Создайте программу, где геометрический примитив (ТОР) из библиотеки GLUT помещен в начало МСК, освещен неподвижным источником и в программе реализованы функции обработки нажатий клавиатуры и клавиш мыши в соответствии с таблицей (2 вариант):

**Листинг**

#include <gl\glut.h>

#include <math.h>

#include <Windows.h>

GLfloat angle = 0;

GLfloat scale = 1.0; // added scale factor

void init(void)

{

glClearColor(0.0, 1.0, 1.0, 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 в CKw

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

glScalef(scale, scale, scale); // added scaling

glutSolidTorus(0.2, 0.5, 100, 100);

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();

Sleep(1);

}

void myKeyboard(unsigned char key, int x, int y)

{

switch (key)

{

case 'x':

// Наблюдатель вращается в пл. (осьY, Е) вокруг объекта по ч.с.

glMatrixMode(GL\_MODELVIEW);

glTranslatef(0, 0, -3);

glRotatef(2, 0, 1, 0);

glTranslatef(0, 0, 3);

glutPostRedisplay();

break;

case 'y':

// Наблюдатель вращается в пл. (осьY, Е) против ч.с.

glMatrixMode(GL\_MODELVIEW);

glTranslatef(0, 0, -3);

glRotatef(-2, 0, 1, 0);

glTranslatef(0, 0, 3);

glutPostRedisplay();

break;

default:

break;

}

}

void myMouse(int button, int state, int x, int y)

{

switch (button)

{

case GLUT\_LEFT\_BUTTON:

// Объект масштабиру-ется с увеличением размеров

glScalef(1.1, 1.1, 1.1);

glutPostRedisplay();

break;

case GLUT\_RIGHT\_BUTTON:

// Объект масштабиру-ется с уменьшением размеров

glScalef(0.9, 0.9, 0.9);

glutPostRedisplay();

break;

default:

break;

}

}

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();

glutKeyboardFunc(myKeyboard);

glutMouseFunc(myMouse);

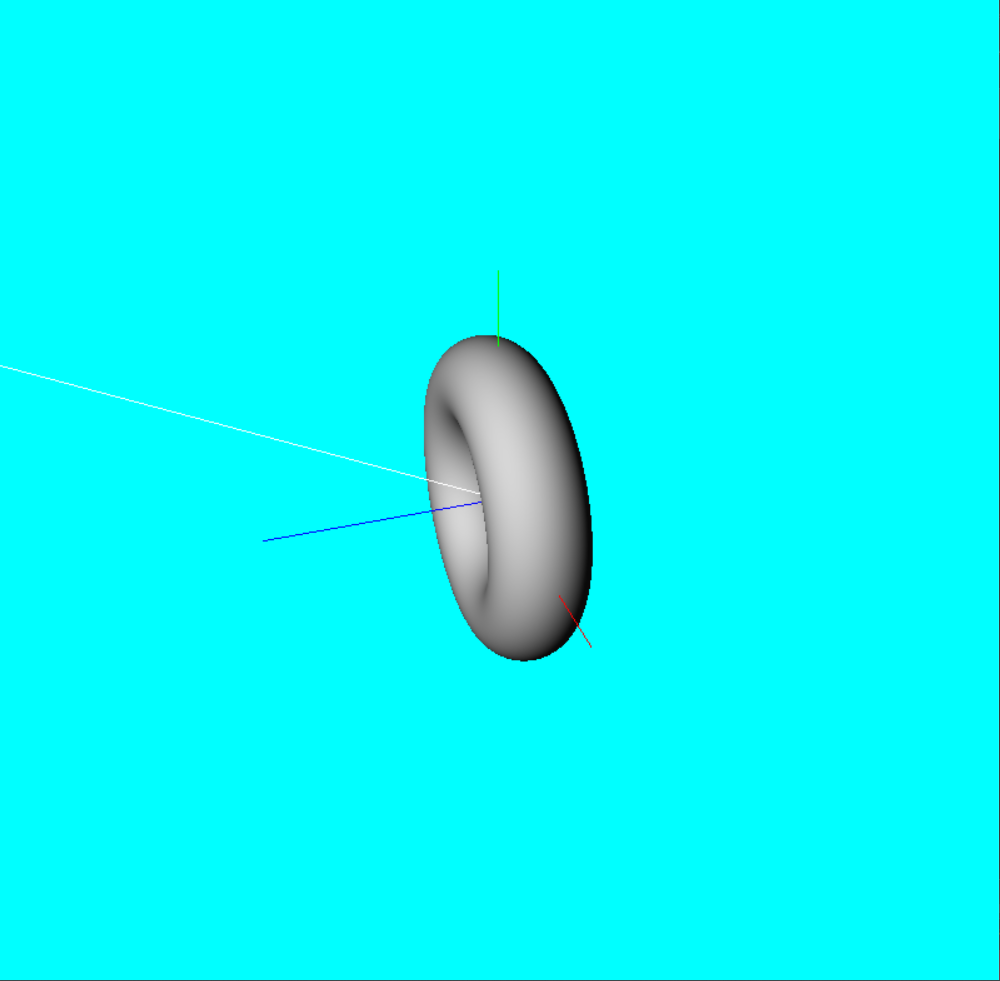
glutMainLoop();

