Project #1 – Create two octahedral with smooth and flat colors.

Due date: Wednesday, January 25, midnight.

Goals: Gain some basic familiarity with triangle fans and triangle strips. Learn how to make triangles of solid color as well as how to shade colors smoothly. Learn how to use key controls to control viewpoint.

What to hand in: When you are done, place your C++ files, executable, and Visual Studio solution together in a separate folder 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 match the turn in date. 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. 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 must be completed no later than Wednesday one week after the due date.

FOR PROJECT #1, PLEASE DO THE FOLLOWING STEPS #1 - #10.

1. Download the Octahedra program from the zip file OctahedraProject.zip. Extract these into a directory named Octahedra. (Full URL for the zip file is: http://www.math.ucsd.edu/~sbuss/CourseWeb/Math155A_2012W/Project1/OctahedraProject.zip.)

2. There is an executable "OctahedraDemo.exe" that shows (mostly) how your program should end up. Experiment with this program. Notice the following items and keyboard commands.
   a. There are two octahedra shown.
   b. The arrow keys (left, right, up down) control the view position. (Optional: if the rate of movement is too slow, press "R" to make it faster. If it is too fast, press "r" to make the rate slower.
   c. The "w" key toggles wire-frame mode. Back faces are culled.
   d. The shape on the left is smooth shaded, (GL_SMOOTH option). The shape on the right is flat-shaded (GL_FLAT option).
   e. The colors on the top half of the flat shaded octahedron are fully saturated, for example, the red is \( \langle 1,0,0 \rangle \), the yellow is \( \langle 1,1,0 \rangle \), etc. The colors on the bottom half are at one third intensity, e.g., the red is \( \langle 0.33,0,0 \rangle \) the yellow is \( \langle 0.33,0.33,0 \rangle \), etc.
   f. Similar numeric values are used for colors on the smooth shaded shape. The top vertex is colored white and the bottom vertex black --- on the smooth shaded shape.

3. Find the solution "Octahedra.sln" and open in Visual Studio. Examine the source code and run this program. This program looks somewhat like the OctahedraDemo.exe. However, it draws cubes instead of the octahedra. When you examine the source, do the following:
a. Examine the code and understand how the cubes are specified (using GL_QUADS and as GL_QUAD_STRIP).
b. Examine the code to understand the OpenGL commands that control the shading (flat or smooth).
c. Examine the code to understand the OpenGL commands that turn wire frame mode off and on.
d. Understand how OpenGL enables back face culling.
e. Find every place in the code where glutPostRedisplay is called. Understand why it is called in these places.

4. **Re-write the code for the shape on the right.** In place of the code given for the cube on the right, write code that draws the shape shown in OctahedraDemo. Use two triangle fans to build the shape.

5. **Re-write the code for the shape on the left.** In place of the code given for the cube on the left, write code that draws the shape shown in OctahedraDemo. This time, use one triangle strip, plus two individual triangles.

6. Be sure that all faces of the shapes are facing in the correct outward direction. It is recommended that you be careful about specifying vertices in the right order. There should not be any holes in your octahedra. It is OK to use glFrontFace(...) if you wish to.

7. **Implement a new keyboard control “c” to toggle back face culling off and on.** Can you see the difference in wireframe mode? Can you see it in solid mode?

8. **Understand the difference between flat and smooth shading.** Be able to discuss the differences with the TA grading your program, and to explain how the shading is caused by the source code.

9. **Examine carefully the way the program works in wire frame mode.** Do you notice anything unusual as the octahedra are rotated in wire frame mode? Are there any artifacts due to aliasing? Can you see any z-fighting? These effects may be very subtle. If you have trouble seeing aliasing problems, try slowing down the motion by pressing "r" a few times, then hold down arrow in key and watch the edges of the octahedra. Try this in both wire-frame mode and non-wire-frame mode. Be ready to discuss what you see with your TA when you are being graded. (These phenomena may be different on different machines! There is a tendency for OpenGL implementations of lines and of wire-frame mode to have small bugs and this may be part of what you see.)

10. **Turn in the project as described above.**

Program grading: Scale of 0 to 10. Personal grading session with TA or professor.