Homework 4 Report

Zack Fravel
010646947

Problem Statement:

The goal of this programming assignment was to implement Phong shading using C++. The program takes in a text file with scene information and outputs the correct color with shading calculations for each point in the input file. Specifically we were tasked with creating a C++ class with a constructor, destructor, and setCamera(), setLight(), setObject(), and getShade() methods. Not a ton of error handling was required for this, the main thing is making sure we're reading in the input file and storing values correctly as well as checking if dot product values are negative so we don't miscalculate the shading.

Design:

Since the output is a collection of points I decided to extend the Point class I used in my homework 1 and homework 2 to hold values for X, Y, and Z position as well as R G B color values and normal vector values Nx, Ny, and Nz. I also have a Phong class with the methods described in the problem statement implemented. The pros of the design decisions I made are that the program works perfectly for this input file, however a con would be if the inputs changed at all it would need to be expanded and modified.

A high level view of the program flow is as follows. In main, I read in "input.txt" line by line and tokenize each line, storing each token in a data vector. On each line read, I check a line count variable to check if I'm setting Camera, Light, or Object information. Once I'm onto the objects I create a point p, set p's (X, Y, Z), and send p into the getShade() function. Once in getShade(), I normalize all the vectors and calculate the shading for that point based on the Phong equation.

Implementation:

The implementation process started with being able to read in the file. I created a Phong class with the necessary variables and created my set() methods. Once I had an outline for the Phong class, I focused on getting the file input working correctly and made sure I was able to store all the correct variables. Once I verified all variables were being stored properly, I moved on to implementing the actual Phong equation. I use a global instance of the Phong class along with a global vector of Points objects to store the points at the end.

Like I said above, I extended my Point class from the previous assignments to work for this implementation and based my getShade() implementation on hw4.cpp in the the sample sources on the class website. The program runs through the input file line by line and performs appropriate operations based on the current line. On point lines, the program runs all the shading methods and stores the point into the points vector. At the end I loop through the vector of points and print the information.

Testing:

I tested the program incrementally. The beginning consisted of printing to the console all my variables as I was attempting to get the input working. This helped me figure out how to store variables properly and learn more about parsing through text files line by line. The meat of the testing came with debugging the Phong equation. This was harder to test because when you arrive at incorrect values it's hard to tell what is wrong. However, I noticed that my output had the correct pattern so I knew it had to be something small that was scaling everything. I finally figured out that I wasn't calculating my Ambient terms correctly, only taking into account the Object color instead of the Object and the Light.

Everything works precisely as expected! The main error handling we had to look out for accuracy was making sure not to add color whenever dot products came out to be negative. Other than that, the Phong implementation is very straight forward. A sample output is included on the page.

```
Zacks-MacBook-Pro:hw4 Meryl$ ./hw4
100,100,200   59,59,0
125,100,178   59,59,0
125,125,145   60,60,1
100,125,167   60,60,1
100,125,167   60,60,1
125,125,145   60,60,1
125,150,118   62,62,2
100,150,140   62,62,2
100,150,140   62,62,2
125,150,118   62,62,2
125,175,97   65,65,5
100,175,119   64,64,4
100,175,119   64,64,4
125,175,97   65,65,5
125,200,82   64,64,4
100,200,104   63,63,3
125,100,178   0,68,68
150,100,160   0,68,68
150,125,127   0,69,69
125,125,145   0,69,69
125,125,145   0,69,69
150,125,127   0,69,69
150,150,100   0,70,70
125,150,118   0,70,70
125,150,118   0,70,70
150,150,100   0,70,70
150,175,79   1,71,71
125,175,97   1,70,70
125,175,97   1,70,70
150,175,79   1,71,71
150,200,64   1,70,70
125,200,82   0,70,70
150,100,160   78,0,78
175,100,146   77,0,77
175,125,113   78,0,78
150,125,127   79,0,79
150,125,127   79,0,79
175,125,113   78,0,78
175,150,86   79,0,79
150,150,100   80,0,80
150,150,100   80,0,80
175,150,86   79,0,79
175,175,65   79,0,79
150,175,79   80,0,80
150,175,79   80,0,80
175,175,65   79,0,79
175,200,50   79,0,79
150,200,64   80,0,80
175,100,146   23,23,23
200,100,136   23,23,23
200,125,103   23,23,23
175,125,113   23,23,23
175,125,113   23,23,23
200,125,103   23,23,23
200,150,76   23,23,23
175,150,86   23,23,23
175,150,86   23,23,23
200,150,76   23,23,23
200,175,55   23,23,23
175,175,65   24,24,24
175,175,65   24,24,24
200,175,55   23,23,23
200,200,40   23,23,23
175,200,50   23,23,23
```

Conclusions:

The overall result of the project was a huge success! The results I got were off by the provided output by about 1 on around 1/3 of the values but I consider that to be well within the acceptable range. I don't think I would've made any design decisions different other than maybe making a method for normalizing vectors so that code didn't have to be reused 4 times. The project took about 6 to 8 hours of work to complete.