```
#define GL_SILENCE_DEPRECATION

#ifdef __APPLE_CC__
#include <GLUT/glut.h>
#else
#include <GL/glut.h>
#endif
#include <stdlib.h>
#include <stdio.h>
#include <iostream>

GLboolean redFlag = true, switchOne = false, switchTwo=false,
switchLamp=false, amb1=true, diff1=true, spec1=true, amb2=true,
diff2=true, spec2=true, amb3=true, diff3=true, spec3=true;
double windowHeight=800, windowWidth=600;
double eyeX=7.0, eyeY=2.0, eyeZ=15.0, refX = 0, refY=0,refZ=0;
double theta = 180.0, y = 1.36, z = 7.97888;

static GLfloat v_cube[8][3] =
{
    {0.0, 0.0, 0.0}, //0
    {0.0, 0.0, 3.0}, //1
    {3.0, 0.0, 3.0}, //2
    {3.0, 0.0, 0.0}, //3
    {0.0, 3.0, 0.0}, //4
    {0.0, 3.0, 3.0}, //5
    {3.0, 3.0, 3.0}, //6
    {3.0, 3.0, 0.0}  //7
};

static GLubyte quadIndices[6][4] =
{
    {0, 1, 2, 3}, //bottom
    {4, 5, 6, 7}, //top
    {5, 1, 2, 6}, //front
    {0, 4, 7, 3}, // back is clockwise
    {2, 3, 7, 6}, //right
    {1, 5, 4, 0}  //left is clockwise
};



static void getNormal3p
(GLfloat x1, GLfloat y1,GLfloat z1, GLfloat x2, GLfloat y2,GLfloat z2,
GLfloat x3, GLfloat y3,GLfloat z3)
{
    GLfloat Ux, Uy, Uz, Vx, Vy, Vz, Nx, Ny, Nz;

    Ux = x2-x1;
    Uy = y2-y1;
    Uz = z2-z1;

    Vx = x3-x1;
    Vy = y3-y1;
```

```
    Vz = z3-z1;

    Nx = Uy*Vz - Uz*Vy;
    Ny = Uz*Vx - Ux*Vz;
    Nz = Ux*Vy - Uy*Vx;

    glNormal3f(Nx,Ny,Nz);
}


void drawCube()
{
    glBegin(GL_QUADS);
    for (GLint i = 0; i <6; i++)
    {
        getNormal3p(v_cube[quadIndices[i][0]][0],
v_cube[quadIndices[i][0]][1], v_cube[quadIndices[i][0]][2],
                    v_cube[quadIndices[i][1]][0],
v_cube[quadIndices[i][1]][1], v_cube[quadIndices[i][1]][2],
                    v_cube[quadIndices[i][2]][0],
v_cube[quadIndices[i][2]][1], v_cube[quadIndices[i][2]][2]);
        glVertex3fv(&v_cube[quadIndices[i][0]][0]);
        glVertex3fv(&v_cube[quadIndices[i][1]][0]);
        glVertex3fv(&v_cube[quadIndices[i][2]][0]);
        glVertex3fv(&v_cube[quadIndices[i][3]][0]);
    }
    glEnd();
}

void drawCube1(GLfloat difX, GLfloat difY, GLfloat difZ, GLfloat ambX=0,
GLfloat ambY=0, GLfloat ambZ=0, GLfloat shine=50)
{
    GLfloat no_mat[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat mat_ambient[] = { ambX, ambY, ambZ, 1.0 };
    GLfloat mat_diffuse[] = { difX, difY, difZ, 1.0 };
    GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat mat_shininess[] = {shine};

    glMaterialfv( GL_FRONT, GL_AMBIENT, mat_ambient);
    glMaterialfv( GL_FRONT, GL_DIFFUSE, mat_diffuse);
    glMaterialfv( GL_FRONT, GL_SPECULAR, mat_specular);
    glMaterialfv( GL_FRONT, GL_SHININESS, mat_shininess);
    glMaterialfv(GL_FRONT, GL_EMISSION, no_mat);

    glBegin(GL_QUADS);

    for (GLint i = 0; i <6; i++)
    {
        getNormal3p(v_cube[quadIndices[i][0]][0],
v_cube[quadIndices[i][0]][1], v_cube[quadIndices[i][0]][2],
                    v_cube[quadIndices[i][1]][0],
v_cube[quadIndices[i][1]][1], v_cube[quadIndices[i][1]][2],
                    v_cube[quadIndices[i][2]][0],
v_cube[quadIndices[i][2]][1], v_cube[quadIndices[i][2]][2]);
```

```
        glVertex3fv(&v_cube[quadIndices[i][0]][0]);
        glVertex3fv(&v_cube[quadIndices[i][1]][0]);
        glVertex3fv(&v_cube[quadIndices[i][2]][0]);
        glVertex3fv(&v_cube[quadIndices[i][3]][0]);
    }
    glEnd();
}

static GLfloat v_trapezoid[8][3] =
{
    {0.0, 0.0, 0.0}, //0
    {0.0, 0.0, 3.0}, //1
    {3.0, 0.0, 3.0}, //2
    {3.0, 0.0, 0.0}, //3
    {0.5, 3.0, 0.5}, //4
    {0.5, 3.0, 2.5}, //5
    {2.5, 3.0, 2.5}, //6
    {2.5, 3.0, 0.5}  //7
};

static GLubyte TquadIndices[6][4] =
{
    {0, 1, 2, 3}, //bottom
    {4, 5, 6, 7}, //top
    {5, 1, 2, 6}, //front
    {0, 4, 7, 3}, // back is clockwise
    {2, 3, 7, 6}, //right
    {1, 5, 4, 0}  //left is clockwise
};

void drawTrapezoid(GLfloat difX, GLfloat difY, GLfloat difZ, GLfloat
ambX, GLfloat ambY, GLfloat ambZ, GLfloat shine=50)
{
    GLfloat no_mat[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat mat_ambient[] = { ambX, ambY, ambZ, 1.0 };
    GLfloat mat_diffuse[] = { difX, difY, difZ, 1.0 };
    GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat mat_emission[] = {difX, difY, difZ, 0.0};
    GLfloat mat_shininess[] = {shine};

    glMaterialfv( GL_FRONT, GL_AMBIENT, mat_ambient);
    glMaterialfv( GL_FRONT, GL_DIFFUSE, mat_diffuse);
    glMaterialfv( GL_FRONT, GL_SPECULAR, mat_specular);
    glMaterialfv( GL_FRONT, GL_SHININESS, mat_shininess);

    if(switchLamp==true){
    glMaterialfv(GL_FRONT, GL_EMISSION, mat_emission);
    } else {
        glMaterialfv(GL_FRONT, GL_EMISSION, no_mat);
    }

    glBegin(GL_QUADS);
    for (GLint i = 0; i <6; i++)
    {
```

```
        getNormal3p(v_trapezoid[TquadIndices[i][0]][0],
v_trapezoid[TquadIndices[i][0]][1], v_trapezoid[TquadIndices[i][0]][2],
        v_trapezoid[TquadIndices[i][1]][0],
v_trapezoid[TquadIndices[i][1]][1], v_trapezoid[TquadIndices[i][1]][2],
        v_trapezoid[TquadIndices[i][2]][0],
v_trapezoid[TquadIndices[i][2]][1], v_trapezoid[TquadIndices[i][2]][2]);

        glVertex3fv(&v_trapezoid[TquadIndices[i][0]][0]);
        glVertex3fv(&v_trapezoid[TquadIndices[i][1]][0]);
        glVertex3fv(&v_trapezoid[TquadIndices[i][2]][0]);
        glVertex3fv(&v_trapezoid[TquadIndices[i][3]][0]);
    }
    glEnd();
}


//Drawing pyramid ************************
static GLfloat v_pyramid[5][3] =
{
    {0.0, 0.0, 0.0},
    {0.0, 0.0, 2.0},
    {2.0, 0.0, 2.0},
    {2.0, 0.0, 0.0},
    {1.0, 4.0, 1.0}
};

static GLubyte p_Indices[4][3] =
{
    {4, 1, 2},
    {4, 2, 3},
    {4, 3, 0},
    {4, 0, 1}
};

static GLubyte PquadIndices[1][4] =
{
    {0, 3, 2, 1}
};

void drawpyramid(GLfloat difX, GLfloat difY, GLfloat difZ, GLfloat ambX,
GLfloat ambY, GLfloat ambZ, GLfloat shine)
{
    GLfloat no_mat[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat mat_ambient[] = { ambX, ambY, ambZ, 1.0 };
    GLfloat mat_diffuse[] = { difX, difY, difZ, 1.0 };
    GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat mat_shininess[] = {shine};

    glMaterialfv( GL_FRONT, GL_AMBIENT, mat_ambient);
