Assignment 3 Report

Problem:

In this assignment, we will be implementing a phong shader model onto a simple red cube to re create a given image.

Algorithm and Methods:

The phong shading model implement three different types of lighting to create a realistic looking lighting on a particular object. The first part is ambient lighting, which just describes a general light in the area, because generally places are not completely dark. The algorithm used to accomplish this effect is rather simple: multiple the color of the light with a small factor.

For the second part, diffuse lighting, we need to calculate the normal vector, the vector that is perpendicular to the surface, and a direction vector that is the difference between the lights position and the fragments position. After these calculations, the resulting cube will have light more prominent on one of the cube faces than the other, giving it the illusion of light coming from a nearby source.

Specular lighting and diffuse lighting are somewhat similar, however specular lighting is changes based on the viewers perspective. Specular lighting depicts reflection off of the objects surface. We also calculate the reflection direction and use both in specular lighting.

Implementation:

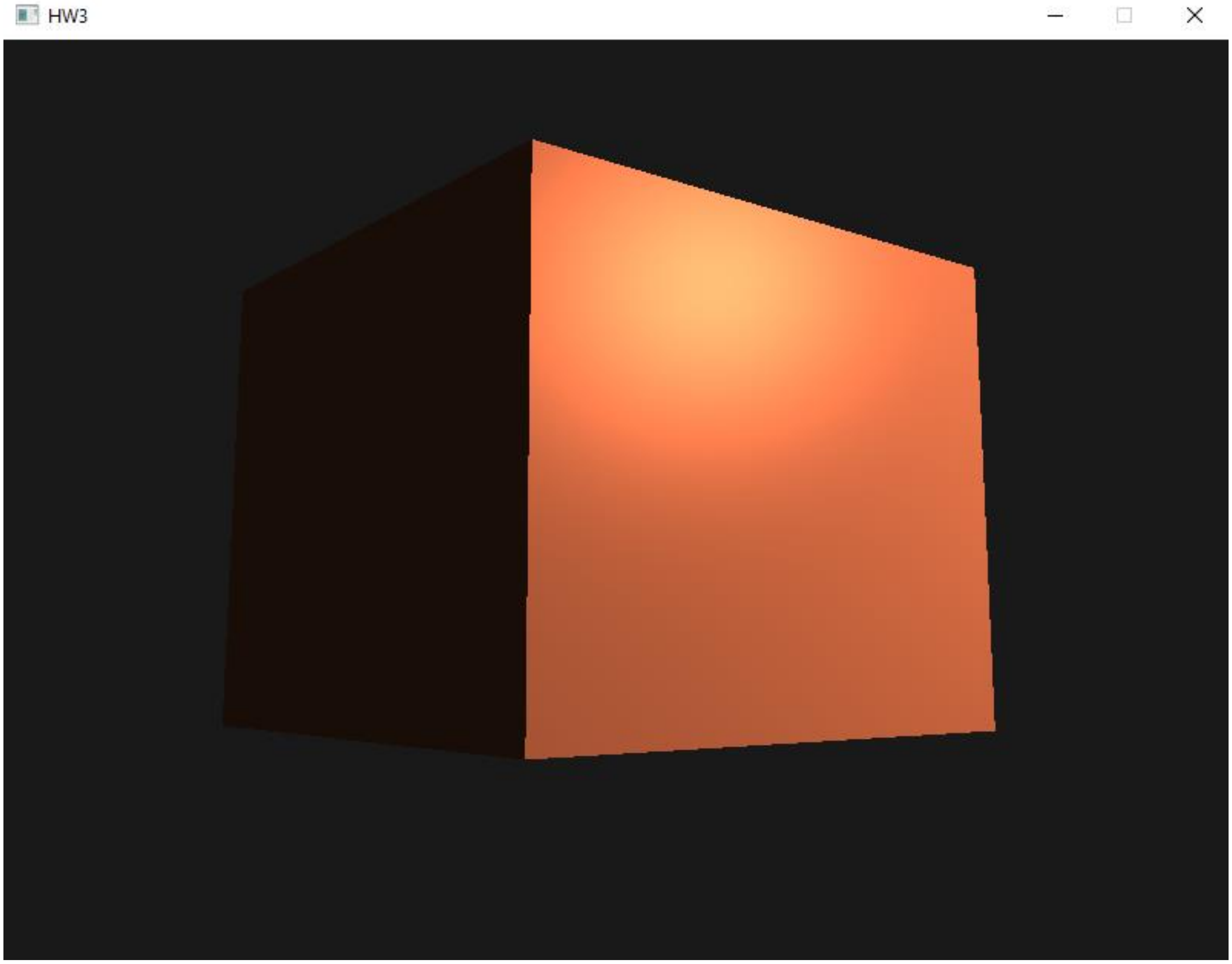
First, we must implement a lookAt function in the GetViewMatrix function in order to get the camera working. The glm function lookAt, takes in a position and a target and an up vector which creates a proper view matrix using the given Euler angles.

Then, we must set up the perspective projection matrix in the main.cpp file. I again used another glm function perspective, which takes in and fov Y, an aspect, zNear, zFar. For the fov, I used the camera.Zoom, and then, experimented with values until the best result came.

For the .vs file, I generated a normal matrix by multiplying the given normal by a transposed and inversed version of the model matrix. And then generated the fragments position by multiplying the model matrix and the vertex positions. These are used for diffuse lighting. Finally, we use the projection, view matrix and frag position to create the gl_Postion.

Finally, in the .frag file. We create the diffuse, ambient, and specular lighting for full phong shader. For ambient, I created the ambient simply by using a magnifying value on the lightColor variable. For the diffuse lighting, I created the normal vector using the normalize function on the normal matrix. Then for the light direction, I used normalized function again on the difference between the light and fragment position. Then we dot product both matrices and multiplied with the lights color to get a the diffused part. Now for the specular part, we get view direction by getting the normalized version of the difference between view position and fragment position. Then, get the reflection direction by using the reflect function on the inverse of the light direction and the normal vector. After getting the dot product of the two, we raise the product by 32 in order to increase the shininess of the light by using the pow function. And then, we add all the ambient diffuse and specular values and multiply by the cube color and make it the Fragment Color.

Results:



This is the resulting cube that mimics the given image.