

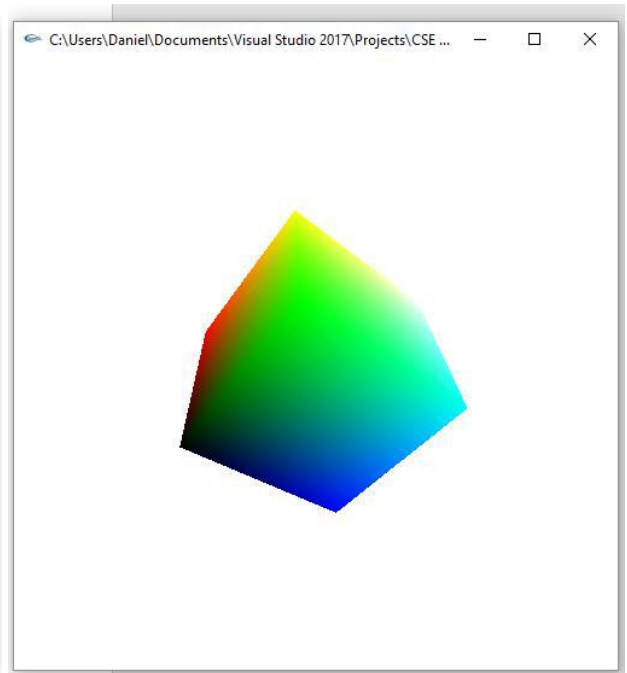
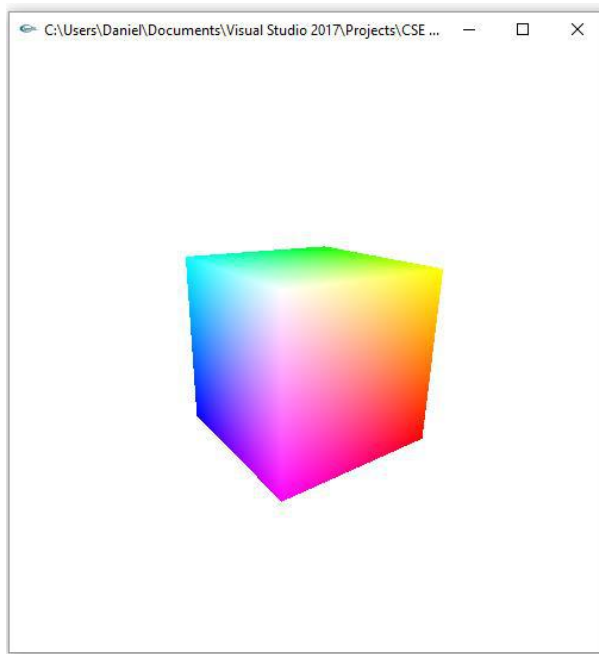
Daniel Meyer

CSE 420-01

Homework Extra Credit

Homework Extra Credit Report

Part 1: (success)



```
#include <stdlib.h>
#include <stdio.h>
#include <iostream>
#include <GL/glut.h>

float ax = 0, ay = 0, az = 0;    // rotations

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

static GLint vertices[] = { -1, -1, -1, //0
                             1, -1, -1,
                             1, 1, -1,
                             -1, 1, -1, //3
```

```

-1, -1, 1,
1, -1, 1,
1, 1, 1,      //6
-1, 1, 1 };    //7

static GLfloat colors[] = { 0.0, 0.0, 0.0,
                            1.0, 0.0, 0.0,
                            1.0, 1.0, 0.0,
                            0.0, 1.0, 0.0,
                            0.0, 0.0, 1.0,
                            1.0, 0.0, 1.0,
                            1.0, 1.0, 1.0,
                            0.0, 1.0, 1.0
};

//0 = black
//1 = red
//2 = yellow
//3 = green
//4 = blue
//5 = magenta
//6 = white
//7 = cyan

void init()
{
    glClearColor(1.0, 1.0, 1.0, 0.0);
    glShadeModel(GL_SMOOTH);

    glEnableClientState(GL_VERTEX_ARRAY);
    glEnableClientState(GL_COLOR_ARRAY);
}

void display(void)
{
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(1.0, 1.0, 1.0);
    glLoadIdentity();
    gluLookAt(3.0, 3.0, 3.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
    glRotatef(ax, 1, 0, 0);
    glRotatef(ay, 0, 1, 0);
    glRotatef(az, 0, 0, 1);

    glVertexPointer(3, GL_INT, 0, vertices);
    glColorPointer(3, GL_FLOAT, 0, colors);
    glEnable(GL_CULL_FACE);
}