}

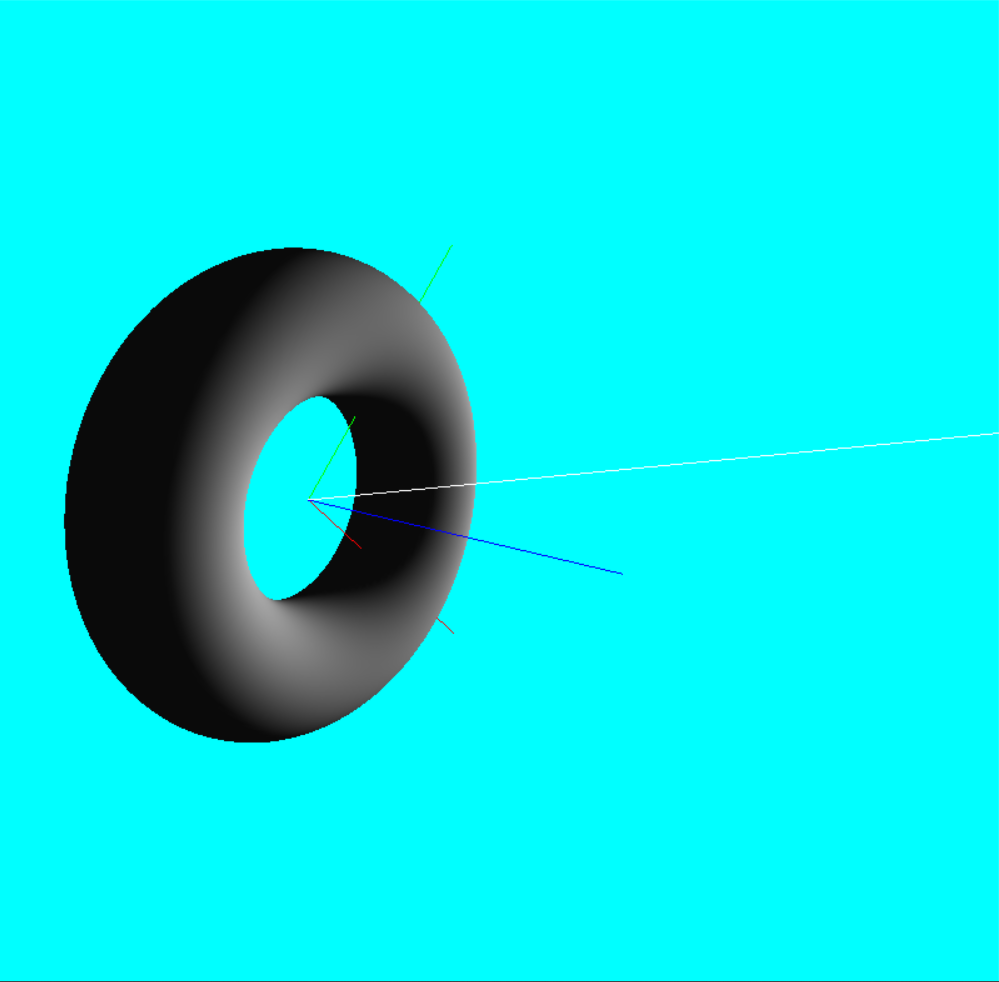
1. Кл . X - наблюдатель вращается в пл. (ось Y, Е) вокруг объекта по ч.с.