    glMaterialfv( GL_FRONT, GL_DIFFUSE, mat_diffuse);
    glMaterialfv( GL_FRONT, GL_SPECULAR, mat_specular);
    glMaterialfv( GL_FRONT, GL_SHININESS, mat_shininess);

    glBegin(GL_TRIANGLES);
```

```
    for (GLint i = 0; i <4; i++)
    {
        getNormal3p(v_pyramid[p_Indices[i][0]][0],
v_pyramid[p_Indices[i][0]][1], v_pyramid[p_Indices[i][0]][2],
                    v_pyramid[p_Indices[i][1]][0],
v_pyramid[p_Indices[i][1]][1], v_pyramid[p_Indices[i][1]][2],
                    v_pyramid[p_Indices[i][2]][0],
v_pyramid[p_Indices[i][2]][1], v_pyramid[p_Indices[i][2]][2]);
        glVertex3fv(&v_pyramid[p_Indices[i][0]][0]);
        glVertex3fv(&v_pyramid[p_Indices[i][1]][0]);
        glVertex3fv(&v_pyramid[p_Indices[i][2]][0]);
    }
    glEnd();

    glBegin(GL_QUADS);
    for (GLint i = 0; i <1; i++)
    {
        getNormal3p(v_pyramid[PquadIndices[i][0]][0],
v_pyramid[PquadIndices[i][0]][1], v_pyramid[PquadIndices[i][0]][2],
                    v_pyramid[PquadIndices[i][1]][0],
v_pyramid[PquadIndices[i][1]][1], v_pyramid[PquadIndices[i][1]][2],
                    v_pyramid[PquadIndices[i][2]][0],
v_pyramid[PquadIndices[i][2]][1], v_pyramid[PquadIndices[i][2]][2]);
        glVertex3fv(&v_pyramid[PquadIndices[i][0]][0]);
        glVertex3fv(&v_pyramid[PquadIndices[i][1]][0]);
        glVertex3fv(&v_pyramid[PquadIndices[i][2]][0]);
        glVertex3fv(&v_pyramid[PquadIndices[i][3]][0]);
    }
    glEnd();

}

void polygon(GLfloat difX, GLfloat difY, GLfloat difZ, GLfloat ambX,
GLfloat ambY, GLfloat ambZ, GLfloat shine)
{
    GLfloat no_mat[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat mat_ambient[] = { ambX, ambY, ambZ, 1.0 };
    GLfloat mat_diffuse[] = { difX, difY, difZ, 1.0 };
    GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat mat_shininess[] = {shine};

    glMaterialfv( GL_FRONT, GL_AMBIENT, mat_ambient);
    glMaterialfv( GL_FRONT, GL_DIFFUSE, mat_diffuse);
    glMaterialfv( GL_FRONT, GL_SPECULAR, mat_specular);
    glMaterialfv( GL_FRONT, GL_SHININESS, mat_shininess);

    glBegin(GL_POLYGON);
    glVertex2f(0,0);
    glVertex2f(6,0);
    glVertex2f(5.8,1);
    glVertex2f(5.2,2);
    glVertex2f(5, 2.2);
    glVertex2f(4, 2.8);
    glVertex2f(3,3);
```

```
    glVertex2f(2, 2.8);
    glVertex2f(1, 2.2);
    glVertex2f(0.8, 2);
    glVertex2f(0.2,1);
    //glVertex2f(0,0);

    glEnd();
}

void polygonLine(GLfloat difX, GLfloat difY, GLfloat difZ, GLfloat ambX,
GLfloat ambY, GLfloat ambZ, GLfloat shine)
{
    GLfloat no_mat[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat mat_ambient[] = { ambX, ambY, ambZ, 1.0 };
    GLfloat mat_diffuse[] = { difX, difY, difZ, 1.0 };
    GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat mat_shininess[] = {shine};

    glMaterialfv( GL_FRONT, GL_AMBIENT, mat_ambient);
    glMaterialfv( GL_FRONT, GL_DIFFUSE, mat_diffuse);
    glMaterialfv( GL_FRONT, GL_SPECULAR, mat_specular);
    glMaterialfv( GL_FRONT, GL_SHININESS, mat_shininess);
    glBegin(GL_LINE_STRIP);
    //glVertex2f(0,0);
    glVertex2f(6,0);
    glVertex2f(5.8,1);
    glVertex2f(5.2,2);
    glVertex2f(5, 2.2);
    glVertex2f(4, 2.8);
    glVertex2f(3,3);
    glVertex2f(2, 2.8);
    glVertex2f(1, 2.2);
    glVertex2f(0.8, 2);
    glVertex2f(0.2,1);
    glVertex2f(0,0);

    glEnd();
}

void drawSphere(GLfloat difX, GLfloat difY, GLfloat difZ, GLfloat ambX,
GLfloat ambY, GLfloat ambZ, GLfloat shine=90)
{
    GLfloat no_mat[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat mat_ambient[] = { ambX, ambY, ambZ, 1.0 };
    GLfloat mat_diffuse[] = { difX, difY, difZ, 1.0 };
    GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat mat_shininess[] = {shine};

    glMaterialfv( GL_FRONT, GL_AMBIENT, mat_ambient);
    glMaterialfv( GL_FRONT, GL_DIFFUSE, mat_diffuse);
    glMaterialfv( GL_FRONT, GL_SPECULAR, mat_specular);
    glMaterialfv( GL_FRONT, GL_SHININESS, mat_shininess);

    glutSolidSphere (3.0, 20, 16);
```

```
}

void cupboard()
{
    //Cupboard/Almari
**********************************************************

        //cupboard
    glPushMatrix();
    glTranslatef(4,0,4.4);
    glScalef(0.5, 1, 0.5);
    drawCube1(0.5,0.2,0.2,  0.25, 0.1, 0.1 );
    glPopMatrix();

    //cupboard's 1st vertical stripline
    glPushMatrix();
    glTranslatef(4,1,5.9);
    glScalef(0.5, 0.01, 0.0001);
    drawCube1(0.2,0.1,0.1,  0.1, 0.05, 0.05);
    glPopMatrix();

    //cupboard's 2nd vertical stripline
    glPushMatrix();
    glTranslatef(4,0.5,5.9);
    glScalef(0.5, 0.01, 0.0001);
    drawCube1(0.2,0.1,0.1,  0.1, 0.05, 0.05);
    glPopMatrix();

    //cupboard's last stripline
    glPushMatrix();
    glTranslatef(4,0,5.9);
    glScalef(0.5, 0.01, 0.0001);
    drawCube1(0.2,0.1,0.1,  0.1, 0.05, 0.05);
    glPopMatrix();

    //cupboard's lst horizontal stripline
    glPushMatrix();
    glTranslatef(5.5,0,5.9);
    glScalef(0.01, 1, 0.0001);
    drawCube1(0.2,0.1,0.1,  0.1, 0.05, 0.05);
    glPopMatrix();

    //cupboard's right side horizontal stripline
    glPushMatrix();
    glTranslatef(4.75,1,5.9);
    glScalef(0.01, 0.67, 0.0001);
    drawCube1(0.2,0.1,0.1,  0.1, 0.05, 0.05);
    glPopMatrix();

    //cupboard's left side horizontal stripline
    glPushMatrix();
    glTranslatef(4,0,5.9);
    glScalef(0.01, 1, 0.0001);
    drawCube1(0.2,0.1,0.1,  0.1, 0.05, 0.05);
```

```
    glPopMatrix();

    //cupboard's handle right
    glPushMatrix();
    glTranslatef(5,1.4,5.9);
    glScalef(0.02, 0.18, 0.01);
    drawCube1(0.2,0.1,0.1,  0.1, 0.05, 0.05);
    glPopMatrix();

    //cupboard's handle right sphere
    glPushMatrix();
    glTranslatef(5.02,1.9,5.91);
    glScalef(0.02, 0.02, 0.01);
    drawSphere(0.2,0.1,0.1,  0.1, 0.05, 0.05,  10);
    glPopMatrix();

    //cupboard's handle left
    glPushMatrix();
    glTranslatef(4.5,1.4,5.9);
    glScalef(0.02, 0.18, 0.01);
    drawCube1(0.2,0.1,0.1,  0.1, 0.05, 0.05);
    glPopMatrix();

    //cupboard's handle left sphere
    glPushMatrix();
    glTranslatef(4.52,1.9,5.91);
    glScalef(0.02, 0.02, 0.01);
    drawSphere(0.2,0.1,0.1,  0.1, 0.05, 0.05,  10);
    glPopMatrix();

