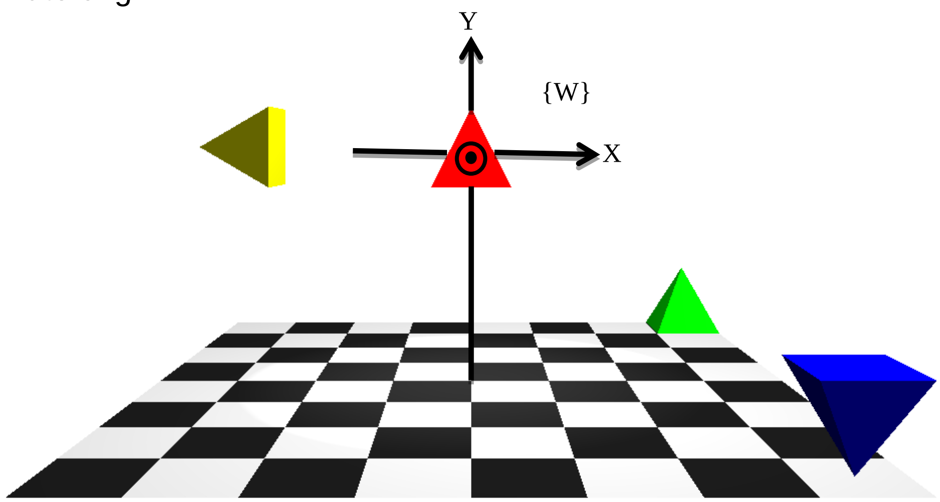
1. In the rendering below, all pyramids where rendered using the same vertices. The World coordinate origin is depicted by the bullseye in the middle of the center pyramid.. The World coordinate X axis points to the right, Y points up, and the World coordinate Z axis is coming out of the page. All of the pyramids pyramids are one unit high and one unit wide. The modeling transformation for the center pyramid is equal to the identity matrix. The checker board is eight units wide. Each square on the board is one unit square. The checker board is three units below the World coordinate origin.



a. The back right pyramid is positioned so that it exactly covers the back right square on the checker board. Write a series of C++ statements to create translation and/or rotation matrices and multiply them together to create the modeling transformation for the back right pyramid.

```
glm::mat4 modelingTransformation =
```

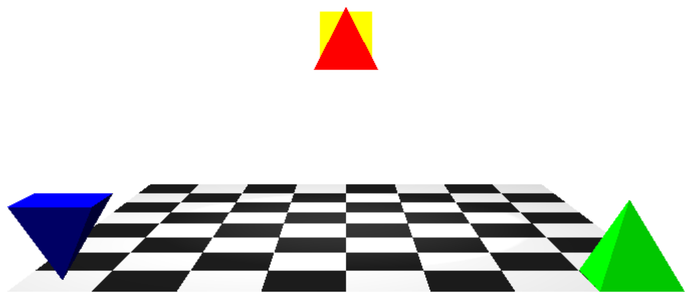
b. The tip of the front right pyramid is just touching the center of the front right square on the checker board. Write a series of C++ statements to create necessary translation and/or rotation matrices and multiply them together to create the modeling transformation for the front right pyramid.

```
glm::mat4 modelingTransformation =
```

c. The pyramid on the left is three units from the center pyramid in the World coordinate negative X direction. Write a series of C++ statements to create necessary translation and/or rotation matrices and multiply them together to create the modeling transformation for the pyramid on the left.

```
glm::mat4 modelingTransformation =
```

2. Recall that without viewing transformations, the eye is always at the World coordinate origin looking in the negative Z direction. Consider the view of the same scene as the previous question that is shown below. In this view the World coordinate X axis is pointing towards the view point. The World coordinate origin (location of the center pyramid) is 15 units from the view point.