**Выполнение программы**

Начальное положение фигуры:

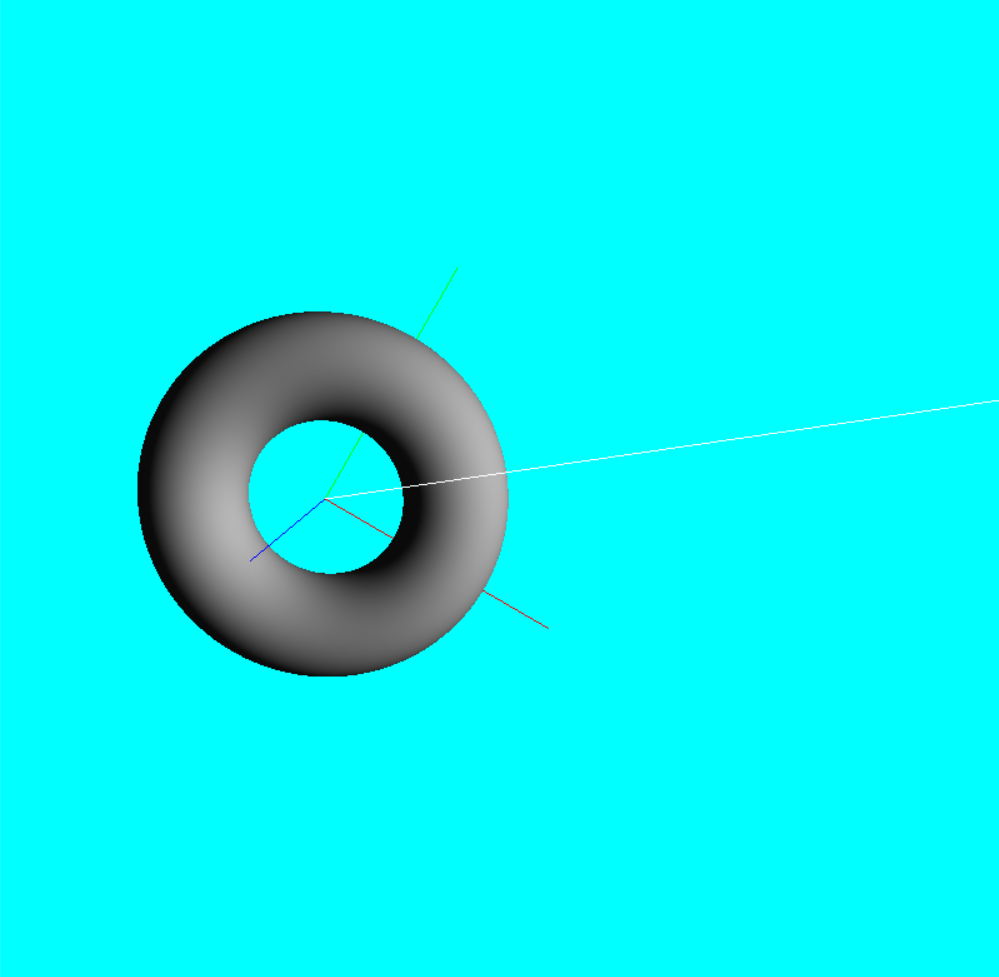


Конечное положение после нескольких нажатий клавиши Х:

