Instructor: Sam Buss, Univ. of California, San Diego

Project #4. Add lights and materials to your scene from Project 4.
Due date: Friday, February 17, 10pm.

Goals: Use illumination and shading to make your scene from Project #4 look more three-dimensional. Learn how to shade objects with the Phong lighting model in OpenGL. Create three lights, two or more materials. Use supplied routines to draw normal vectors to help debug your code. Add keyboard controls to your program.

What to hand in: When you are done, place your C++ files, executable, and Visual Studio solution and project files together in a separate folder, named “Project4” preferably, in your PC computer account in the APM basement labs. There should be nothing in this folder except your files for this homework assignment, and the creation/modification dates should be before the turn in deadline. The program must compile and run on these computers. Grading will be personalized and one-on-one with a TA or with Sam Buss. Your program must run on the PC lab, you must come into the PC lab and meet one of us. As usual, you will have to show your source code, run the program, make changes on the spot to your program and recompile as requested by the grader, and be able to explain how your program works and why it renders what it does. The grading should be completed no later than one week after the due date.

Important to note: This may well be the most difficult programming project for the course, as it can be a difficult task to create materials and lights to get a decent combination of ambient, diffuse, and specular lighting. It is also hard to get the various OpenGL features to all work together. You may also need to change the way you form your \((\cos r)(5+r^2)\) surface from Project 3. The textbook web page has two sample programs with source code, SimpleLight and LightTorus, that illustrate many of the techniques you will need for this assignment. It is highly advised you look at these programs. In particular, these programs give examples of how to make lights and materials work. You should not follow these sample programs too closely, but rather should try forming your own material and light properties for your Project 4. Nonetheless, they should be a big help in showing you how to structure your code.

Very important: Before you do any other work on this project, locate your source files from the Project #3. Back them up safely somewhere so you will have a permanent copy. Then start a new project with new copies of these files to be the basis for your Project #4 work.

INSTRUCTIONS:

1. Download the file Project4LightedScene2017.zip from the web. (Full URL for the executable is: http://www.math.ucsd.edu/~sbuss/CourseWeb/Math155A_2017Winter/Project4/Project4LightedScene2017.zip)

2. Run the executable LightedSceneDemo on a PC. You will see a scene with a surface of rotation and a letter "S". Note the following commands act on the scene:
   a. Pressing the arrow keys changes the view position. The rate of movement is controlled by pressing the "R" or "r" key to increase or decrease the step size for movements.
   b. Pressing the "w" key toggles wire frame mode.
c. Pressing the "c" key toggles culling of back faces. Try this in both wireframe mode and in filled-in mode. Try viewing the surface of rotation from the top and the bottom while culling back faces.
d. Pressing the "M" or "m" increases or decreases the mesh resolution on the surface of rotation.
e. Pressing "s" toggles the use of smooth shading and flat shading. The objects look faceted under flat shading, but the facets disappear (somewhat) under smooth shading.
f. Pressing the "1", "2" and "3" keys (one at a time) toggles the three lights off and on.
g. Pressing the "h" switches between
   (a) positional lights and a local viewer
   (b) directional lights and a non-local viewer.
h. Pressing “N” turns on normal vectors on the surface of rotation. Pressing “n” turns off the display of the normal vectors. These are to help you debug your normals.
i. Press the escape key to exit.

3. Experiment with all of these controls. Try increasing and decreasing the mesh fineness while switching back and forth between smooth and flat shading modes. Notice how the specular highlights change size and position when you toggle the "h" command -- understand why this happen. Rotate slowly around the object in smooth and flat shading modes -- note how a little bit of the faceting can be seen even in smooth mode, especially if the mesh size is not very high.

4. **Your job is to re-create this program** -- but with your own three-dimensional geometry from your Project #3.

5. **Form a new project and solution**, and call it LightedScene. Include YOUR source files from Project #3. **Be sure to save an extra copy of your files from Project #3** so you do not lose them!!

6. **Add three lights and ambient lighting.**
   a. When positional, they should be placed at <-5, 5, 0>, <0,5,0> and <5,5,0> (you may change these positions slightly, depending on your geometries from Project #3). When the lights are directional, they should have l vectors equal to these three vectors.
   b. In addition to placing light sources at these position, add three small spheres that show the positions of the lights (these spheres' materials have emission properties that cause them to be visible, but do not need any ambient, diffuse or specular colors). These spheres are visible if the light is both turned on and is positional.
   c. One of the lights must be white. One of the lights must be colored. You are free to choose the same color, another color, or white for the third light.
   d. Also, add some low-level ambient light so that objects can be faintly seen with all three lights turned off.

