1. Look at vertex.h. What’s added from BallAndTorus?  
   glm::vec3 normal;  
   glm::vec3 bumpedNormal;
2. Look at light.h and material.h. Recall before that the light matrix has rows for: Ambient, Diffuse, Specular  
     
   And columns for:  
   Red, Green, Blue
3. Find the light matrix in main.cpp . Look back at light.h. Where is the location of the light?  
   vec4(0.0, 1.0, 100.0, 0.0 )
4. Change the location of the light to the z-axis and what happens?  
    vec4(0.0, 0.0, 100.0, 0.0 ) , it does not illuminate the plane.
5. The material matrix has rows for:  
   Ambient, Diffuse, Specular, Emittance  
     
   And columns for:  
   Red, Green, Blue, Alpha
6. Look at plane.h and plane.cpp. Here again we have a parametrized space used to build a surface. What is the parametrized space for the plane? (Going backwards from code to space instead of the way we did cylinder, hemisphere, torus)

i:0-200, j:0-200

1. Is this an x-y plane, y-z plane, x-z plane, or some weird plane?  
   x-z plane.
2. What is the non-bumped normal to this plane?  
   vec4(0.0, 1.0, 0.0, 0.0)
3. What is the equation for the bumped normal to this plane, the one that gives the ILLUSION that the plane has a bumpy surface?

x = 2.0 \* cos(2.0 \* u);  
y = 1.0;  
z = 0.0;

Does the plane have a bumpy surface?  
No.

1. Go Through the vertexShader.glsl and find all of these calculations. copy and paste:

ambient light \* material property:  
fAndBGlobAmb = globAmb \* planeFandB.ambRefl;

cos(θ) \* diffuse light \* material prop where cos(θ) = ?  
max(dot(normal, lightDirection), 0.0)

cos(φ) \* specular light \* material prop where cos(φ) = ?  
pow(max(dot(normal, halfway), 0.0), planeFandB.shininess)

1. What kind of viewing volume do we have?  
   Frustum
2. Is there a camera? Where is it? Where is it looking? What is the up vector?  
   Yes, it’s at vec3(0.0, 5.0, 30.0), looking at vec3(0.0, 0.0, 0.0), and its up is vec3(0.0, 1.0, 0.0)
3. Lastly, what is the normal matrix?  
   A 4x4 matrix.
4. We have code that transforms objects. What happens with the normals? Are they transformed with the same modelview matrix?  
   Yes, the normal are transformed as well.
5. Look at picture on page 474 of translating a torus. Do the normal change in translations?  
   Yes.
6. Where is the translational part of the modelview matrix?  
   Translational: a14, a24, a34
7. What part of the above matrix is the rotational and scaling part?  
   Rotational: a11, a12, a13, a21, a22, a23, c31, c32, c33  
   Scaling: a11, a22, a33
8. What is the inverse of a matrix?  
   The inverse of a matrix is the matrix where the product of the two is the identity matrix.  
   N-1N = Identity matrix
9. Let’s do an easy one.   
   Suppose N = S(2,3,4).  
   Write N = What can we do to “undo” S(2,3,4)?  
   Multiply it by its inverse!  
     
   N-1 =
10. Suppose N = Rx(90)=What transformation “undoes” this rotation? N-1 =Rx(-90)=Check that N-1N= Identity matrix.  
    It does.
11. How do we find the inverse of a product, N=S(2,3,4)Rx(90)Rz(45)? What is N-1?  
    Multiply it by its inverses in a backwards order.  
    N-1= Rz(-45) Rx(-90) S(0.5,1/3,0.25)
12. Finally, what is the transpose of a vector?  
    A transpose vector is a the vector as a column.  
     \_ \_  
     | 2 |  
    [2 3 4]T = | 3 |  
     | 4 |  
     -- --
13. What is the transpose of a matrix?  
    Rows becomes columns.
14. What is the transpose of a product of matrices?

(AB)T = BTAT

1. How are normals transformed? Suppose *n* is the normal vector at vertex V in object O. Let vector *x* be tangent to the surface of object O at vertex V. Then What about *n*?  
   *n* is perpendicular to *x*.