    //cupboard's drawer's 1st handle
    glPushMatrix();
    glTranslatef(4.5,0.7,5.9);
    glScalef(0.16, 0.02, 0.01);
    drawCube1(0.2,0.1,0.1,  0.1, 0.05, 0.05);
    glPopMatrix();

    //cupboard's drawer's 2nd handle
    glPushMatrix();
    glTranslatef(4.5,0.25,5.9);
    glScalef(0.16, 0.02, 0.01);
    drawCube1(0.2,0.1,0.1,  0.1, 0.05, 0.05);
    glPopMatrix();
}

void room()
{
    // carpet
    //glColor3f(0.4, 0.1, 0.0);
    glPushMatrix();
    //glScalef(5, 0.1, 7);
    glTranslatef(3,-0.2,7);
    glScalef(1.3, 0.01, 1.7);
    drawCube1(0.4, 0.1, 0.0,  0.20, 0.05, 0.0);
```

```
    glPopMatrix();

    // right wall
    //glColor3f(1, 0.8, 0.5);
    glPushMatrix();
    glTranslatef(-1.5,-1,.5);
    glScalef(5, 2, 0.1);
    //drawCube1(1, 0.8, 0.5,  0.5,0.4,0.25);
    drawCube1(1, 0.8, 0.7,  0.5, 0.4, 0.35);
    glPopMatrix();

    // left wall
    //glColor3f(1, 0.8, 0.7);
    glPushMatrix();
    glTranslatef(-4.5,-1,0);
    glScalef(1, 2, 5);
    drawCube1(1, 0.8, 0.7,  0.5, 0.4, 0.35);
    glPopMatrix();

    // wall besides the right wall
    //glColor3f(1, 0.8, 0.7);
    glPushMatrix();
    glTranslatef(8,-1,0);
    glScalef(0.2, 2, 5);
    drawCube1(1, 0.8, 0.7,  0.5, 0.4, 0.35);
    glPopMatrix();

    //ceiling
     //glColor3f(1.0, 0.9, 0.8);
     glPushMatrix();
     glTranslatef(-2,5.1,0);
     glScalef(5, 0.1, 7);
     drawCube1(1.0, 0.9, 0.8,  0.5,0.45,0.4);
     glPopMatrix();

    // floor
    glPushMatrix();
    glScalef(5, 0.1, 7);
    glTranslatef(-1,-5,0); //-1,-5,.5
    //glScalef(5, 0.1, 7);
    drawCube1(0.5, 0.1, 0.0,  0.25,0.05,0);
    glPopMatrix();
}

void bed()
{
    //bed headboard
    glPushMatrix();
    glScalef(0.1, 0.5, 0.9);
    glTranslatef(-2,-0.5,6.2);
    drawCube1(0.5,0.2,0.2,  0.25,0.1,0.1);
    glPopMatrix();

    //bed body
```

```
    glPushMatrix();
    glScalef(1, 0.2, 0.9); //1, 0.2, 0.9
    glTranslatef(0,-0.5,6.2);
    drawCube1(0.824, 0.706, 0.549,   0.412,0.353,0.2745);
    glPopMatrix();

    //pillow right far
    //glColor3f(0.627, 0.322, 0.176);
    glPushMatrix();
    glTranslatef(0.5,0.5,6);
    glRotatef(20, 0,0,1);
    glScalef(0.1, 0.15, 0.28);
    drawCube1(0.627, 0.322, 0.176,  0.3135,0.161,0.088);
    glPopMatrix();

    //pillow left near
    //glColor3f(0.627, 0.322, 0.176);
    glPushMatrix();
    glTranslatef(0.5,0.5,7.2);
    glRotatef(22, 0,0,1);
    glScalef(0.1, 0.15, 0.28);
    drawCube1(0.627, 0.322, 0.176,  0.3135,0.161,0.088);
    glPopMatrix();

    //blanket
    //glColor3f(0.627, 0.322, 0.176);
    glPushMatrix();
    glTranslatef(1.4,0.45,5.5);
    //glRotatef(22, 0,0,1);
    glScalef(0.5, 0.05, 0.95);
    drawCube1(0.627, 0.322, 0.176,  0.3135,0.161,0.088);
    glPopMatrix();

    //blanket side left part
    //glColor3f(0.627, 0.322, 0.176);
    glPushMatrix();
    glTranslatef(1.4,-0.3,8.16);
    //glRotatef(22, 0,0,1);
    glScalef(0.5, 0.25, 0.05);
    drawCube1(0.627, 0.322, 0.176,  0.3135,0.161,0.088);
    glPopMatrix();
}

void bedsideDrawer()
{
    //bedside drawer ************************************

      //side drawer
      glPushMatrix();
      glTranslatef(0.5,-0.1,8.7); //0.5,-0.1,9
      glScalef(0.12, 0.2, 0.23);
      drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
      glPopMatrix();
```

```
      //side drawer's drawer
      glPushMatrix();
      glTranslatef(0.88,0,8.8);
      glScalef(0.0001, 0.11, 0.18);
      drawCube1(0.3,0.2,0.2, 0.15,0.1,0.1);
      glPopMatrix();

      //side drawer's knob
      glPushMatrix();
      glTranslatef(0.9,0.15,9.05);
      glScalef(0.01, 0.02, 0.02);
      drawSphere(0.3, 0.1, 0.0,  0.15,0.05,0.0);
      glPopMatrix();
}

void lamp()
{
    //lamp base
    glPushMatrix();
    glTranslatef(.6,0.5,8.95);
    glScalef(0.07, 0.02, 0.07);
    drawCube1(0,0,1,  0,0,0.5);
    glPopMatrix();

    //lamp stand
    glColor3f(1,0,0);
    glPushMatrix();
    glTranslatef(.7,0.35,9.05);
    glScalef(0.01, 0.2, 0.01);
    drawCube1(1,0,0,  0.5,0.0,0.0);
    glPopMatrix();

    //lamp shade
    glColor3f(0.000, 0.000, 0.545);
    glPushMatrix();
    glTranslatef(.6,0.9,8.9);
    glScalef(0.08, 0.09, 0.08);
    drawTrapezoid(0.000, 0.000, 0.545,  0,0,0.2725);
    //drawCube1(0.000, 0.000, 0.545,  0,0,0.2725);
    glPopMatrix();
}

void LinkinParkPoster()
{
    //Linkin Park Poster ****************************************

       //poster black
       glColor3f(0.0,0.0,0.0);
       glPushMatrix();
       glTranslatef(-1,1.4,4.6);
       glScalef(0.0001, .65, .8);
       drawCube1(0,0,0, 0,0,0, 10);
       glPopMatrix();
```

```
        //Linkin Park logo
        glColor3f(1.0,1.0,1.0);
        glPushMatrix();
        glTranslatef(-0.9,2.1,5.5);
        //glRotatef(22, 0,0,1);
        glScalef(0.0001, .02, .25);
    drawCube1(1.0,1.0,1.0,  1.0,1.0,1.0, 10);
        glPopMatrix();

        //Linkin Park logo
        glColor3f(1.0,1.0,1.0);
        glPushMatrix();
        glTranslatef(-0.9,2.1,6.2);
        glRotatef(-14, 1,0,0);
        glScalef(0.0001, .28, .02);
    drawCube1(1.0,1.0,1.0,  1.0,1.0,1.0, 10);
        glPopMatrix();

        //Linkin Park logo
        glColor3f(1.0,1.0,1.0);
        glPushMatrix();
        glTranslatef(-0.9,1.8,6);
        glRotatef(-14, 1,0,0);
        glScalef(0.0001, .29, .02);
    drawCube1(1.0,1.0,1.0,  1.0,1.0,1.0, 10);
        glPopMatrix();

        //Linkin Park logo
        glColor3f(1.0,1.0,1.0);
        glPushMatrix();
        glTranslatef(-0.9,2.1,5.5);
        glRotatef(23, 1,0,0);
        glScalef(0.0001, .25, .02);
        drawCube1(1.0,1.0,1.0,  1.0,1.0,1.0, 10);
        glPopMatrix();
}

void wardrobe()
{
        //wardrobe
        glPushMatrix();
        glTranslatef(0,0,4);
        glScalef(0.12, 0.6, 0.4);
        drawCube1(0.3,0.1,0, 0.15,0.05,0);
        glPopMatrix();

        //wardrobe's 1st drawer
        glPushMatrix();
        glTranslatef(0.36,1.4,4.05);
        //glRotatef(22, 0,0,1);
        glScalef(0.0001, 0.11, 0.38);
        drawCube1(0.5,0.2,0.2, 0.25,0.1,0.1);
        glPopMatrix();
```

```
   //wardrobe's 2nd drawer
   glPushMatrix();