7. **Keyboard controls "1", "2" and "3" toggle these lights off and on.** Add these controls to the myKeyboardFunc routine. The function calls glEnable(GL_LIGHTi) and glDisable(GL_LIGHTi), for i = 0, 1, 2, are the relevant commands to turn lights on and off.

8. **Give objects in your scene materials and material properties.**
   a. You do not need to match the materials in the supplied executable. Feel free to choose your own colors.
   b. There must be at least two different materials on the objects in the scene. They should have ambient, diffuse and specular color, but should not have emissive color.
c. You will need to add normal vectors for the vertices in the surface of rotation (step #9) before it will look good. See discussion below for normal vectors.

d. Specular highlights must be visible on the surface of rotation and on other materials. White (or gray) is usually the best choice for the specular color. The specular exponent values of your materials should not be higher than 127. **Note:** Specular highlights will not show up on your surface of rotation until you specify normals for its vertices (see steps 8c and 9).

e. The three-dimensional geometries based on your first initial must have a material and show specular highlights. If all of your geometries from Project #3 were square, or flat-faces, you may just add a sphere somewhere to give an extra rounded surface that can show off specular properties.

f. It will take some careful work adjusting the material properties and the light properties to make the scene look good. **This can be hard to do well, so leave some extra time for it.**

g. Part of adjusting the lights and materials is setting the scene to not be too dark or too light. If it is too light (that is, too brightly lit), you may not realize it at first: symptoms of being too light include not being able to see good specular highlights or have the surface look "flat" or "washed out". These are caused when final color values are clamped to the maximum value of 1.0. The fix for this problem is to decrease the brightness of the lights, or the color values of the materials.

h. To repeat: if the specular highlights, or other lighting, do not look right, be sure you are specifying surface normals correctly.

9. **Give normal vectors for the vertices in your surface of rotation.** Normal vectors are automatically given for the built-in glut Solid Objects. If you built some custom geometric shapes you MUST add normal vectors for them too.

   - You probably will need to use the glEnable(GL_NORMALIZE) command (why??!).

   I have written some routines “glsbRenderNormals” to help you visualize the normal in your scene. In the Demo version, these are shown as green lines. To include this in your code, do the following steps:

   a. Include the source files glsbRenderNormals.cpp and glsbRenderNormals.h in your project, by moving the files to same directory as your other source files, and including them in your project.

   b. Add `#include "glsbRenderNormals.h"` at the beginning of your program (after the other #include statements).

   c. Where you first start drawing your surface of rotation, at the first `glBegin(…)` command, add an extra line

      ```cpp
glsbBegin();
```

      somewhere *before* your `glBegin()` command. (If it is in a loop, you may add the glsBegin() command before the loop. You do not need to add multiple glsBegin commands, unless you change the ModelView matrix while drawing the surface of rotation (this is unlikely).

   d. Everywhere you give a pair of commands:

      ```cpp
      glNormal3f(a,b,c);
      glVertex3f(d,e,f);
      ```

      replace them with the four commands

      ```cpp
      glNormal3f(a,b,c);
      glsNormal3f(a,b,c);
      glVertex3f(d,e,f);
      glsVertex3f(d,e,f);
      ```

      These extra “glsb…” commands register the normal for later drawing.
e. Somewhere after your code is finished drawing the surface of rotation (anywhere after the last glEnd() is called), call
   `glsbRenderNormals();`
   This command causes the saved normals to be drawn.

f. Add the following code to the `myKeyboardFunc` to let the drawing of normals be turned off and on:
   ```
   case 'N':
       glsbSetDrawLength(0.3f);  // Enable normals, at the specified length
       glutPostRedisplay();
       break;
   case 'n':
       glsbSetDrawLength(0.0f);  // Disable drawing normal
       glutPostRedisplay();
       break;
   ```

10. **Add the keyboard command** “s” that toggles between flat and smooth shading mode for all materials in the scene. Be able to explain what this makes look different, and why.

11. **Add the “h” keyboard command** to toggle between the two modes (and be able to explain what this does):
    a. Mode (a) with positional lights and local viewer, and
    b. Mode (b) with directional lights and non-local viewer.

12. **Turn in** everything as described above.

Grading is on a scale of 0-10.