CSC 433 / 533: Spring 2015
Assignment 3

Assigned: Monday, March 30th
Due: 11:59pm Sunday, April 12th

For this assignment extend your viewer program from assignment 2a and replace your vertex and fragment shaders with a vertex and fragment shader pair based on Chapter 7 in the red book. The vertex and fragment shader pair implements the classic lighting model with the lighting calculations done in the fragment shader. Your viewer should support up to four lights and three types of lights. As before, your program will be graded on the linux workstations on the 9th floor of Gould-Simpson building.

For all programs this semester:

1. Provide a Makefile that builds the program. We will not grade programs that do not compile.
2. Your directory should be self-contained and not need any files from other directories. So, it will contain all source code for both the application on the CPU and shader files for the GPU, and it will include any required data files.
3. Report any errors to stderr and provide a usage message when the program is invoked with no parameters.
4. Include a readme.txt file and include any special instructions or assumptions.

Task:

1. **viewer program**
    1. In your makefile create an executable named viewer.
    2. Input to the viewer is an expanded control file that contains both viewing and lighting descriptions.
    3. Usage statement for *viewer*
        -c controlFile
    4. The control file is text file with keyword-value lines. Additions to the control file are the light and viewing keywords. There are no default values for lights or view so they are required entries in a control file.
        i. **obj** <name>
        ii. **rx** <angle> **ry** <angle> **rz** <angle> (angle values are in degrees)
        iii. **t** <x-amount> <y-amount> <z-amount>
        iv. **s** <x-factor> <y-factor> <z-factor>
        v. Apply the transformations in the order they are listed in the file
        vi. **light** [local|spot|directional] ambient r g b color r g b position x y z
            [constAtt ca linearAtt la quadAtt qa]
            [coneDirection x y z spotCosCutoff cutoff spotExponent exp]
            1. [local|spot|directional] – specifies type of light source
            2. For a light source, specify the ambient color and light color as r,g,b
            3. Position specifies position of spot light or local light or otherwise direction to light source. This value needs to be in eye coordinates.

4. Both local and spot light sources are attenuated. The attenuation for a local light source is based on distance from light to location on surface. The attenuation is specified with ConstAtt, linearAtt and quadAtt. The light is further attenuated for a spot light and is based on width of the cone of light and falls off based on an exponent, much like specular highlights. This additional attenuation is specified with coneDirection, spotCosCutoff and spotExponent.
5. All position and direction values for lights are given in eye coordinates

vii. There is a maximum of 4 light sources that can be described in a control file. Ignore any additional light descriptions encountered in a control file.

viii. **view** camera x,y,z focal x,y,z and viewup x,y,z – specifies the initial viewing parameters with the location of the camera, the focal point and the viewup vector, all in world coordinates.

ix. The description of a light source or viewing parameters will be on a single line

5. As in assignment 2, input geometry will be described in Wavefront OBJ files in ASCII format for only polygonal descriptions.
6. You will be reading and using more detail in the material descriptions in a mtllib file. The description of material files by Paul Bourke, http://paulbourke.net/dataformats/mtl/ will be the principle source for the description of material file contents. You only need to process color specified in r g b syntax. Fields in the *newmtl* description to use in this assignment are:
   i. Ka r g b – ambient reflectivity using RGB values
   ii. Kd r g b – diffuse reflectivity using RGB values
   iii. Ks r g b – specular reflectivity using RGB values
   iv. Ns exponent – specular exponent
   i. Skip other directives in the material file.
7. We will switch to do lighting calculations in eye coordinates to match the examples in chapter 7 of red book. And, using the example in chapter 7 you will be able to add lighting and shading as described in this assignment by using a single vertex and fragment shading pair.
8. Control of 3D viewing remains the same as assignment 2.

2. Grading rubric – 100 points total

1. [90 points] reading and displaying obj objects.
    ii. View one object with 1) a single directional light source, 2) a single local light source, and 3) a single spot light source.
    iii. View three objects with 1) a single directional light source, 2) a single local light source, and 3) a single spot light source. The use of modeling transforms will be included in the control file.
    iv. View one object with multiple light sources, 1) of a single type of light source, and 2) with varying types of light sources.
    v. View multiple objects with multiple, varying types of light sources.
2. [10 points] For the written assignment answering the questions below.

