Daniel Meyer

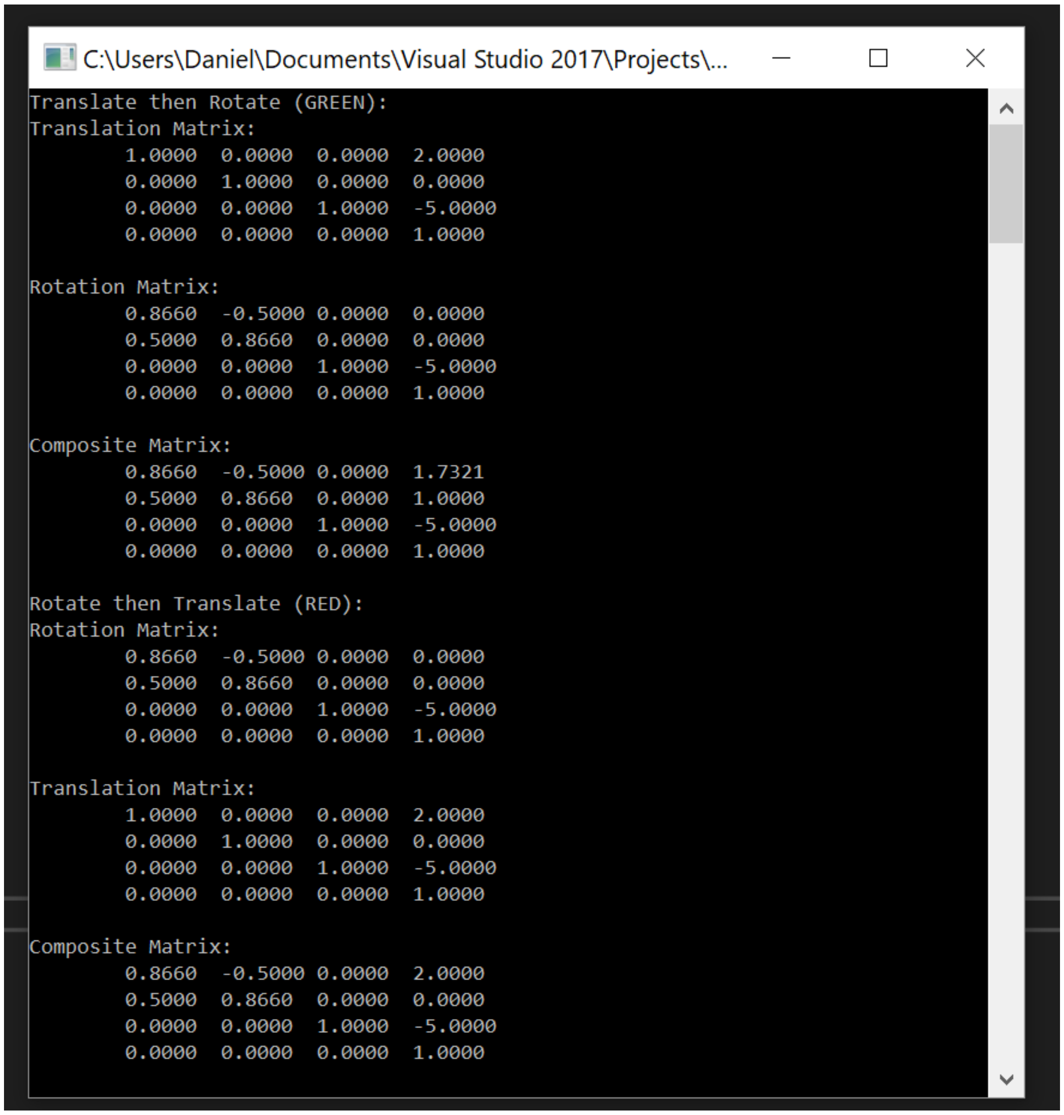
CSE 420-01

Lab 8

Transformations

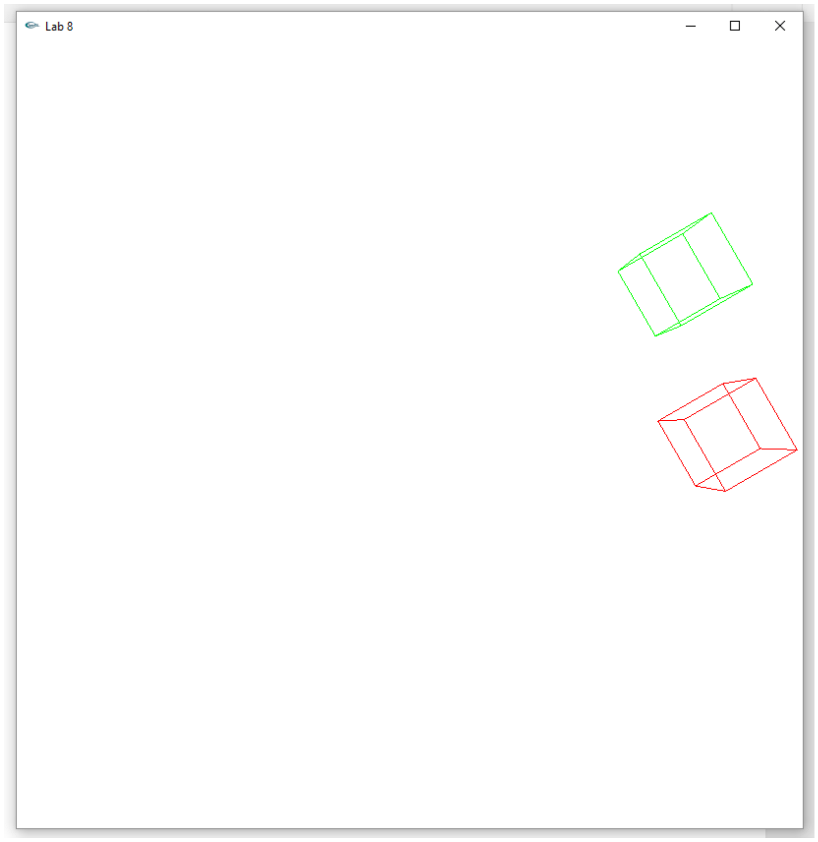
Lab 8 Report

Part 1: (Success)



C:\Users\Daniel\Documents\Visual Studio 2017\Projects\...

```
Translate then Rotate (GREEN):
Translation Matrix:
        1.0000  0.0000  0.0000  2.0000
        0.0000  1.0000  0.0000  0.0000
        0.0000  0.0000  1.0000  -5.0000
        0.0000  0.0000  0.0000  1.0000

Rotation Matrix:
        0.8660  -0.5000 0.0000  0.0000
        0.5000  0.8660  0.0000  0.0000
        0.0000  0.0000  1.0000  -5.0000
        0.0000  0.0000  0.0000  1.0000

Composite Matrix:
        0.8660  -0.5000 0.0000  1.7321
        0.5000  0.8660  0.0000  1.0000
        0.0000  0.0000  1.0000  -5.0000
        0.0000  0.0000  0.0000  1.0000

Rotate then Translate (RED):
Rotation Matrix:
        0.8660  -0.5000 0.0000  0.0000
        0.5000  0.8660  0.0000  0.0000
        0.0000  0.0000  1.0000  -5.0000
        0.0000  0.0000  0.0000  1.0000

Translation Matrix:
        1.0000  0.0000  0.0000  2.0000
        0.0000  1.0000  0.0000  0.0000
        0.0000  0.0000  1.0000  -5.0000
        0.0000  0.0000  0.0000  1.0000

Composite Matrix:
        0.8660  -0.5000 0.0000  2.0000
        0.5000  0.8660  0.0000  0.0000
        0.0000  0.0000  1.0000  -5.0000
        0.0000  0.0000  0.0000  1.0000
```



