Homework #2
OpenGL, Lighting, Shading, and Viewing
Due Tuesday, October 9 by the end of class
(Grade out of 100 points)

Question #1 - Basic OpenGL: (25 points)

(a) (5 pts) Consider that you have a function listOfVertices() that generates a list of properly formatted vertices in OpenGL. Write the OpenGL code needed to draw a red polygon composed of these vertices.
(b) (10 pts) Consider that you have a function drawShape() that draws an arbitrary polygon in OpenGL centered at (0,0). Write the OpenGL code needed to scale the shape by a factor of 2 in x and a factor of 3 in y, then rotate by 45º in the xy-plane, then translate by 5 in x and -1 in y.
(c) (7 pts) Write the OpenGL commands needed to choose the projection matrix as the active matrix, reset it to the identify matrix, then set it to a perspective projection with a 20 degree field of view, an aspect ratio of 1024/768, a near plane at 0.1 and a far plane at 10.
(d) (3 pts) What is the other OpenGL command you could use to define a perspective projection matrix?

Question #2 - Lighting and Shading: (20 points)

Assume you are viewing the point at (0, 0, 1) on the surface of a radius-1 sphere centered at the origin. There is a viewer at (2, 2, 2) and a point light at (-3, 3, 3). Compute the **n**, **l**, **v**, and **r** vectors (normalized, please) required for the Phong lighting equations.

Question #3 - Viewing: (15 points)

The equation on slide 39 of the September 27 lecture projects space onto a single plane. Starting from this equation, derive the equation for projecting space into a volume defined by 2 planes, z_{near} and z_{far}. How x and y are projected is unchanged, but z should now map to a range, not to a single value. (*i.e.* z_{near} should map to 1, and z_{far} should map to -1, and the values in between should decrease monotonically with distance)

Question #3 - Cohen-Sutherland Line Clipping: (20 points)

For each of the following line segments,

(a) (6 pts) Determine the 2D Cohen-Sutherland codes for each vertex

(b) (6 pts) Determine whether this line is trivially rejected, trivially accepted, or clipped

(c) (8 pts) If the line segment must be clipped, compute the new endpoints

NOTE: Viewport size is 800x600, origin is at (0, 0) (bottom-left corner), points are defined in window coordinates

Line segment 1: (-200, 700), (400, -300)
Line segment 2: (100, 100), (400, 600)
Line segment 3: (400, 300), (1000, 300)

Question #4 - Polygon Clipping: (20 points)

(a) (12 pts) Clip the polygon shown below using the Cohen-Sutherland algorithm against the left, right, top, and bottom sides (in that order). NOTE: Just draw the output (approximately, but indicate vertices), don't need to show any math.

(a) (8 pts) Clip the polygon shown below using the Weiler-Atherton algorithm. NOTE: Just draw the output (approximately, but indicate vertices), don't need to show any math.