Questions

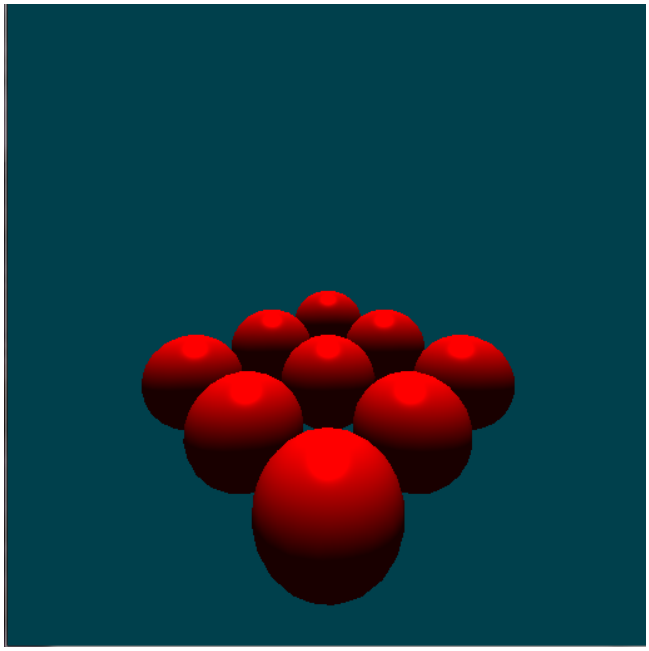
1. [3 points] Describe each of the terms of the classic lighting model, given by the equation

   $I*Ka + I*Kd (N \text{ dot } L) + I*Ks * (N \text{ dot } H)^n$

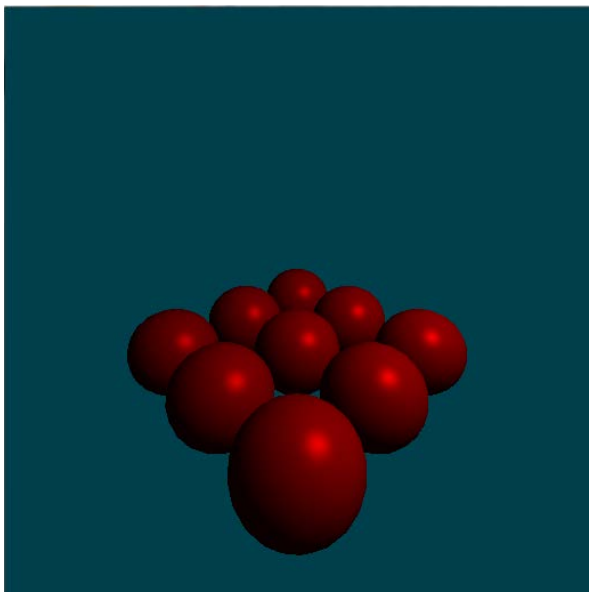
2. [3 points] Describe the meaning of the three terms below, focusing on the application of a lighting model and the surface normal.
   a. Flat shading
   b. Gouraud shading
   c. Blinn-Phong shading

3. [4 points] In the OpenGL 4 Shading Language Cookbook, Second Edition (there is a link on Documentation web page for the class) in chapter 3 there is a description of how to simulate a fog effect.
   a. Provide a summary of how the effect is implemented.
   b. Could this technique be added to a fragment shader that implements the Blinn-Phong lighting model? Why or why not?