Green Cube -> Translate then Rotate

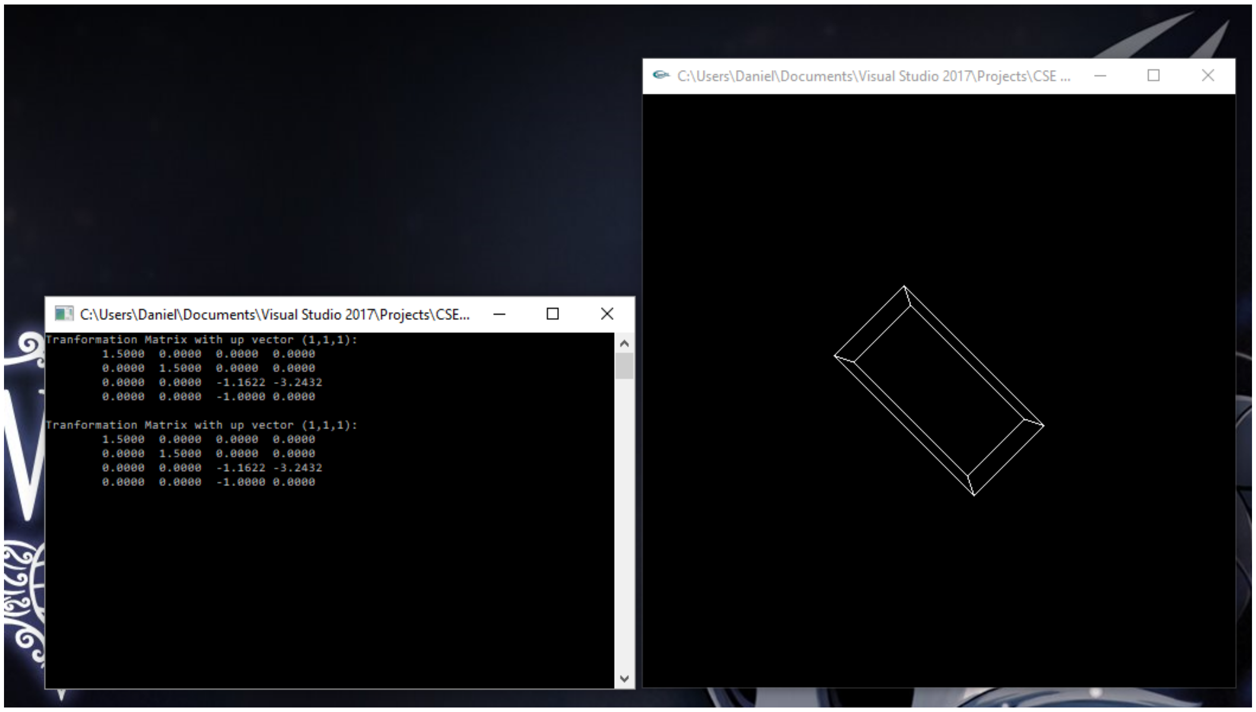
Red Cube -> Rotate then Translate

```
void display(void)
{
	float p[4][4];
	double pd[4][4];

	glClear(GL_COLOR_BUFFER_BIT);
	glMatrixMode(GL_PROJECTION);
	glLoadIdentity();                   // clear the matrix
	                                  // viewing transformation
	glFrustum(-1.0, 1.0, -1.0, 1.0, 2.0, 20.0);

	glMatrixMode(GL_MODELVIEW); // position and aim the camera
	glLoadIdentity();
	gluLookAt(0.0, 0.0, 5.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);

	cout << "Translate then Rotate (GREEN):" << endl;
	glColor3f(0.0, 1.0, 0.0);           //green color
```

```
glPushMatrix();
glTranslatef(2.0, 0.0, 0.0);
glGetDoublev(GL_MODELVIEW_MATRIX, &pd[0][0]);
cout << "Translation Matrix:" << endl;
print_mat(pd);
glPopMatrix();

glPushMatrix();
glRotatef(30, 0, 0, 1);
glGetDoublev(GL_MODELVIEW_MATRIX, &pd[0][0]);
cout << "Rotation Matrix:" << endl;
print_mat(pd);
glPopMatrix();

glPushMatrix();
glRotatef(30, 0, 0, 1);
glTranslatef(2.0, 0.0, 0.0);
glGetFloatv(GL_MODELVIEW_MATRIX, &p[0][0]);
cout << "Composite Matrix:" << endl;
print_mat(p);

glutWireCube(0.5);
glPopMatrix();

cout << "Rotate then Translate (RED):" << endl;
glColor3f(1.0, 0.0, 0.0);        //red color

glPushMatrix();
glRotatef(30, 0, 0, 1);
glGetDoublev(GL_MODELVIEW_MATRIX, &pd[0][0]);
cout << "Rotation Matrix:" << endl;
print_mat(pd);
glPopMatrix();

glPushMatrix();
glTranslatef(2.0, 0.0, 0.0);
glGetDoublev(GL_MODELVIEW_MATRIX, &pd[0][0]);
cout << "Translation Matrix:" << endl;
print_mat(pd);
glPopMatrix();

glPushMatrix();
glTranslatef(2.0, 0.0, 0.0);
glRotatef(30, 0, 0, 1);
glGetFloatv(GL_MODELVIEW_MATRIX, &p[0][0]);
cout << "Composite Matrix:" << endl;
```

```
    print_mat(p);

    glutWireCube(0.5);
    glPopMatrix();

    glFlush();
}
```

