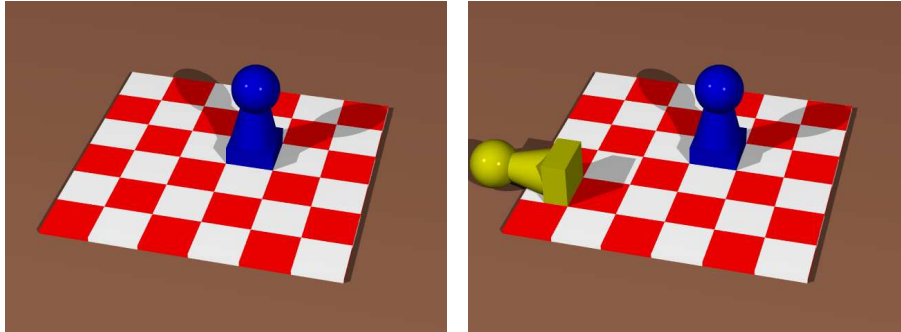


1. Consider the two images below. In the image on the left, the blue pawn is positioned with its front left corner at (0,0,0). The side of each square on the chess board is 1 unit in world coordinates. You are located at (3, 9, 12) and looking at the origin as you view this image. The blue pawn was rendered with the function call `drawPawn("blue")`



You must add code to this to produce the image on the right, that is, the image with the toppled yellow pawn in addition to the blue pawn. The function call `drawPawn("yellow")` will draw the yellow pawn in the exact same position and orientation as the blue pawn. Indicate how the call to `drawPawn` for the yellow pawn could be combined with appropriate transformations in the form of calls to *scale*, *rotateX/Y/Z*, *translate* to produce the image on the right.

- (a)







```
modelViewMatrix = mat4();
modelViewMatrix = mult(modelViewMatrix, rotateZ(90));
modelViewMatrix = mult(modelViewMatrix, translate(-2.0,0.0,2.0));
drawPawn("yellow");
```
- (b)

```
modelViewMatrix = mat4();
modelViewMatrix = mult(modelViewMatrix, translate(-2.0,0.0,2.0));
modelViewMatrix = mult(modelViewMatrix, rotateY(90));
drawPawn("yellow");
```
- (c)

```
modelViewMatrix = mat4();
modelViewMatrix = mult(modelViewMatrix, translate(-2.0,0.0,2.0));
modelViewMatrix = mult(modelViewMatrix, rotateZ(90));
drawPawn("yellow");
```
- (d)

```
modelViewMatrix = mat4();
modelViewMatrix = mult(modelViewMatrix, rotateY(-90));
modelViewMatrix = mult(modelViewMatrix, translate(-2.0,0.0,2.0));
drawPawn("yellow");
```

Assume that you are on the positive z-axis, looking down the negative z-axis, and you see a car centered at the origin. Below there is a table of six pictures on the left. In those pictures, you see the original image and five other images labeled A, B, C, D, and E. In the column on the right you see six WebGL code segments, labeled “original” and “Problem 2” through “Problem 6”. The code segment labeled “original” produced the picture labeled “original”. The other code segments each represent a transformation that is applied to the original. Given that information, indicate for each problem-numbered code segment the letter of the picture corresponding to it.

Original	
A	
B	
C	
D	
E	

Original:

```
modelViewMatrix = mat4();
// Three do-nothing transformations
modelViewMatrix = mult(modelViewMatrix, translate(0,0,0));
modelViewMatrix = mult(modelViewMatrix, rotate(0, vec3(0,1,0)));
modelViewMatrix = mult(modelViewMatrix, scalem(1,1,1));
```

Problem 2.

```
modelViewMatrix = mat4();
modelViewMatrix = mult(modelViewMatrix, translate(0,0,0));
modelViewMatrix = mult(modelViewMatrix, rotate(45,vec3(1,1,0)));
modelViewMatrix = mult(modelViewMatrix, scalem(1,1,1));
```

Problem 3.

```
modelViewMatrix = mat4();
modelViewMatrix = mult(modelViewMatrix, rotate(45, vec3(0,1,0)));
modelViewMatrix = mult(modelViewMatrix, translate(1,0,0));
modelViewMatrix = mult(modelViewMatrix, scalem(1,1,1));
```

Problem 4.

```
modelViewMatrix = mat4();
modelViewMatrix = mult(modelViewMatrix, translate(0,0,-1));
modelViewMatrix = mult(modelViewMatrix, rotate(0, vec3(0,1,0)));
modelViewMatrix = mult(modelViewMatrix, scalem(1,1,1));
```

Problem 5.

```
modelViewMatrix = mat4();
modelViewMatrix = mult(modelViewMatrix, translate(0,0,0));
modelViewMatrix = mult(modelViewMatrix, rotate(45, vec3(0,1,0)));
modelViewMatrix = mult(modelViewMatrix, scalem(1,2,1));
```

Problem 6.

```
modelViewMatrix = mat4();
modelViewMatrix = mult(modelViewMatrix, rotate(45, vec3(0,1,0)));
modelViewMatrix = mult(modelViewMatrix, translate(0,0,0));
modelViewMatrix = mult(modelViewMatrix, scalem(1,1,1));
```

7. A rectangular solid with world coordinate vertices at $(0, 0, 0)$, $(-20, 0, 0)$, $(-20, -40, 0)$, $(0, -40, 0)$, $(0, 0, -100)$, $(-20, 0, -100)$, $(-20, -40, -100)$, $(0, -40, -100)$ is to be rotated *clockwise* by 45 degrees about the line through its geometric center that is parallel to the z-axis. On the sheet of paper that you bring with you to class, write the matrix product that will achieve this transformation. Do not carry out the matrix multiplication – leave it in the form of a factored product of individual 4×4 matrices. The value of any sin or cos should not be expressed in decimal form but rather as $\pm \frac{1}{2}$, $\pm \frac{\sqrt{3}}{2}$, $\pm \frac{\sqrt{2}}{2}$, 0, or ± 1 . If you need a sin or cos value other than these three, you’re doing something wrong – start over.