Sample output



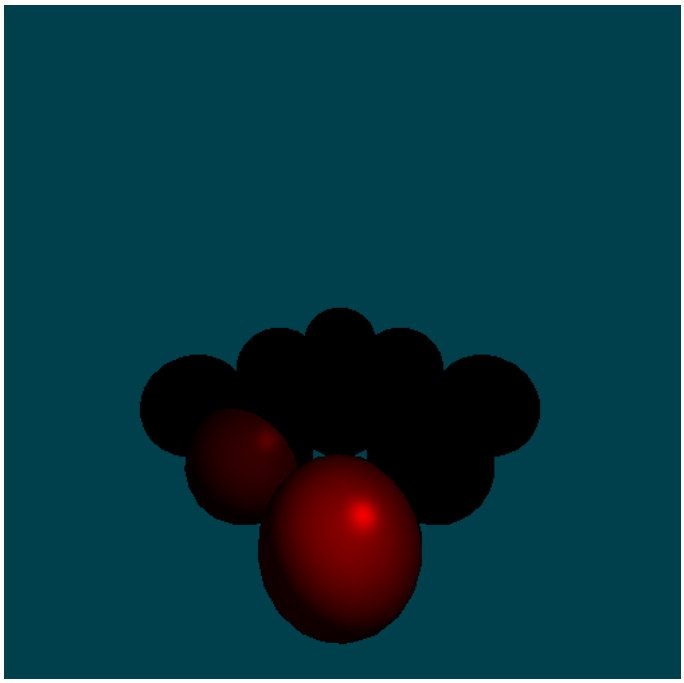
```
obj objFiles/sphere2.obj
obj objFiles/sphere2.obj
t 2. 0. 0.
obj objFiles/sphere2.obj
t 4. 0. 0.
obj objFiles/sphere2.obj
t 0. 2. 0.
obj objFiles/sphere2.obj
t 2. 2. 0.
obj objFiles/sphere2.obj
t 4. 2. 0.
obj objFiles/sphere2.obj
t 0. 4. 0.
obj objFiles/sphere2.obj
t 2. 4. 0.
obj objFiles/sphere2.obj
t 4. 4. 0.

light directional ambient 1. 1. 1. color 1. 1. 1. position 0. 1. 0.

view  camera 6. 6. 4.  focal 0. 0. 0.  viewup 0. 0. 1.
```



```
obj objFiles/sphere2.obj
obj objFiles/sphere2.obj
t 2. 0. 0.
obj objFiles/sphere2.obj
t 4. 0. 0.
obj objFiles/sphere2.obj
t 0. 2. 0.
obj objFiles/sphere2.obj
t 2. 2. 0.
obj objFiles/sphere2.obj
t 4. 2. 0.
obj objFiles/sphere2.obj
t 0. 4. 0.
obj objFiles/sphere2.obj
t 2. 4. 0.
obj objFiles/sphere2.obj
t 4. 4. 0.
```

```
light local ambient 1. 1. 1. color 1. 1. 1. position 5. 5. 0. constAtt 0.2 linearAtt 0.2 quadAtt .002
```

```
view  camera 6. 6. 4.  focal 0. 0. 0.  viewup 0. 0. 1.
```



```
obj objFiles/sphere2.obj
obj objFiles/sphere2.obj
t 2. 0. 0.
obj objFiles/sphere2.obj
t 4. 0. 0.
obj objFiles/sphere2.obj
t 0. 2. 0.
obj objFiles/sphere2.obj
t 2. 2. 0.
obj objFiles/sphere2.obj
t 4. 2. 0.
obj objFiles/sphere2.obj
t 0. 4. 0.
obj objFiles/sphere2.obj
t 2. 4. 0.
obj objFiles/sphere2.obj
t 4. 4. 0.
```

```
light spot ambient 1. 1. 1. color 1. 1. 1. position 5. 5. 0. constAtt 0.001 linearAtt 0.001 quadAtt .002 coneDirection -5. -5. 0. spotCosCutoff .867 spotExponent 25.
```

```
view  camera 6. 6. 4.  focal 0. 0. 0.  viewup 0. 0. 1.
```

Submission Instructions

Submit your files on the host **lectura.cs.arizona.edu** using the command
turnin cs433s15-assg3 [files] We will use your makefile to create your viewer for testing.

Assignment Advice

1. Start early.
2. Do your own work.
3. Check piazza regularly.