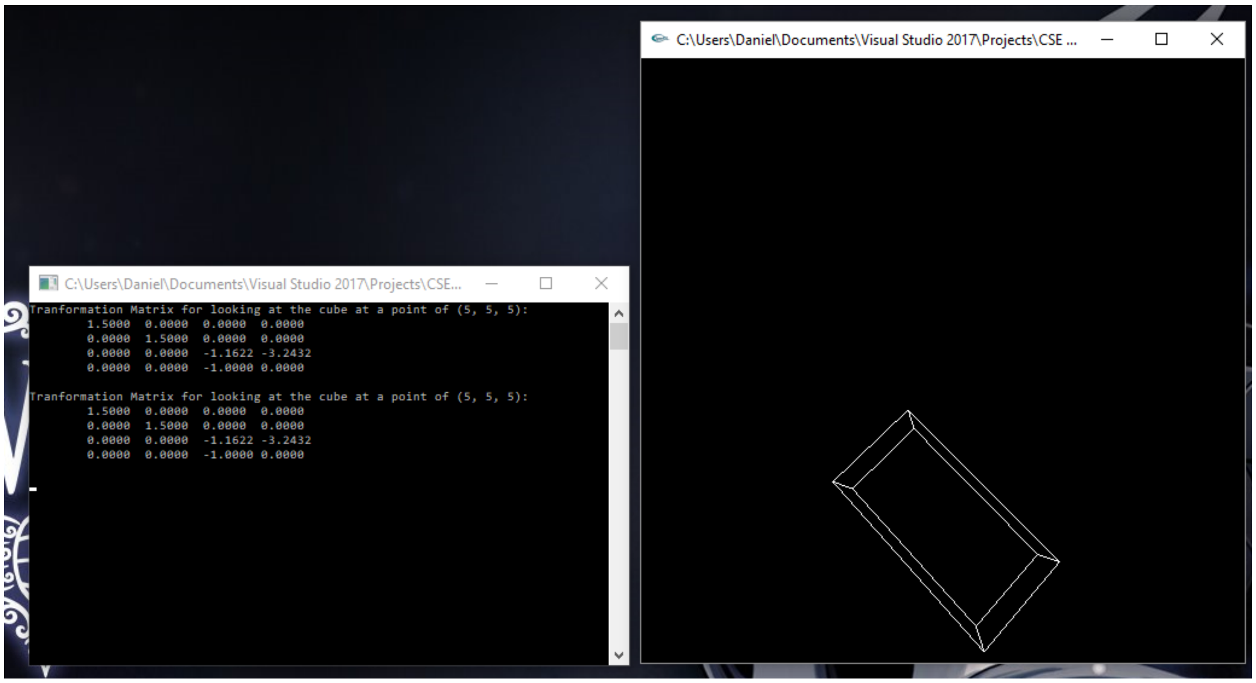
Part 2: (Success)