   glTranslatef(0.36,1,4.05);
   //glRotatef(22, 0,0,1);
   glScalef(0.0001, 0.11, 0.38);
drawCube1(0.5,0.2,0.2, 0.25,0.1,0.1);
   glPopMatrix();

   //wardrobe's 3rd drawer
   glPushMatrix();
   glTranslatef(0.36,0.6,4.05);
   //glRotatef(22, 0,0,1);
   glScalef(0.0001, 0.11, 0.38);
drawCube1(0.5,0.2,0.2, 0.25,0.1,0.1);
   glPopMatrix();

   //wardrobe's 4th drawer
   glPushMatrix();
   glTranslatef(0.36,0.2,4.05);
   //glRotatef(22, 0,0,1);
   glScalef(0.0001, 0.11, 0.38);
drawCube1(0.5,0.2,0.2, 0.25,0.1,0.1);
   glPopMatrix();

   //wardrobe's 1st drawer handle
   glColor3f(0.3,0.1,0);
   glPushMatrix();
   glTranslatef(0.37,1.5,4.3);
   //glRotatef(22, 0,0,1);
   glScalef(0.01, 0.03, 0.2);
   drawCube1(0.3,0.1,0, 0.15,0.05,0.0);
   glPopMatrix();

   //wardrobe's 2nd drawer handle
   glColor3f(0.3,0.1,0);
   glPushMatrix();
   glTranslatef(0.37,1.1,4.3);
   //glRotatef(22, 0,0,1);
   glScalef(0.01, 0.03, 0.2);
drawCube1(0.3,0.1,0, 0.15,0.05,0.0);
glPopMatrix();

   //wardrobe's 3rd drawer handle
   glColor3f(0.3,0.1,0);
   glPushMatrix();
   glTranslatef(0.37,0.7,4.3);
   //glRotatef(22, 0,0,1);
   glScalef(0.01, 0.03, 0.2);
drawCube1(0.3,0.1,0, 0.15,0.05,0.0);
glPopMatrix();

   //wardrobe's 4th drawer handle
   glColor3f(0.3,0.1,0);
   glPushMatrix();
```

```
       glTranslatef(0.37,0.3,4.3);
       //glRotatef(22, 0,0,1);
       glScalef(0.01, 0.03, 0.2);
    drawCube1(0.3,0.1,0, 0.15,0.05,0.0);
    glPopMatrix();

}

void dressingTable()
{
    //Dressing table ************************************************

        //dressing table left body
        glPushMatrix();
        glTranslatef(5.9,0,4.6);
        glScalef(0.2, 0.2, 0.2);
        drawCube1(0.545, 0.271, 0.075,  0.2725,0.1355,0.0375);
        glPopMatrix();

     /*   //dressing table left body left stripe
        glColor3f(0.2,0.1,0.1);
        glPushMatrix();
        glTranslatef(5.9,0,5.2);
        //glRotatef(22, 0,0,1);
        glScalef(0.01, 0.3, 0.0001);
        drawCube();
        glPopMatrix();

        //dressing table left body right stripe
        glColor3f(0.2,0.1,0.1);
        glPushMatrix();
        glTranslatef(6.5,0,5.2);
        //glRotatef(22, 0,0,1);
        glScalef(0.01, 0.2, 0.0001);
        drawCube();
        glPopMatrix();

        //dressing table left body bottom stripe
        glColor3f(0.2,0.1,0.1);
        glPushMatrix();
        glTranslatef(5.9,0,5.2);
        //glRotatef(22, 0,0,1);
        glScalef(0.2, 0.01, 0.0001);
        drawCube();
        glPopMatrix();  */

        //dressing table right body
        glPushMatrix();
        glTranslatef(7,0,4.6);
        glScalef(0.2, 0.2, 0.2);
    drawCube1(0.545, 0.271, 0.075,  0.2725,0.1355,0.0375);
        glPopMatrix();

   /*     //dressing table right body left stripe
```

```
    glColor3f(0.2,0.1,0.1);
    glPushMatrix();
    glTranslatef(7,0,5.2);
    //glRotatef(22, 0,0,1);
    glScalef(0.01, 0.2, 0.0001);
    drawCube();
    glPopMatrix();

    //dressing table right body right stripe
    glColor3f(0.2,0.1,0.1);
    glPushMatrix();
    glTranslatef(7.6,0,5.2);
    //glRotatef(22, 0,0,1);
    glScalef(0.01, 0.3, 0.0001);
    drawCube();
    glPopMatrix();

    //dressing table right body bottom stripe
    glColor3f(0.2,0.1,0.1);
    glPushMatrix();
    glTranslatef(7,0,5.2);
    //glRotatef(22, 0,0,1);
    glScalef(0.2, 0.01, 0.0001);
    drawCube();
    glPopMatrix(); */

    //dressing table upper body
    glPushMatrix();
    glTranslatef(5.9,0.6,4.6);
    glScalef(0.57, 0.1, 0.2);
drawCube1(0.545, 0.271, 0.075,  0.2725,0.1355,0.0375);
    glPopMatrix();

    //dressing table upper body bottom stripe
    glPushMatrix();
    glTranslatef(5.9,0.6,5.2);
    //glRotatef(22, 0,0,1);
    glScalef(0.57, 0.01, 0.0001);
drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
    glPopMatrix();

    //dressing table upper body upper stripe
    glPushMatrix();
    glTranslatef(5.9,0.9,5.2);
    //glRotatef(22, 0,0,1);
    glScalef(0.57, 0.01, 0.0001);
drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
    glPopMatrix();

    //dressing table upper body handle
    glColor3f(0.2,0.1,0.1);
    glPushMatrix();
    glTranslatef(6.5,0.75,5.2);
    //glRotatef(22, 0,0,1);
```

```
    glScalef(0.16, 0.02, 0.0001);
    drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
    glPopMatrix();

    //dressing table left body handle
    glColor3f(0.2,0.1,0.1);
    glPushMatrix();
    glTranslatef(6.4,0.1,5.2);
    //glRotatef(22, 0,0,1);
    glScalef(0.02, 0.13, 0.0001);
drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
    glPopMatrix();

    //dressing table right body handle
    glColor3f(0.2,0.1,0.1);
    glPushMatrix();
    glTranslatef(7.1,0.1,5.2);
    //glRotatef(22, 0,0,1);
    glScalef(0.02, 0.13, 0.0001);
drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
    glPopMatrix();

    //dressing table main mirror
    glPushMatrix();
    glTranslatef(6.2,0.9,4.7);
    //glRotatef(22, 0,0,1);
    glScalef(0.36, 0.5, 0.0001);
    drawCube1(0.690, 0.878, 0.902,  0.345,0.439,0.451, 10);
    glPopMatrix();

    //dressing table left mirror
    glPushMatrix();
    glTranslatef(5.92,0.9,4.7);
    //glRotatef(0, 0,1,0);
    glScalef(0.1, 0.48, 0.0001);
drawCube1(0.690, 0.878, 0.902,  0.345,0.439,0.451, 10);
    glPopMatrix();

    //dressing table left mirror left stripe
    glColor3f(0.2,0.1,0.1);
    glPushMatrix();
    glTranslatef(5.92,0.9,4.71);
    //glRotatef(22, 0,0,1);
    glScalef(0.019, 0.48, 0.0001);
drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
    glPopMatrix();

    //dressing table left mirror left stripe
    glColor3f(0.2,0.1,0.1);
    glPushMatrix();
    glTranslatef(6.17,0.9,4.71);
    //glRotatef(22, 0,0,1);
    glScalef(0.019, 0.48, 0.0001);
drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
```

```
    glPopMatrix();

    //dressing table mirror  stripe
    glColor3f(0.2,0.1,0.1);
    glPushMatrix();
    glTranslatef(5.92,0.9,4.71);
    //glRotatef(22, 0,0,1);
    glScalef(0.55, 0.019, 0.0001);
drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
    glPopMatrix();