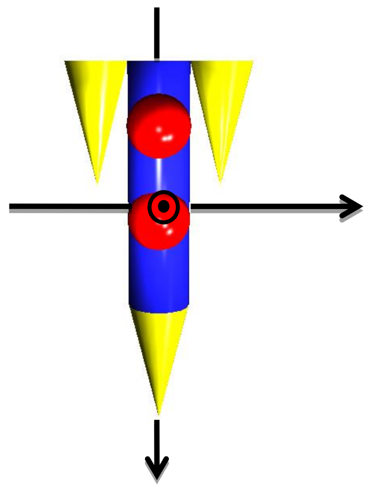
   a. Write a series of statements to create translation and/or rotation matrices and multiply them together to create the viewing transformation for the view of the scene shown above.

```
glm::mat4 viewingTransformation =
```

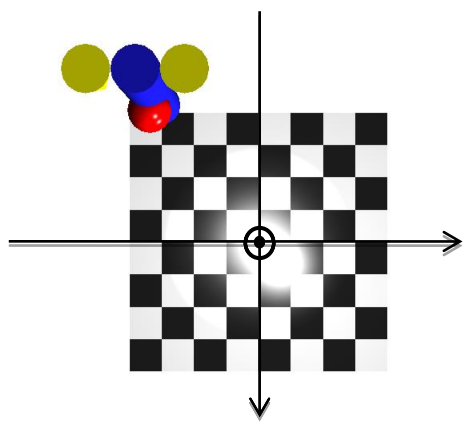
   b. Use the GLM lookat function to create a viewing transformation that is the same as that which is shown in part a.

```
glm::mat4 viewingTransformation =

   glm::lookat(  glm::vec3(________________, ________________, ________________),

                 glm::vec3(________________, ________________, ________________),

                 glm::vec3(________________, ________________, ________________));
```

3. The spaceship model is depicted below relative to object coordinates i.e. the picture shows the position of the spaceship without any modeling transformations applied. The object coordinate Y axis is coming out of the page. The “nose” is pointing the positive Z direction. The “top” of the model is represented by the two red spheres and is oriented in the positive Y direction.

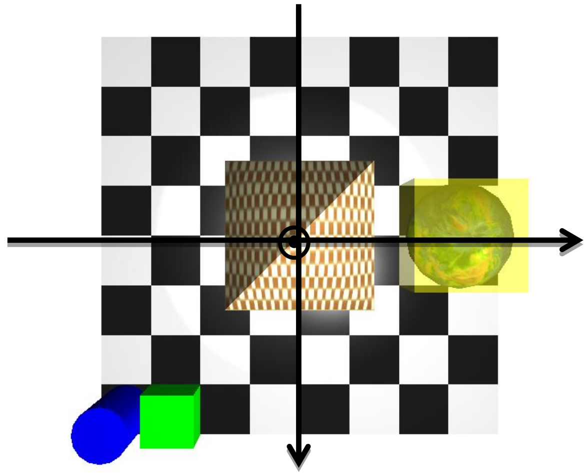


Write a series of statements to create translation and rotation matrices and multiply them together to create the composite modeling transformation that will position and orient the spaceship as depicted below. Relative to World coordinates, the nose of the spaceship is pointing in the negative Y direction.. The position of the ship is X = -2.5, Y = 3, Z = -3.5 relative to the World coordinate origin. (9 points)



```
glm::mat4 modelingTransformation =
```

4. The picture belows depict the locations and orientation of objects in a scene relative to the World coordinate origin when no viewing transformations applied. The Y axis is coming out of the page. The tip of the pyaramid is at the World coordinate origin. The floor/board is 3 units below the World coordinate origin. (Recall that without viewing transformations, the eye is always at the World coodinate origin looking in the negative Z direction.)



   a. Write a series of statements to create translation and rotation matrices and multiply them together to create the viewing transformation for the view of the same scene that is shown below. The tip of the pyramid is centered in the view at a distance of 20. (9 points)

```
glm::mat4 viewingTransformation =
```

b. Use the GLM lookAt function to create a viewing transformation that is the same as that which is shown in part a. (6 points)

```
glm::mat4 viewingTransformation =

   glm::lookat(  glm::vec3(________________, ________________, ________________),

                 glm::vec3(________________, ________________, ________________),

                 glm::vec3(________________, ________________, ________________));
```