C:\Users\Daniel\Documents\Visual Studio 2017\Projects\CSE ...
C:\Users\Daniel\Documents\Visual Studio 2017\Projects\CSE...

```
Tranformation Matrix with up vector (1,1,1):
        1.5000  0.0000  0.0000  0.0000
        0.0000  1.5000  0.0000  0.0000
        0.0000  0.0000  -1.1622 -3.2432
        0.0000  0.0000  -1.0000 0.0000

Tranformation Matrix with up vector (1,1,1):
        1.5000  0.0000  0.0000  0.0000
        0.0000  1.5000  0.0000  0.0000
        0.0000  0.0000  -1.1622 -3.2432
        0.0000  0.0000  -1.0000 0.0000
```

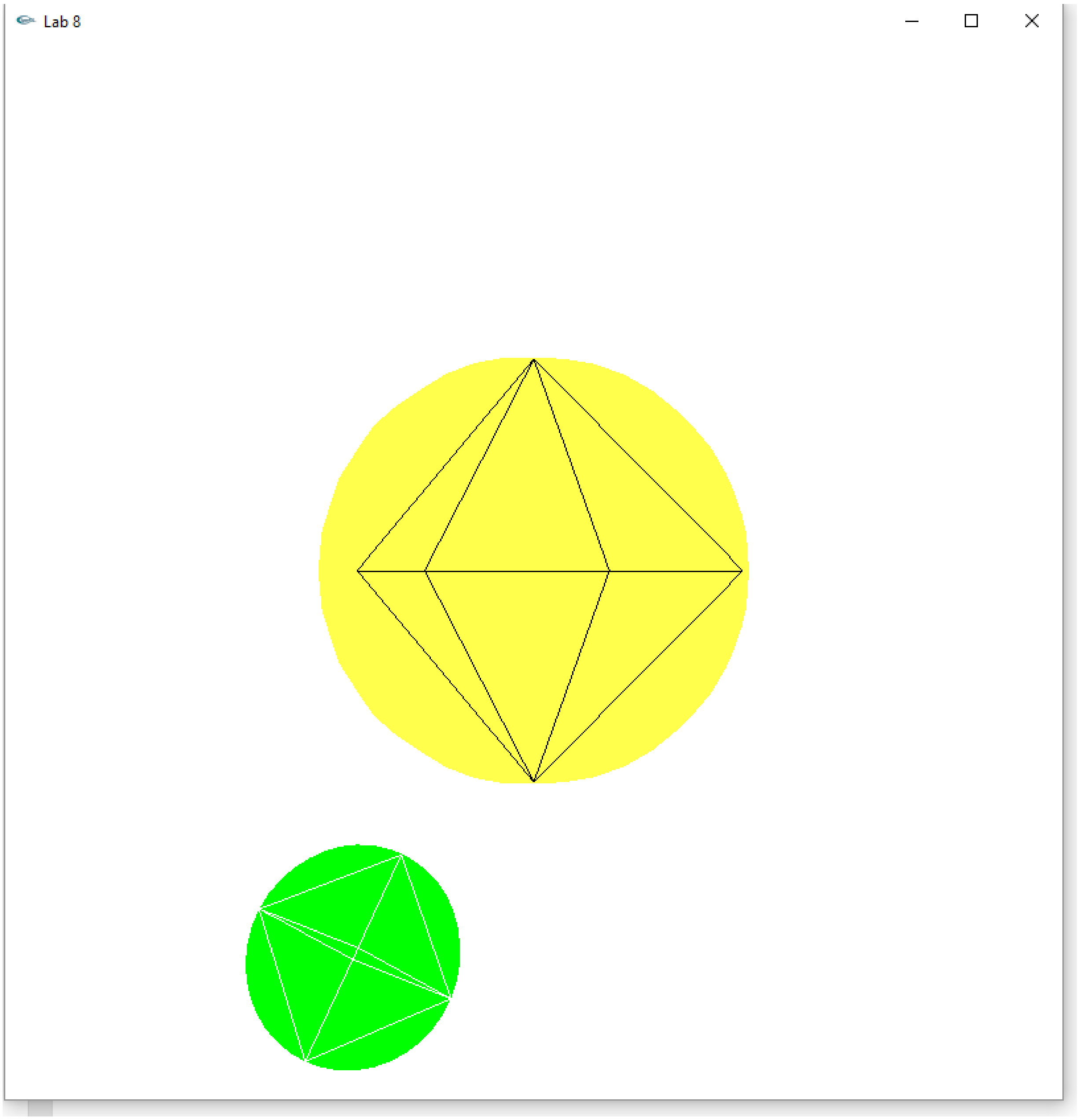
```
gluLookAt(0.0, 0.0, 5.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0);
glScalef(1.0, 2.0, 1.0);
glutWireCube(1.0);

glGetDoublev(GL_PROJECTION_MATRIX, &pd[0][0]);
cout << "Tranformation Matrix with up vector (1,1,1):" << endl;
print_mat(pd);
```