    //dressing table left mirror upper stripe
    glColor3f(0.2,0.1,0.1);
    glPushMatrix();
    glTranslatef(5.92,2.3,4.71);
    //glRotatef(22, 0,0,1);
    glScalef(0.1, 0.019, 0.0001);
drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
    glPopMatrix();

    //dressing table right mirror
    glPushMatrix();
    glTranslatef(7.25,0.9,4.7);
    //glRotatef(-40, 0,1,0);
    glScalef(0.1, 0.48, 0.0001);
drawCube1(0.690, 0.878, 0.902,  0.345,0.439,0.451, 10);
    glPopMatrix();

    //dressing table left mirror upper stripe
    glColor3f(0.2,0.1,0.1);
    glPushMatrix();
    glTranslatef(7.25,2.3,4.71);
    //glRotatef(22, 0,0,1);
    glScalef(0.1, 0.019, 0.0001);
drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
    glPopMatrix();

    //dressing table right mirror left stripe
    glColor3f(0.2,0.1,0.1);
    glPushMatrix();
    glTranslatef(7.25,0.9,4.71);
    //glRotatef(22, 0,0,1);
    glScalef(0.019, 0.48, 0.0001);
drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
    glPopMatrix();

    //dressing table right mirror right stripe
    glColor3f(0.2,0.1,0.1);
    glPushMatrix();
    glTranslatef(7.5,0.9,4.71);
    //glRotatef(22, 0,0,1);
    glScalef(0.019, 0.48, 0.0001);
drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
    glPopMatrix();
```

```
        //dressing table main mirror polygon part
        glColor3f(0.690, 0.878, 0.902);
        glPushMatrix();
        glTranslatef(6.2,2.4,4.7);
        //glRotatef(22, 0,0,1);
        glScalef(0.18, 0.18, 2);
        polygon(0.690, 0.878, 0.902,  0.345,0.439,0.451, 10);
        glPopMatrix();

        //dressing table upper round srtipe
        glColor3f(0.2,0.1,0.1);
        glPushMatrix();
        glTranslatef(6.2,2.4,4.71);
        glScalef(.18, .18, 1);
        polygonLine(0.2,0.1,0.1,  0.1,0.05,0.05, 50);
        glPopMatrix();

}

void wallshelf()
{
    //Wall Shelf ********************************************

      //wall shelf one
      glPushMatrix();
      glTranslatef(1.5,2.7,3);
      glScalef(0.4, 0.03, 0.2);
    drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
      glPopMatrix();

      //wall shelf two
      glPushMatrix();
      glTranslatef(1,2.3,3);
      glScalef(0.4, 0.03, 0.2);
    drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
      glPopMatrix();

      //wall shelf three
      glPushMatrix();
      glTranslatef(0.5,1.9,3);
      glScalef(0.4, 0.03, 0.2);
    drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
      glPopMatrix();

      //wall shelf four
    glPushMatrix();
      glTranslatef(1,1.5,3);
      glScalef(0.4, 0.03, 0.2);
    drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
      glPopMatrix();

      //wall shelf five
      glPushMatrix();
```

```
  glTranslatef(1.5,1.1,3);
  glScalef(0.4, 0.03, 0.2);
drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
glPopMatrix();

  //showpiece on the bottom shelf from left 1
  glPushMatrix();
  glTranslatef(1.5,1.2,3);
  glScalef(0.04, 0.06, 0.2);
  drawCube1(0.698,0.133,0.133,  0.349,0.0665,0.0665);
  glPopMatrix();

  //showpiece on the bottom shelf from left 2
  glPushMatrix();
  glTranslatef(2,1.2,3);
  glScalef(0.04, 0.06, 0.2);
  drawCube1(0.729,0.333,0.827,  0.3645,0.1665,0.4135);
  glPopMatrix();

  //showpiece on the bottom shelf from left 3 lower portion
  glPushMatrix();
  glTranslatef(2.5,1.2,3);
  glScalef(0.04, 0.06, 0.2);
  drawCube1(0.098,0.098,0.439,  0.049,0.049,0.2195);
  glPopMatrix();

  //showpiece on the bottom shelf from left 3 upper portion
  glPushMatrix();
  glTranslatef(2.51,1.35,3);
  glScalef(0.01, 0.05, 0.2);
  drawCube1(0.529,0.808,0.980,  0.2645,0.404,0.490);
  glPopMatrix();

  //showpiece on the top shelf  left 2
  glColor3f(0.502, 0.502, 0.000);
  glPushMatrix();
  glTranslatef(2.5,2.71,3);
  //glRotatef(22, 0,0,1);
  glScalef(0.05, 0.16, 0.01);
  drawCube1(0.502,0.502,0.000,  0.251,0.251,0);
  glPopMatrix();

  //showpiece on the top shelf left 1
  glPushMatrix();
  glTranslatef(1.8,2.71,3);
  glScalef(0.16, 0.1, 0.01);
  drawCube1(0,0,0.9,  0,0,0.45);
  glPopMatrix();

  //showpiece on 2nd shelf
  glColor3f(.416, 0.353, 0.804);
  glPushMatrix();
  glTranslatef(1.3,2.4,3);
  glScalef(0.16, 0.08, 0.01);
```

```
      drawCube1(.416,0.353,0.804,  0.208,0.1765,0.402);
      glPopMatrix();

      //showpiece on 3rd shelf left 1
      glPushMatrix();
      glTranslatef(0.4,1.9,3);
      glScalef(0.05, 0.16, 0.01);
      drawCube1(0.863,0.078,0.235,  0.4315,0.039,0.1175);
      glPopMatrix();

      //showpiece on 3rd shelf left 2
      glPushMatrix();
      glTranslatef(0.7,1.9,3);
      glScalef(0.05, 0.12, 0.01);
      drawCube1(0.780,0.082,0.522,  0.39,0.041,0.261);
      glPopMatrix();

      //showpiece on 3rd shelf left 3
      glColor3f(0.600, 0.196, 0.800);
      glPushMatrix();
      glTranslatef(1,1.9,3);
      glScalef(0.05, 0.09, 0.01);
      drawCube1(0.6,0.196,0.8,  0.3,0.098,0.4);
      glPopMatrix();

      //showpiece on 4th shelf
      glPushMatrix();
      glTranslatef(1.8,1.5,3);
      glScalef(0.2, 0.1, 0.2);
      drawpyramid(0.282,0.239,0.545,  0.141,0.1195,0.2725, 50);
      glPopMatrix();

      //showpiece on 4th shelf
      glPushMatrix();
      glTranslatef(1.4,1.5,3);
      glScalef(0.15, 0.1, 0.2);
      drawpyramid(0.251,0.878,0.816,  0.1255,0.439,0.408, 50);
      glPopMatrix();

}

void Clock()
{
    //Clock ************************************************************

        //clock body
        glColor3f(0.545, 0.271, 0.075);
        glPushMatrix();
        glTranslatef(-0.9,1.8,7.87);
        //glRotatef(22, 0,0,1);
        glScalef(0.08, 0.25, 0.1);
        drawCube1(0.545,0.271,0.075,  0.271,0.1335,0.0375,  50);
        glPopMatrix();
```

```
    //clock body white
    glPushMatrix();
    glTranslatef(-0.83,1.9,7.9);
    //glRotatef(22, 0,0,1);
    glScalef(0.06, 0.2, 0.08);
    drawCube1(1.000,0.894,0.710,  1.000,0.894,0.710);
    glPopMatrix();

    //clock hour handle
    glPushMatrix();
    glTranslatef(-0.65,2.18,8.01);
    glRotatef(45, 1,0,0);
    glScalef(0.0001, 0.01, 0.04);
    drawCube1(0,0,0,  0,0,0);
    glPopMatrix();

   //clock minute handle
    glPushMatrix();
    glTranslatef(-0.65,2.18,8.01);
    glRotatef(90, 1,0,0);
    glScalef(0.0001, 0.012, 0.08);
drawCube1(0,0,0,  0,0,0);
    glPopMatrix();

 /*   //clock body bottom strip
    glColor3f(0.2,0.1,0.1); //0.2,0.1,0.1
    glPushMatrix();
    glTranslatef(-0.66,1.8,7.89);
    //glRotatef(22, 0,0,1);
    glScalef(0.001, 0.01, 0.1);
    drawCube();
    glPopMatrix();

    //clock body right strip
    glColor3f(0.0,0.0,0.0); //0.2,0.1,0.1
    glPushMatrix();
    glTranslatef(-0.66,1.8,7.89);
    //glRotatef(22, 0,0,1);
    glScalef(0.005, 0.25, 0.01);
    drawCube();
    glPopMatrix();

    //clock body left strip
    glColor3f(0.2,0.1,0.1); //0.2,0.1,0.1
    glPushMatrix();
    glTranslatef(-0.65,1.8,8.2);
    //glRotatef(22, 0,0,1);
    glScalef(0.0001, 0.25, 0.01);
    drawCube();
    glPopMatrix();  */

    //clock pendulum stick
    glColor3f(0.2,0.1,0.1); //0.2,0.1,0.1
    glPushMatrix();
```

```
        glTranslatef(-0.7,2,8.1);
        glRotatef(theta, 1,0,0);
        glScalef(0.0001, 0.2, 0.03);
        drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
        glPopMatrix();

        //clock pendulum ball
        glColor3f(0.2,0.1,0.1); //0.2,0.1,0.1
        glPushMatrix();
        glTranslatef(-0.72,1.42,z);
        //glRotatef(x, 1,0,0);
        glScalef(0.035, 0.035, 0.035);
        //drawCube();
    drawSphere(0.2,0.1,0.1, 0.1,0.05,0.05, 10);
        glPopMatrix();

    //clock top pyramid
        glPushMatrix();
        glTranslatef(-0.9,2.5,7.81);
        //glRotatef(x, 1,0,0);
        glScalef(0.16, 0.1, 0.2);
        drawpyramid(0.5,0.2,0,  0.25,0.1,0,  50);
        glPopMatrix();
}

void window()
{
    //Window ******************************************
    //window white open
    glPushMatrix();
    glTranslatef(-0.9,1,8.9); //0.5,1,9.6
    glScalef(0.0001, .6, .3);
    drawCube1(1.0,1.0,1.0, 0.05,0.05,0.05);
    glPopMatrix();

    //window right side corner
    glPushMatrix();
    glTranslatef(-0.9,1,8.9);
    glScalef(0.04, 0.6, 0.0001);
    drawCube1(0.8,0.6,0.4, 0.4,0.3,0.2);
    glPopMatrix();

    //window left side corner
    glPushMatrix();
    glTranslatef(-0.9,1,9.8);
    glScalef(0.04, 0.6, 0.0001);
    drawCube1(0.8,0.6,0.4, 0.4,0.3,0.2);
    glPopMatrix();

    //window upper side corner
    glPushMatrix();
    glTranslatef(-0.7,2.7,8.9);
    glScalef(0.0001, 0.05, 0.4);
    drawCube1(0.7,0.6,0.5,  0.35,0.3,0.25);
```

```
    glPopMatrix();


    //window lower side corner
    glPushMatrix();
    glTranslatef(-0.8,1.02,8.9);
    glScalef(0.0001, 0.02, 0.34);
    drawCube1(0.7,0.6,0.5,  0.35,0.3,0.25);
    glPopMatrix();

        //window vertical bar 1
        glPushMatrix();
        glTranslatef(-0.87,2.1,8.9);
        //glRotatef(22, 0,0,1);
        glScalef(0.0001, 0.02, 0.3);
        drawCube1(0.0,0.0,0.0, 0.0,0.0,0.0, 5);
        glPopMatrix();

        //window vertical bar 2
        glPushMatrix();
        glTranslatef(-0.87,1.6,8.9);
        //glRotatef(22, 0,0,1);
        glScalef(0.0001, 0.02, 0.3);
    drawCube1(0.0,0.0,0.0, 0.0,0.0,0.0, 5);
        glPopMatrix();

        //window horizontal bar
        glPushMatrix();
        glTranslatef(-0.87,1,9.3);
        //glRotatef(22, 0,0,1);
        glScalef(0.0001, 0.6, 0.02);
    drawCube1(0.0,0.0,0.0, 0.0,0.0,0.0, 5);
        glPopMatrix();
}

void sphericalObject()
{
    //table top part
    glPushMatrix();
    glTranslatef(5, 0.2, 10);
    glScalef(0.1, 0.02, 0.1);
    drawSphere(0.5,0.2,0,  0.25,0.1,0,  20);
    glPopMatrix();

    //table leg
    glColor3f(0.2,0.1,0.1); //0.2,0.1,0.1
    glPushMatrix();
    glTranslatef(4.98,-0.1,10);
    glScalef(0.02, 0.1, 0.02);
    drawCube1(0.2,0.1,0.1,  0.1,0.05,0.05);
    glPopMatrix();

    //base
    glPushMatrix();
```

```
    glTranslatef(5, -0.1, 10);
    glScalef(0.05, 0.01, 0.05);
    drawSphere(0.5,0.2,0,  0.25,0.1,0,  20);
    glPopMatrix();
}


void lightBulb1()
{
    GLfloat no_mat[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat mat_ambient[] = { 0.7, 0.7, 0.7, 1.0 };
    GLfloat mat_ambient_color[] = { 0.8, 0.8, 0.2, 1.0 };
    GLfloat mat_diffuse[] = { 1.000, 0.843, 0.000, 1.0 };
    GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat high_shininess[] = { 100.0 };
    GLfloat mat_emission[] = {1.000, 1,1, 0.0};

    glPushMatrix();
    glTranslatef (5, 5, 8);
    glScalef(0.2, 0.2, 0.2);
    glMaterialfv(GL_FRONT, GL_AMBIENT, no_mat);
    glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
    glMaterialfv(GL_FRONT, GL_SPECULAR, no_mat);
    glMaterialfv(GL_FRONT, GL_SHININESS, high_shininess);

    if(switchOne == true){
        glMaterialfv(GL_FRONT, GL_EMISSION, mat_emission);
    }else{
        glMaterialfv(GL_FRONT, GL_EMISSION, no_mat);
    }

    glutSolidSphere(1.0, 16, 16);
    glPopMatrix();
}

void lightBulb2()
{
    GLfloat no_mat[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat mat_ambient[] = { 0.7, 0.7, 0.7, 1.0 };
    GLfloat mat_ambient_color[] = { 0.8, 0.8, 0.2, 1.0 };
    GLfloat mat_diffuse[] = { 1.000, 0.843, 0.000, 1.0 };
    GLfloat high_shininess[] = { 100.0 };
    GLfloat mat_emission[] = {1,1,1, 1.0};

    glPushMatrix();
    glTranslatef (0,5,8);
    glScalef(0.2, 0.2, 0.2);
    glMaterialfv(GL_FRONT, GL_AMBIENT, no_mat);
    glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
    glMaterialfv(GL_FRONT, GL_SPECULAR, no_mat);
    glMaterialfv(GL_FRONT, GL_SHININESS, high_shininess);
    if(switchTwo == true){
        glMaterialfv(GL_FRONT, GL_EMISSION, mat_emission);
    }else{
```

```
        glMaterialfv(GL_FRONT, GL_EMISSION, no_mat);
    }
    glutSolidSphere(1.0, 16, 16);
    glPopMatrix();
}

void lightBulb3()
{
    GLfloat no_mat[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat mat_ambient[] = { 0.7, 0.7, 0.7, 1.0 };
    GLfloat mat_ambient_color[] = { 0.8, 0.8, 0.2, 1.0 };
    GLfloat mat_diffuse[] = { 1.000, 0.843, 0.000, 1.0 };
    GLfloat high_shininess[] = { 100.0 };
    GLfloat mat_emission[] = {1,1,1, 1.0};

    glPushMatrix();