```

```

        glCullFace(GL_BACK);

        glDrawElements(GL_QUADS, 24, GL_UNSIGNED_BYTE, allIndices);

        glFlush();
    }

    void reshape(int w, int h)
    {
        glViewport(0, 0, (GLsizei)w, (GLsizei)h);
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        glFrustum(-1.0, 1.0, -1.0, 1.0, 1.5, 20.0);
        glMatrixMode(GL_MODELVIEW);
    }

    void keyboard(unsigned char key, int x, int y)
    {
        switch (key)
        {
            case 'x':        // up
                ax += 5;
                break;
            case 'X':        // down
                ax -= 5;
                break;
            case 'y':        //north
                ay += 5;
                break;
            case 'Y':        //south
                ay -= 5;
                break;
            case 'z':        //west
                az -= 5;
                break;
            case 'Z':        //east
                az += 5;
                break;
            case 27:
                exit(0);
                break;
        }
        glutPostRedisplay();
    }

    int main(int argc, char** argv)

```

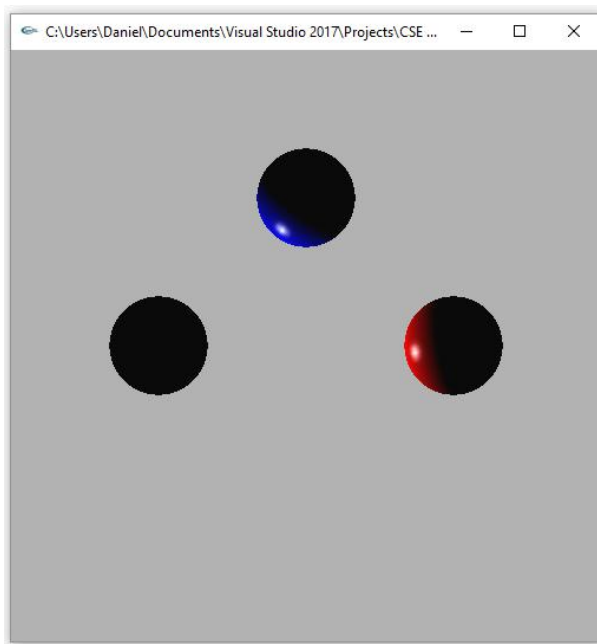
```

{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(500, 500);
    glutInitWindowPosition(600, 600);
    glutCreateWindow(argv[0]);
    init();
    glutDisplayFunc(display);
    glutKeyboardFunc(keyboard);
    glutReshapeFunc(reshape);

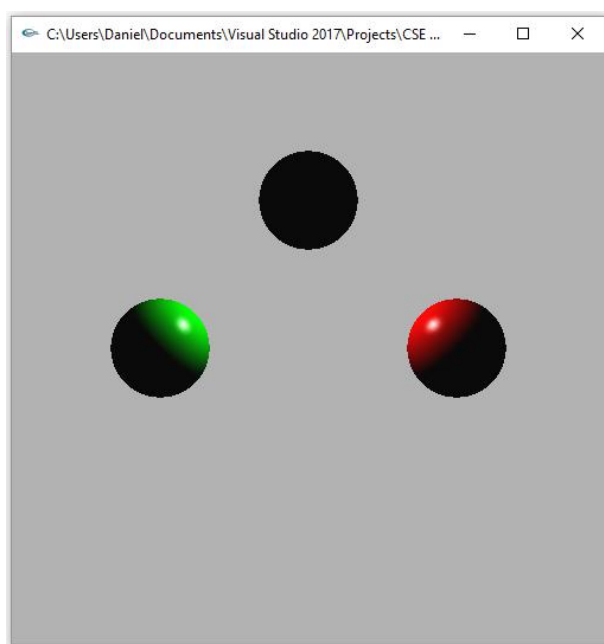
    glutMainLoop();
    return 0;
}

```

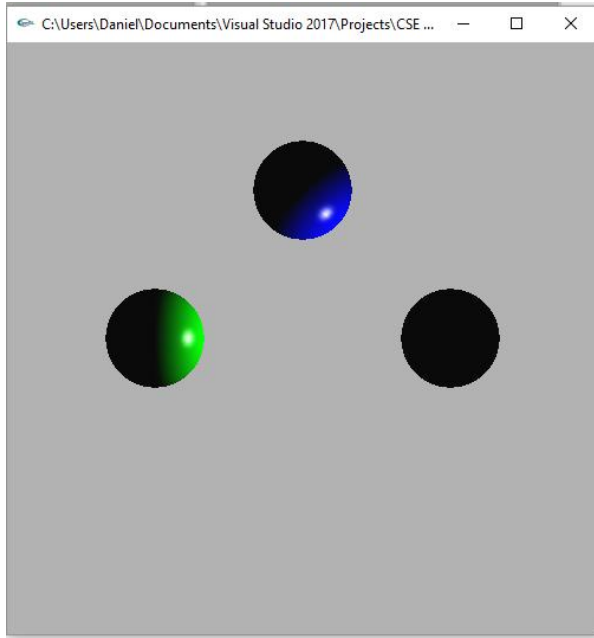
Part 2: (success)



