() добавить строку glutMouseFunc(mouse);

**·** ПКМ – уменьшение размера фигуры

В функцию обработки нажатия mouse(int button, int state, int x, int y) добавить строку:

if (button == GLUT\_RIGHT\_BUTTON && state == GLUT\_UP) { scale -= 0.1; }

**·** Кл. V – направление вращения объекта меняется на противоположное

Ввести глобальную переменную GLfloat rot = 1;

Изменить функцию, вращающую фигуру:

glRotatef(-angle\*rot, 0, 1, 0);

Написать функцию обработки нажатий клавиш:

void move(unsigned char key, int x, int y) {

if (key == 'V' || key == 'v') { rot = -1 \* rot; }

}

В функцию main() добавить строку glutKeyboardFunc(move);

**·** Кл. X – наблюдатель приближается к объекту в направлении оси Х

Ввести глобальную переменную GLfloat loc = 1;

Изменить функцию, устанавливающую положение наблюдателя:

gluLookAt(loc\*1, 2, 1, 0, 0, 0, 0, 1, 0);

В функцию move(unsigned char key, int x, int y) добавить строку:

if (key == 'X' || key == 'x') { loc += 0.1; }

**·** Кл. Z – наблюдатель удаляется от объекта в направлении оси Х

В функцию move(unsigned char key, int x, int y) добавить строку:

if (key == 'Z' || key == 'z') { loc -= 0.1; }

Листинг:

#include <math.h>

#include <Gl\glut.h>

GLfloat angle = 0;

GLfloat scale = 1;

GLfloat rot = 1;

GLfloat loc = 1;

void init(void)

{

glClearColor(1.0, 1.0, 0.0, 0.0);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluPerspective(60, 1, 1, 10);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

}

void myDisplay()

{

glPushMatrix(); //Сохраняем VM = 1

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

glEnable(GL\_DEPTH\_TEST);

gluLookAt(loc\*1, 2, 1, 0, 0, 0, 0, 1, 0); // VM=Fwe

GLfloat myLightPosition[] = { 1.5, 3.0, 2.0, 1.0 }; // Источник света

glLightfv(GL\_LIGHT0, GL\_POSITION, myLightPosition); /\*Позиция

источника света будет преобразована в CKe\*/

glEnable(GL\_LIGHTING);

glEnable(GL\_LIGHT0);

glPushMatrix(); //Сохраняем VM=Fwe

//glRotatef(angle, 0, 1, 0); // VM=Fwe\*R

glRotatef(-angle\*rot, 0, 1, 0); // VM=Fwe\*R\*R

//glRotatef(angle, 0, 0, 1); // VM=Fwe\*R\*R\*R

glutSolidCone(scale\*0.5, scale\*1, 20, 40);

//glutWireCube(0.95);

glPopMatrix(); // Восстанавливаем VM=Fwe

glDisable(GL\_LIGHTING); //Выключаем освещение

glBegin(GL\_LINES);

glColor3f(1, 0, 0); glVertex3f(0, 0, 0); glVertex3f(1, 0, 0);

glColor3f(0, 1, 0); glVertex3f(0, 0, 0); glVertex3f(0, 1, 0);

glColor3f(0, 0, 1); glVertex3f(0, 0, 0); glVertex3f(0, 0, 1);

glEnd();

glBegin(GL\_LINES);

glColor3f(1, 1, 1); glVertex3f(1.5, 3, 2); glVertex3f(0.0, 0.0, 0.0);

glEnd();

glPopMatrix();

glutSwapBuffers();

Sleep(25);

}

void myReshape(int width, int height)

{

if (width / height < 1) glViewport(0, 0, width, width);

else glViewport(0, 0, height, height);

}

void myIdle()

{

angle += 0.5;

if (angle > 360.0) angle = 0;

glutPostRedisplay();

}

void mouse(int button, int state, int x, int y) {

if (button == GLUT\_LEFT\_BUTTON && state == GLUT\_UP) { scale += 0.1; }

if (button == GLUT\_RIGHT\_BUTTON && state == GLUT\_UP) { scale -= 0.1; }

}

void move(unsigned char key, int x, int y) {

if (key == 'V' || key == 'v') { rot = -1 \* rot; }

if (key == 'X' || key == 'x') { loc += 0.1; }

if (key == 'Z' || key == 'z') { loc -= 0.1; }

}

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

{

glutInit(&argc, argv);

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

glutInitWindowSize(800, 800);

glutInitWindowPosition(0, 0);

glutCreateWindow("my window");

glutDisplayFunc(myDisplay);

glutReshapeFunc(myReshape);

glutIdleFunc(myIdle);

glutMouseFunc(mouse);

glutKeyboardFunc(move);

init();

glutMainLoop();

}