1. The diagram below shows a BSP tree. The numbered darker lines are polygons, and the arrows show the orientation of the polygon.

![Displaying a BSP Tree](image)

a. Suppose we look at this scene from the right side, instead of the left. Use the BSP tree to determine the order in which triangles should be rendered.

b. Construct a valid BSP tree starting at triangle 1, instead of triangle 3 (starting from the left, as shown).

2. Suppose we set up a camera using `gluLookAt(eye=[1,1,3], center=[1,-1,3], up=[1,0,0])`, and then render a scene with three spheres of radius 1, centered at 
   (4, -1, 4), (5, -3, -2), and (7, -4, 1).

   a. What are the best (or “tightest”) distances for the near and far planes?

3. You are rendering a scene for the highway department (since they are the government, you have scored a $1 million dollar contract to make one picture). They insist that your renderings include the shiny little reflective markers along the side of the road. It turns out these are made from retroreflective material, meaning that light is reflected mostly back in the direction of the light source.

   a. Sketch a goniometric diagram for a retroreflective surface.

   b. You remember learning that the OpenGL lighting model is 
      \[ k_a + k_d (N \cdot L) + k_s (N \cdot H)^n \]. How would you change this equation to model retroreflection?
4. Suppose we have a graphics system that rasterizes polygons by turning on pixels that are strictly inside the bounds of the polygon. For example:
   a. Suppose we execute the following, which pixels will be turned on?

   ```
   glBegin();
   glVertex(0, 0);
   glVertex(6, 0);
   glVertex(6, 6);
   glVertex(0, 6);
   glEnd();
   ```

   b. Suppose that this system cannot rasterize quadrilaterals directly, and instead breaks them into triangles. What are the vertices of two triangles that would represent this quad?

   c. Suppose we now rasterize these two triangles, which pixels will be turned on?

   d. What is wrong with this rasterizer and how might we fix it?

5. You are designing a graphics system that only supports texture maps that are 256 x 256. Obviously, proper texture filtering will be required in some cases. As a filter, you select box filtering over the area covered by a pixel.

   As a particular example of texture filtering, let's consider the pixel marked with an X in the diagram above.
   a. If T(256, 256) references the texel color at u, v coordinates (1, 1) in the texture image, write an expression (math involving multiple texel values) for the color which should be copied into the frame buffer at pixel X.

   b. Next, your boss tells you that you must support summed area tables. Write an expression for the color at X, assuming your texture is stored in this new format.