    glTranslatef (0.7, 1.5, 9.0);
    glScalef(0.2, 0.2, 0.2);
    glMaterialfv(GL_FRONT, GL_AMBIENT, no_mat);
    glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
    glMaterialfv(GL_FRONT, GL_SPECULAR, no_mat);
    glMaterialfv(GL_FRONT, GL_SHININESS, high_shininess);
    if(switchLamp == true){
        glMaterialfv(GL_FRONT, GL_EMISSION, mat_emission);
    }else{
        glMaterialfv(GL_FRONT, GL_EMISSION, no_mat);
    }
    glutSolidSphere(1.0, 16, 16);
    glPopMatrix();
}


void lightOne()
{
    glPushMatrix();
    GLfloat no_light[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat light_ambient[]  = {0.5, 0.5, 0.5, 1.0};
    GLfloat light_diffuse[]  = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat light_specular[] = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat light_position[] = { 5.0, 5.0, 8.0, 1.0 }; //5 5 10

    //glEnable( GL_LIGHT0);

    if(amb1 == true){glLightfv( GL_LIGHT0, GL_AMBIENT, light_ambient);}
    else{glLightfv( GL_LIGHT0, GL_AMBIENT, no_light);}

    if(diff1 == true){glLightfv( GL_LIGHT0, GL_DIFFUSE, light_diffuse);}
    else{glLightfv( GL_LIGHT0, GL_DIFFUSE, no_light);}

    if(spec1 == true){glLightfv( GL_LIGHT0, GL_SPECULAR,
light_specular);}
    else{glLightfv( GL_LIGHT0, GL_SPECULAR, no_light);}

    glLightfv( GL_LIGHT0, GL_POSITION, light_position);
```

```
    glPopMatrix();
}

void lightTwo()
{
    glPushMatrix();
    GLfloat no_light[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat light_ambient[]  = {0.5, 0.5, 0.5, 1.0};
    GLfloat light_diffuse[]  = { 1.0, 1.0, 0.9, 1.0 };
    GLfloat light_specular[] = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat light_position[] = { 0.0, 5.0, 8.0, 1.0 };

    //glEnable( GL_LIGHT1);

    if(amb2 == true){glLightfv( GL_LIGHT1, GL_AMBIENT, light_ambient);}
    else{glLightfv( GL_LIGHT1, GL_AMBIENT, no_light);}

    if(diff2 == true){glLightfv( GL_LIGHT1, GL_DIFFUSE, light_diffuse);}
    else{glLightfv( GL_LIGHT1, GL_DIFFUSE, no_light);}

    if(spec2 == true){glLightfv( GL_LIGHT1, GL_SPECULAR,
light_specular);}
    else{glLightfv( GL_LIGHT1, GL_SPECULAR, no_light);}

    glLightfv( GL_LIGHT1, GL_POSITION, light_position);
    glPopMatrix();
}

void lampLight()
{
    glPushMatrix();
    GLfloat no_light[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat light_ambient[]  = {0.5, 0.5, 0.5, 1.0};
    GLfloat light_diffuse[]  = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat light_specular[] = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat light_position[] = { 0.7, 1.5, 9.0, 1.0 };  //0.7, 4.5, 9.0

    //glEnable( GL_LIGHT2);

    if(amb3 == true){glLightfv( GL_LIGHT2, GL_AMBIENT, light_ambient);}
    else{glLightfv( GL_LIGHT2, GL_AMBIENT, no_light);}

    if(diff3 == true){glLightfv( GL_LIGHT2, GL_DIFFUSE, light_diffuse);}
    else{glLightfv( GL_LIGHT2, GL_DIFFUSE, no_light);}

    if(spec3 == true){glLightfv( GL_LIGHT2, GL_SPECULAR,
light_specular);}
    else{glLightfv( GL_LIGHT2, GL_SPECULAR, no_light);}

    glLightfv( GL_LIGHT2, GL_POSITION, light_position);
    GLfloat spot_direction[] = { 0.3, -1, -0.8 };
    glLightfv(GL_LIGHT2, GL_SPOT_DIRECTION, spot_direction);
    glLightf( GL_LIGHT2, GL_SPOT_CUTOFF, 35.0);
    glPopMatrix();
```

```
}



void display(void)
{
    glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);

    glMatrixMode( GL_PROJECTION );
    glLoadIdentity();
    gluPerspective(60,1,1,100);

    glMatrixMode( GL_MODELVIEW );
    glLoadIdentity();
    gluLookAt(eyeX,eyeY,eyeZ,  refX,refY,refZ,  0,1,0); //7,2,15, 0,0,0,
0,1,0

    glEnable(GL_LIGHTING);
    lightOne();
    lightTwo();
    lampLight();
    room();
    bed();
    bedsideDrawer();
    lamp();
    LinkinParkPoster();
    wallshelf();
    wardrobe();
    cupboard();
    dressingTable();
    Clock();
    window();
    sphericalObject();
    lightBulb1();
    lightBulb2();
    //lightBulb3();
    glDisable(GL_LIGHTING);

    glFlush();
    glutSwapBuffers();
}

void myKeyboardFunc( unsigned char key, int x, int y )
{
    switch ( key )
    {
        case 'w': // move eye point upwards along Y axis
            eyeY+=1.0;
            break;
        case 's': // move eye point downwards along Y axis
            eyeY-=1.0;
            break;
        case 'a': // move eye point left along X axis
            eyeX-=1.0;
```

```
            break;
        case 'd': // move eye point right along X axis
            eyeX+=1.0;
            break;
        case 'o':  //zoom out
            eyeZ+=1;
            break;
        case 'i': //zoom in
            eyeZ-=1;
            break;
        case 'q': //back to default eye point and ref point
                eyeX=7.0; eyeY=2.0; eyeZ=15.0;
                refX=0.0; refY=0.0; refZ=0.0;
            break;
        case 'j': // move ref point upwards along Y axis
            refY+=1.0;
            break;
        case 'n': // move ref point downwards along Y axis
            refY-=1.0;
            break;
        case 'b': // move ref point left along X axis
            refX-=1.0;
            break;
        case 'm': // move eye point right along X axis
            refX+=1.0;
            break;
        case 'k':  //move ref point away from screen/ along z axis
            refZ+=1;
            break;
        case 'l': //move ref point towards screen/ along z axis
            refZ-=1;
            break;
        case '1': //to turn on and off light one
            if(switchOne == false)
            {
                switchOne = true; amb1=true; diff1=true; spec1=true;
                glEnable( GL_LIGHT0); break;
            }
            else if(switchOne == true)
            {
                switchOne = false; amb1=false; diff1=false; spec1=false;
glDisable( GL_LIGHT0); break;
            }
        case '2': //to turn on and off light two
            if(switchTwo == false)
            {
                switchTwo = true; amb2=true; diff2=true; spec2=true;
                glEnable( GL_LIGHT1); break;
            }
            else if(switchTwo == true)
            {
                switchTwo = false; amb2=false; diff2=false; spec2=false;