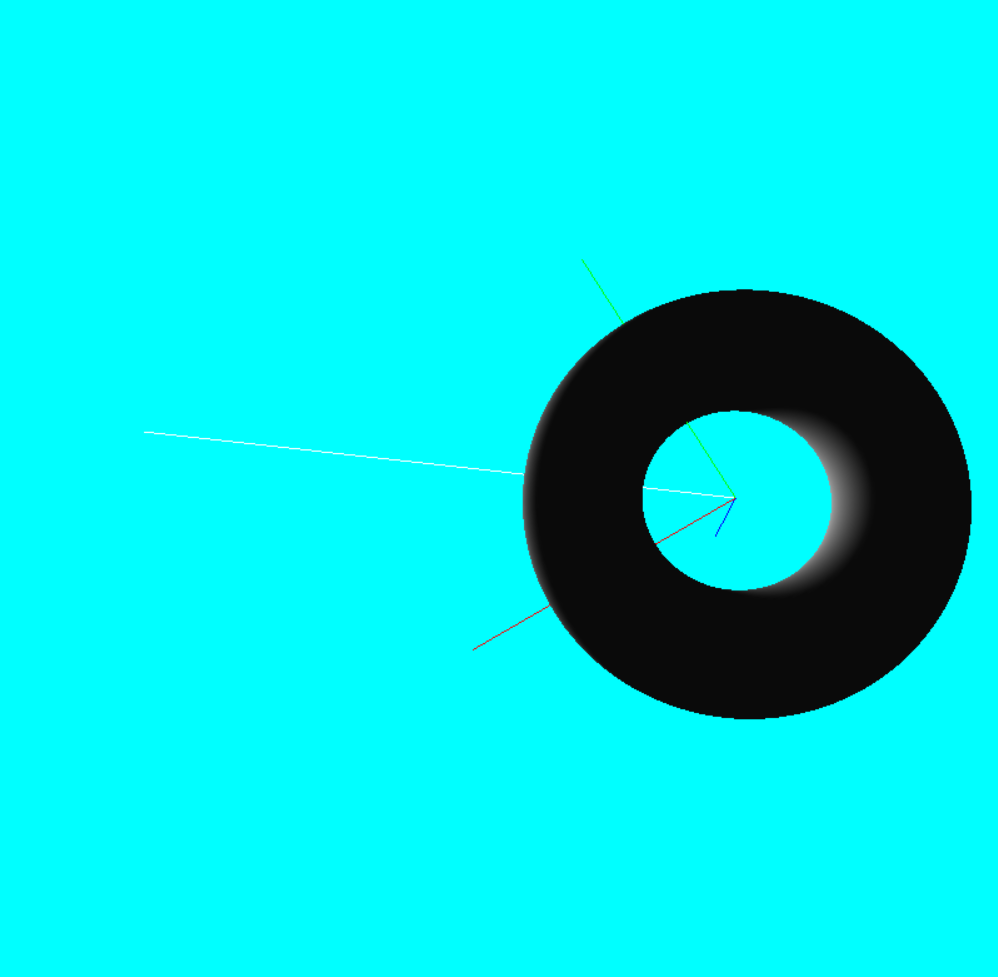


1. по кл. Y - наблюдатель вращается в пл. (осьY, Е) против ч.с.

Начальное положение фигуры:

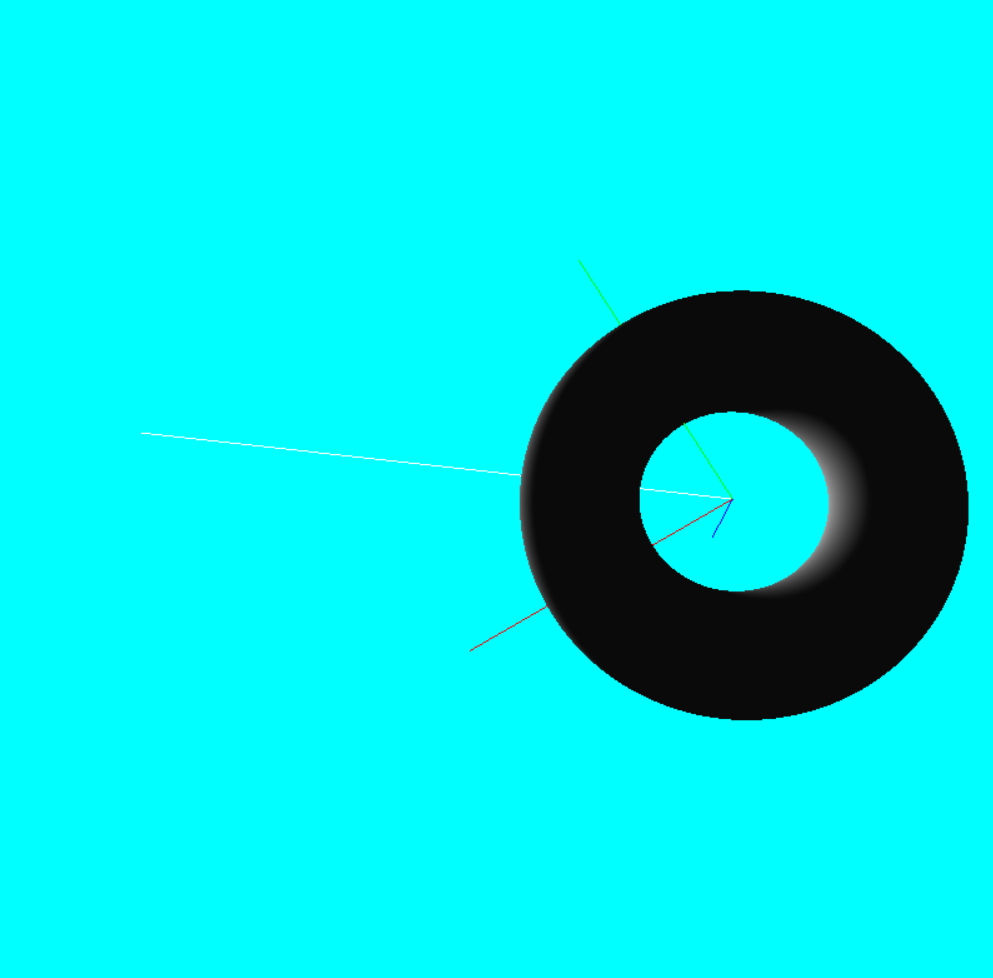


Конечное положение после нескольких нажатий клавиши Y:

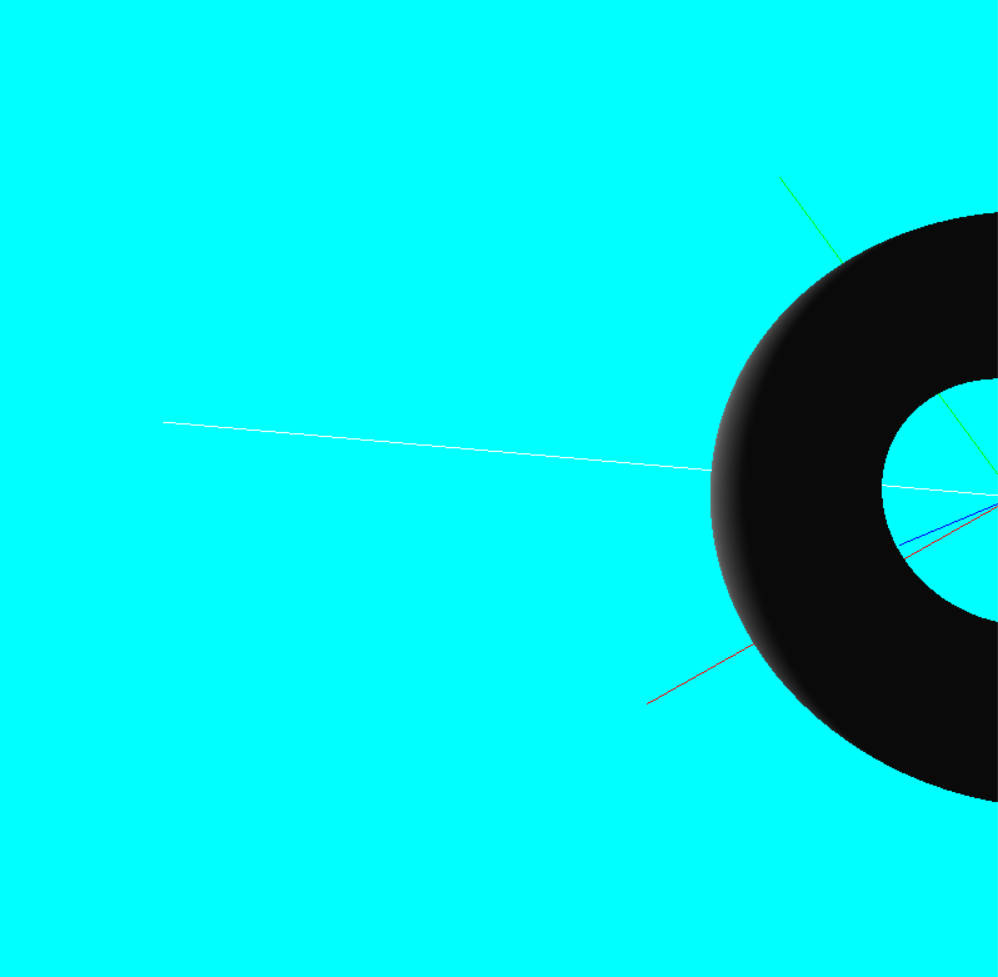


1. левая кл. – объект масштабируется с увеличением размеров

начальное положение:

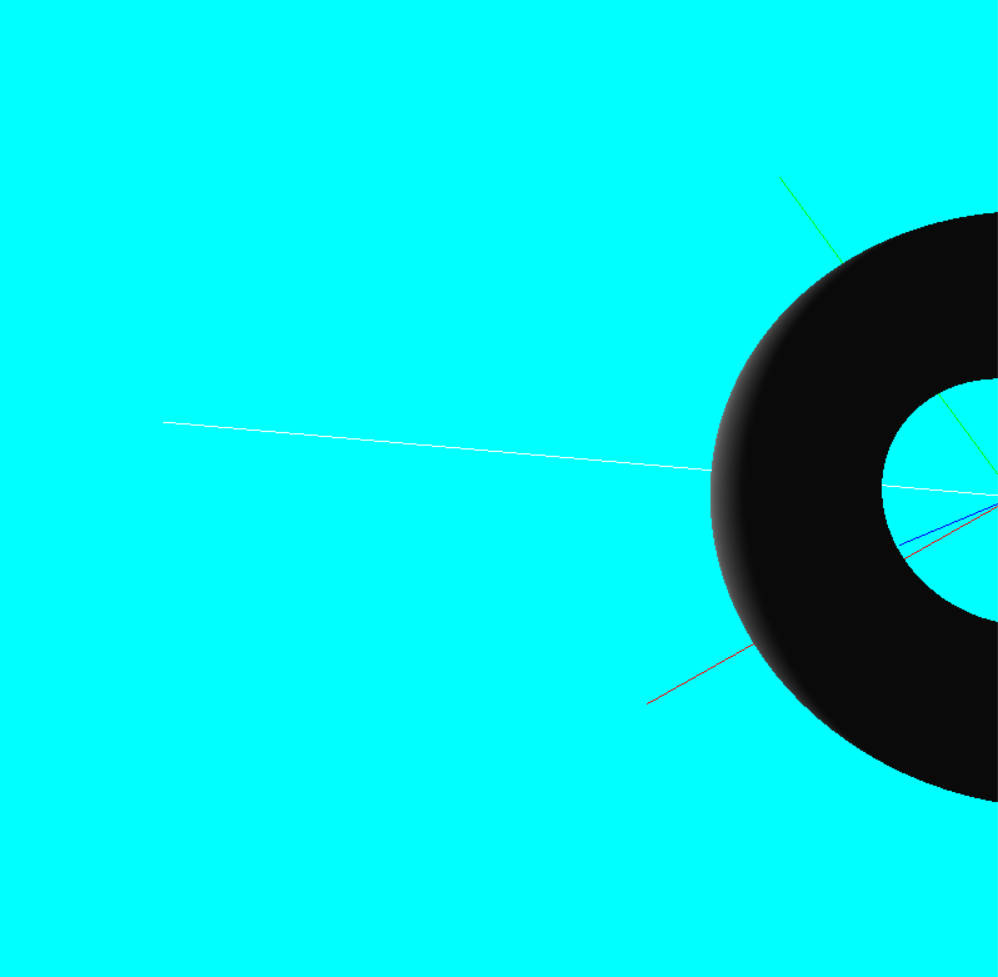


Конечное положение после одного нажатия ЛКМ:



1. правая кл. – объект масштабируется с уменьшением размеров

начальное положение:



Конечное положение после 3 нажатий ПКМ:

