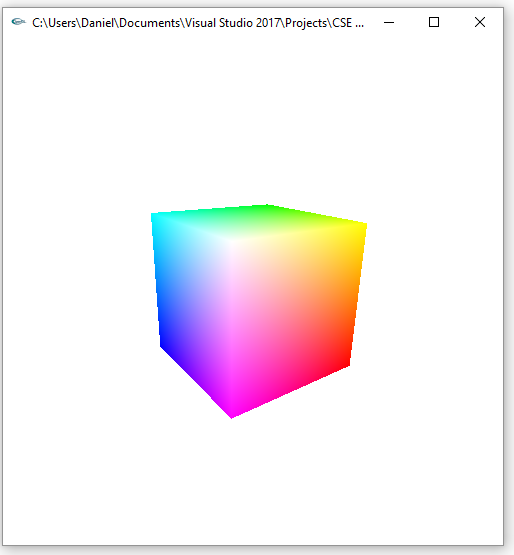
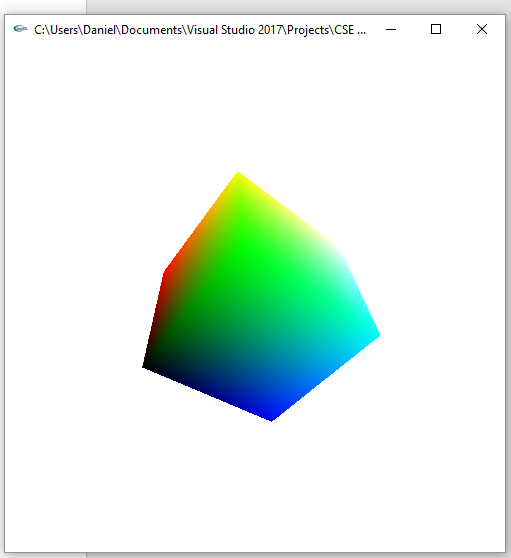
Daniel Meyer

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);

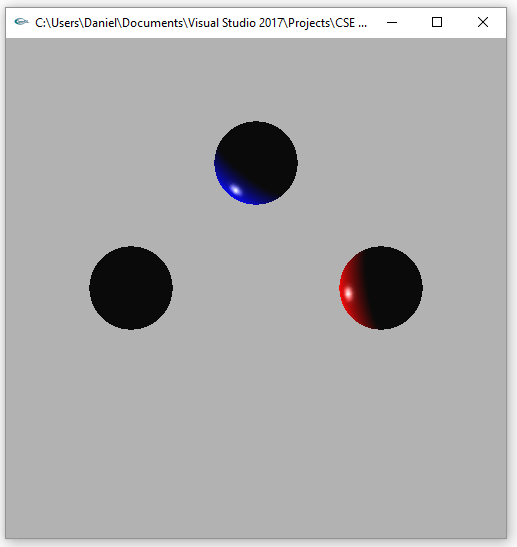
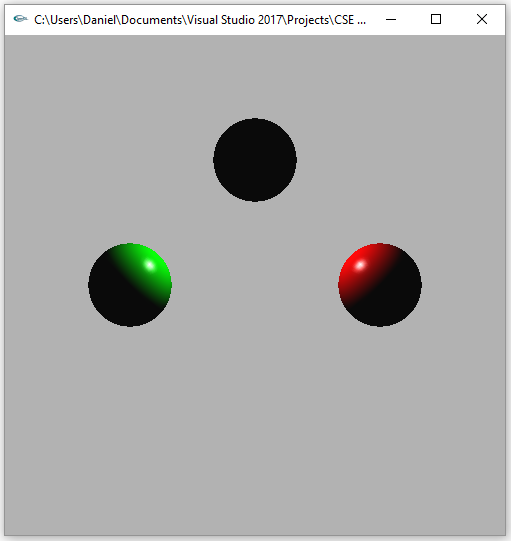
glutReshapeFunc(reshape);

glutMainLoop();

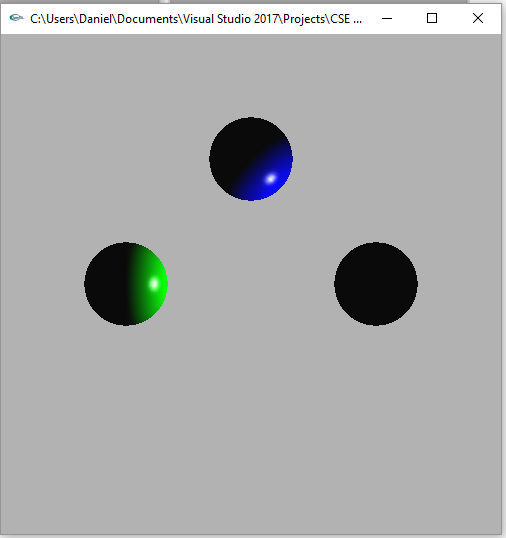
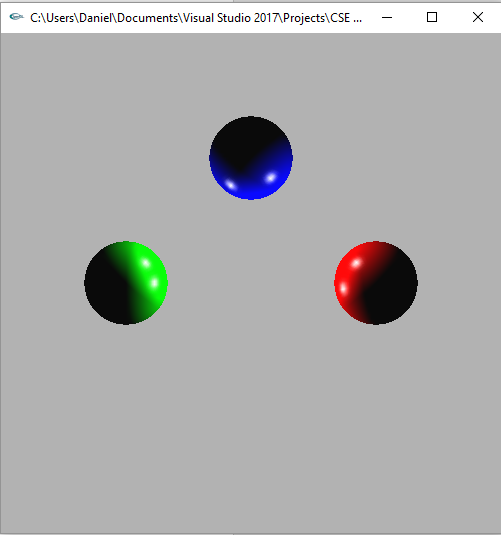
return 0;

}

**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.