                glDisable( GL_LIGHT1); break;
            }
```

```
        case '3': //to turn on and off light one
            if(switchLamp == false)
            {
                switchLamp = true; amb3=true; diff3=true; spec3=true;
                glEnable( GL_LIGHT2); break;
            }
            else if(switchLamp == true)
            {
                switchLamp = false; amb3=false; diff3=false; spec3=false;
                glDisable( GL_LIGHT2); break;
            }
        case'4': //turn on/off ambient light 1
            if(amb1 == false) {amb1=true; break;}
            else{amb1=false; break;}
        case'5':
            if(diff1 == false) {diff1=true; break;}
            else{diff1=false; break;}
        case'6':
            if(spec1 == false) {spec1=true; break;}
            else{spec1=false; break;}
        case'7': //turn on/off ambient light 2
            if(amb2 == false) {amb2=true; break;}
            else{amb2=false; break;}
        case'8':
            if(diff2 == false) {diff2=true; break;}
            else{diff2=false; break;}
        case'9':
            if(spec2 == false) {spec2=true; break;}
            else{spec2=false; break;}
        case'e': //turn on/off ambient lamp light
            if(amb3 == false) {amb3=true; break;}
            else{amb3=false; break;}
        case'r':
            if(diff3 == false) {diff3=true; break;}
            else{diff3=false; break;}
        case't':
            if(spec3 == false) {spec3=true; break;}
            else{spec3=false; break;}
        case 27:    // Escape key
            exit(1);
    }

    glutPostRedisplay();
}


void animate()
{
    if(redFlag == true)
    {
        theta+=2;
        z-=0.02; //0.016667;
        if(theta >= 196 && theta <= 210)
        {
```

```
            y = 1.44;
        }
        else if(theta >= 180 && theta <= 194)
        {
            y = 1.42;
        }
        else if(theta >= 180 && theta <= 194)
        {
            y = 1.4;
        }
        else if(theta >= 164 && theta <= 178)
        {
            y = 1.42;
        }

        if(theta == 210)
        {
            redFlag = false;
        }
    }
    else if(redFlag == false)
    {
        theta-=2;
        z+=0.02;//0.016667;

        if(theta >= 196 && theta <= 210)
        {
            y = 1.44;
        }
        else if(theta >= 180 && theta <= 194)
        {
            y = 1.42;
        }
        else if(theta >= 180 && theta <= 194)
        {
            y = 1.4;
        }
        else if(theta >= 164 && theta <= 178)
        {
            y = 1.42;
        }

        if(theta == 150)
        {
            redFlag = true;
        }
    }

    glutPostRedisplay();

}

void fullScreen(int w, int h)
{
```

```
    //Prevent a divide by zero, when window is too short;you cant make a
window of zero width.
    if (h == 0)
        h = 1;
    float ratio = (GLfloat)w / (GLfloat)h;         //Calculate aspect
ratio of the window

    //Set the perspective coordinate system
    glMatrixMode(GL_PROJECTION);                   //Use the Projection
Matrix
    glLoadIdentity();                              //Reset Matrix

    glViewport(0, 0, w, h);                        //Set the viewport to
be the entire window
    gluPerspective(60, ratio, 1, 500);             //Set the correct
perspective.
    //glFrustum(-2.5,2.5,-2.5,2.5, ratio, 200);
    glMatrixMode(GL_MODELVIEW);                    //Get Back to the
Modelview
}


int main (int argc, char **argv)
{
    glutInit(&argc, argv);

    std::cout<<"To move Eye point:"<< std::endl;
    std::cout<<"w: up"<<std::endl;
    std::cout<<"s: down"<<std::endl;
    std::cout<<"a: left"<<std::endl;
    std::cout<<"d: right"<<std::endl;
    std::cout<<"i: zoom in"<<std::endl;
    std::cout<<"o: zoom out"<<std::endl;
    std::cout<<"      "<<std::endl;
    std::cout<<"To move Camera point:"<< std::endl;
    std::cout<<"j: up"<<std::endl;
    std::cout<<"n: down"<<std::endl;
    std::cout<<"b: left"<<std::endl;
    std::cout<<"m: right"<<std::endl;
    std::cout<<"l: move nearer"<<std::endl;
    std::cout<<"k: move far"<<std::endl;
    std::cout<<"      "<<std::endl;
    std::cout<<"Press q to move to default position"<<std::endl;
    std::cout<<"      "<<std::endl;
    std::cout<<"For lighting:      "<<std::endl;
    std::cout<<"Light source 1 [the light on the right on the screen
"<<std::endl;
    std::cout<<"1: to turn on/off light one     "<<std::endl;
    std::cout<<"4: to turn on/off ambient light one     "<<std::endl;
    std::cout<<"5: to turn on/off diffusion light one     "<<std::endl;
    std::cout<<"6: to turn on/off specular light one      "<<std::endl;
    std::cout<<"      "<<std::endl;
    std::cout<<"Light source 2 [the light on the left on the screen
"<<std::endl;
```

```
    std::cout<<"2: to turn on/off light two     "<<std::endl;
    std::cout<<"7: to turn on/off ambient light two     "<<std::endl;
    std::cout<<"8: to turn on/off diffusion light two     "<<std::endl;
    std::cout<<"9: to turn on/off specular light two      "<<std::endl;
    std::cout<<"      "<<std::endl;
    std::cout<<"Lamp light (spot light)" <<std::endl;
    std::cout<<"3: to turn on/off lamp     "<<std::endl;
    std::cout<<"e: to turn on/off ambient lamp light    "<<std::endl;
    std::cout<<"r: to turn on/off diffusion lamp light      "<<std::endl;
    std::cout<<"t: to turn on/off specular lamp light      "<<std::endl;
    std::cout<<"      "<<std::endl;
    std::cout<<"_____________________"<<std::endl;
    std::cout<<"      "<<std::endl;
    std::cout<<"      "<<std::endl;

    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);

    glutInitWindowPosition(100,100);
    glutInitWindowSize(windowHeight, windowWidth);
    glutCreateWindow("1607063 Bedroom");

    glShadeModel( GL_SMOOTH );
    glEnable( GL_DEPTH_TEST );
    glEnable(GL_NORMALIZE);

    glutReshapeFunc(fullScreen);
    glutDisplayFunc(display);
    glutKeyboardFunc(myKeyboardFunc);
    glutIdleFunc(animate);
    glutMainLoop();

    return 0;
}
```