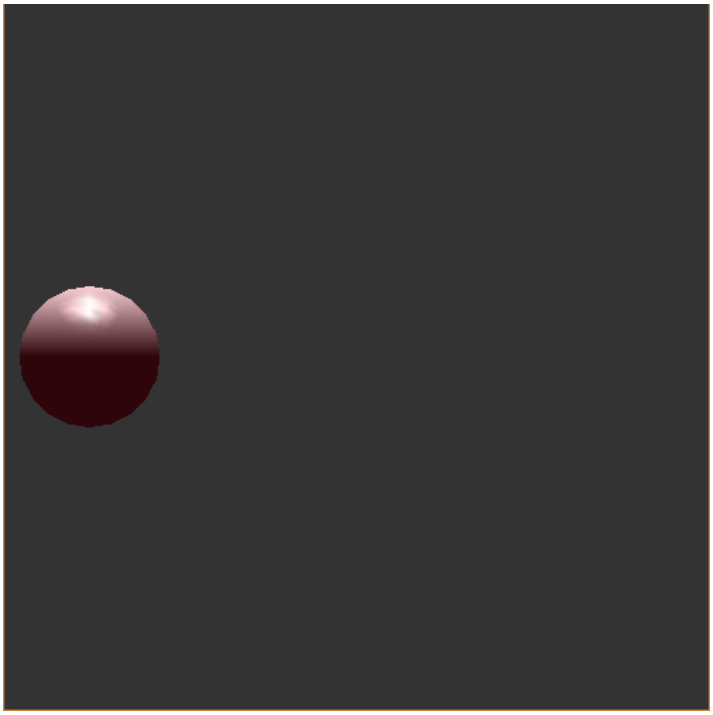
Duc Nguyen
CSE-520
Lab 1: Animation and Lighting

1. As a review of what you learned in CSE 420, write a graphics program that has the following features:
   a. A ball, which can be created using glutSolidSphere() is located at the origin and is lit by some light (any color) from the top.
   b. The ball moves to the right for some distance and then moves in the opposite direction to the left for some distance, then reverses direction again.
2. Compile and run the program.



Sphere on the left side of the screen

Sphere on the right side of the screen

Report:
This lab was not too difficult for me to implement considering my final project consisted a lot of this type of programming and it was entertaining. Anyways, I was easy implementing this and having it work based on what I learned the last time and it should be an easy grade.

```
#include <math.h>
#include <GL/glut.h>

static double sphereTranslateX = 0;

void init(void)
{
        glShadeModel(GL_FLAT);

        GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
        GLfloat mat_diffuse[] = { 1.0, 1.0, 1.0, 1.0 };
        GLfloat mat_shininess[] = { 50.0 };

        GLfloat light_position[] = { 0.0, 0.5, 0.0, 0.0 };
        GLfloat lights[] = { 1.0, 1.0, 1.0 };
        GLfloat lightd[] = { 1.0, 1.0, 1.0 };
        GLfloat lmodel_ambient[] = { 0.9, 0.1, 0.2, 1.0 };
        glClearColor(0.2, 0.2, 0.2, 0.0);
        glShadeModel(GL_SMOOTH);

        glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
        glMaterialfv(GL_FRONT, GL_SHININESS, mat_diffuse);
        glMaterialfv(GL_FRONT, GL_SHININESS, mat_shininess);
        glLightfv(GL_LIGHT0, GL_POSITION, light_position);

        glLightfv(GL_LIGHT0, GL_DIFFUSE, lights);
        glLightfv(GL_LIGHT0, GL_SPECULAR, lightd);
        glLightModelfv(GL_LIGHT_MODEL_AMBIENT, lmodel_ambient);

        glEnable(GL_LIGHTING);
        glEnable(GL_LIGHT0);
        glEnable(GL_DEPTH_TEST);
        glColor3f(0, 1, 0);
}

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

        // The sphere translate between -1 and 1
        glTranslatef((GLfloat) 2 * sin(sphereTranslateX), 0.0, 0.0);
        glColor3f(1.0, 0.0, 0.0);
        glutSolidSphere(0.5, 20, 20);

        glFlush();
        glutSwapBuffers();
}

// Function to Auto Animate the Sphere
void update(int)
{
        sphereTranslateX += 0.1;
        glutPostRedisplay();
        glutTimerFunc(25, update, 0);
}

void reshape(int w, int h)
{
        glViewport(0, 0, (GLsizei)w, (GLsizei)h);
```

```
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        if (w <= h)
        {
                glOrtho(-2.5, 2.5, -2.5*(GLfloat)h / (GLfloat)w, 2.5*(GLfloat)h / (GLfloat)w, -10.0,
10.0);
        }
        else
        {
                glOrtho(-2.5*(GLfloat)w / (GLfloat)h, 2.5*(GLfloat)w / (GLfloat)h, -2.5, 2.5, -10.0,
10.0);
        }
        glMatrixMode(GL_MODELVIEW);
        glLoadIdentity();
}

int main(int argc, char** argv)
{
        glutInit(&argc, argv);
        glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB | GLUT_DEPTH);
        glutInitWindowSize(500, 500);
        glutInitWindowPosition(100, 100);
        glutCreateWindow("Lab 1, Animation and Lighting");
        init();
        glutDisplayFunc(display);
        glutReshapeFunc(reshape);
        glutTimerFunc(25, update, 0);
        glutMainLoop();
        return 0;
}
```