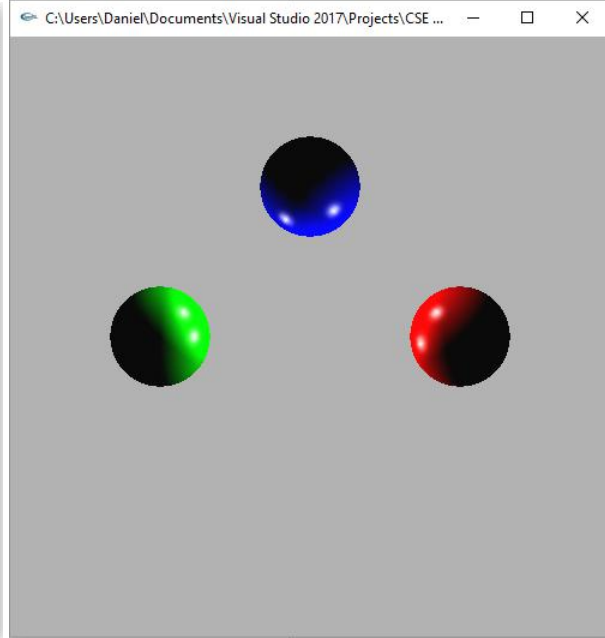
LIGHT0 enabled



LIGHT1 enabled



LIGHT2 enabled



LIGHT0, LIGHT1, LIGHT2 enabled

```
void init(void)
{
    GLfloat no_mat[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat diffuseMaterial[4] = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat ambient[] = { 0.0, 0.0, 0.0, 1.0 };
    GLfloat light[] = { 1.0, 1.0, 1.0 };

    GLfloat light_position0[] = { -3.0, -1.0, -1.0, 1.0 };
    GLfloat light_position1[] = { 0.0, 3.0, 1.0, 1.0 };
    GLfloat light_position2[] = { 3.0, 0.0, 1.0, 1.0 };
    GLfloat spot_direction0[] = { 1.0, 1.0, 1.0 };
    GLfloat spot_direction1[] = { -1.0, -1.0, -1.0 };
    GLfloat spot_direction2[] = { 0.0, -1.0, 0.0 };

    GLfloat local_view[] = { 0.0 };

    glClearColor(0.7, 0.7, 0.7, 0.0);
    glShadeModel(GL_SMOOTH);
    glEnable(GL_DEPTH_TEST);

    glMaterialfv(GL_FRONT, GL_DIFFUSE, diffuseMaterial);
    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
    glMaterialf(GL_FRONT, GL_SHININESS, 100.0);

    glLightfv(GL_LIGHT0, GL_AMBIENT, ambient);
```

```

    glLightfv(GL_LIGHT0, GL_DIFFUSE, light);
    glLightfv(GL_LIGHT0, GL_SPECULAR, light);
    glLightfv(GL_LIGHT0, GL_POSITION, light_position0);
    glLightfv(GL_LIGHT0, GL_SPOT_DIRECTION, spot_direction0);

    glLightfv(GL_LIGHT1, GL_AMBIENT, ambient);
    glLightfv(GL_LIGHT1, GL_DIFFUSE, light);
    glLightfv(GL_LIGHT1, GL_SPECULAR, light);
    glLightfv(GL_LIGHT1, GL_POSITION, light_position1);
    glLightfv(GL_LIGHT1, GL_SPOT_DIRECTION, spot_direction1);

    glLightfv(GL_LIGHT2, GL_AMBIENT, ambient);
    glLightfv(GL_LIGHT2, GL_DIFFUSE, light);
    glLightfv(GL_LIGHT2, GL_SPECULAR, light);
    glLightfv(GL_LIGHT2, GL_POSITION, light_position2);
    glLightfv(GL_LIGHT2, GL_SPOT_DIRECTION, spot_direction2);

    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);
    glEnable(GL_LIGHT1);
    glEnable(GL_LIGHT2);

    glColorMaterial(GL_FRONT, GL_DIFFUSE);
    glEnable(GL_COLOR_MATERIAL);
}

```

```

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

    //Sphere A
    glPushMatrix();
    glColor3f(1, 0, 0);
    glTranslatef(3, 0, 0);
    glutSolidSphere(1.0, 50, 50);
    glPopMatrix();

    //Sphere B
    glPushMatrix();
    glColor3f(0, 1, 0);
    glTranslatef(-3, 0, 0);
    glutSolidSphere(1.0, 50, 50);
    glPopMatrix();
}

```

```

    //Sphere C
    glPushMatrix();
    glColor3f(0, 0, 1);
    glTranslatef(0, 3, 0);
    glutSolidSphere(1.0, 50, 50);
    glPopMatrix();

    glFlush();
}

```

Summary:

For the first part of this assignment I had to create a colored cube with specified colors at each of the 8 vertices. To do this I used 3 vertex arrays: 1 for the indices, 1 for the vertices, and 1 for the colors at each of the vertices. Then I used the GL_VERTEX_ARRAY and GL_COLOR_ARRAY states and set the vertex and color pointers to their respective arrays. Finally I used glDrawElements() with a pointer to the indices array to draw the colored cube. I also added the ability to rotate the cube about each axis using x, X, y, Y, z, Z for each axis respectively. Each time the keys were pressed, the cube would rotate about the specified axis +/- 5 degrees, depending on which key was pressed. The second part of the assignment was to light 3 different colored spheres with 3 different directional light sources with specified directions. Each sphere was to only be lit by the specified 2 directional lights. As seen in each of the screenshots above I was able to do this. Both programs compiled and ran without errors and performed their respective tasks per the assignment instructions. As a result, I believe I earned the full 30 points for this assignment.