C:\Users\Daniel\Documents\Visual Studio 2017\Projects\CSE...
C:\Users\Daniel\Documents\Visual Studio 2017\Projects\CSE ...

```
Tranformation Matrix for looking at the cube at a point of (5, 5, 5):
        1.5000  0.0000  0.0000  0.0000
        0.0000  1.5000  0.0000  0.0000
        0.0000  0.0000  -1.1622 -3.2432
        0.0000  0.0000  -1.0000 0.0000

Tranformation Matrix for looking at the cube at a point of (5, 5, 5):
        1.5000  0.0000  0.0000  0.0000
        0.0000  1.5000  0.0000  0.0000
        0.0000  0.0000  -1.1622 -3.2432
        0.0000  0.0000  -1.0000 0.0000
```

```
gluLookAt(0.0, 0.0, 5.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0);
glScalef(1.0, 2.0, 1.0);
glutWireCube(1.0);
glGetDoublev(GL_PROJECTION_MATRIX, &pd[0][0]);
cout << "Tranformation Matrix for looking at the cube at a point of
(5, 5, 5):" << endl;

print_mat(pd);
```

Extra Credit: (Success)



Sun (Yellow Sphere) rotates about y-axis.

Earth (Green Sphere) rotates about its own y-axis and then translated 2 units on the x-axis then rotated about the z-axis.

```
void init(void)
{
	glClearColor(1.0, 1.0, 1.0, 0.0);
	glShadeModel(GL_SMOOTH);
	glMatrixMode(GL_PROJECTION);
	glLoadIdentity();
}

void display(void)
{
	glClear(GL_COLOR_BUFFER_BIT);
	glMatrixMode(GL_PROJECTION);
	glLoadIdentity();                // clear the matrix
	glFrustum(-1.0, 1.0, -1.0, 1.0, 2.0, 20.0);
	glMatrixMode(GL_MODELVIEW); // position and aim the camera
	glLoadIdentity();
	gluLookAt(0.0, 0.0, 5.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);

	glPushMatrix();
	glRotatef(ay, 0, 1, 0);
	glColor3f(1, 1, 0.5);
	glutSolidSphere(1.0, 20, 20);
	glColor3f(1, 1, 1);
	glutWireSphere(1.0, 4, 2);
	glPopMatrix();

	glPushMatrix();
	glRotatef(ay, 0, 0, 1);
	glTranslatef(2, 0, 0);
	glRotatef(ay, 0, 1, 0);
	glColor3f(0, 1, 0);
	glutSolidSphere(0.5, 20, 20);
	glColor3f(1, 1, 1);
	glutWireSphere(0.5, 4, 2);
	glPopMatrix();

	glFlush();
}
```

Summary:

For the first part of this lab I created a cube and performed two different transformation cases. The first case was to perform a translation along the x-axis followed by a rotation about the z-axis. The second case was to rotate about the z-axis and then translate along the x-axis. After each transformation I printed out the matrix as well as the composite matrices for each case. The second part of the lab required I create a glutWireCube and then experiment with the camera using gluLookAt(). I performed the suggested cases of an up vector, causing the cube to appear angled, and looking at a specific point, causing the cube to appear lower on the screen. The transformation matrices for the projection matrix were printed for each. Finally for the extra credit portion we were asked to create two spheres, one for the sun (yellow) and one for the earth (green). These were then supposed to perform their respective rotations: the sun about its own axis and the earth about its own axis and the sun. Each step of the program compiled and ran without error. As a result I believe I earned the total 40 points (20 for lab, 20 for EC).