October 4, 2017 Name ___________________

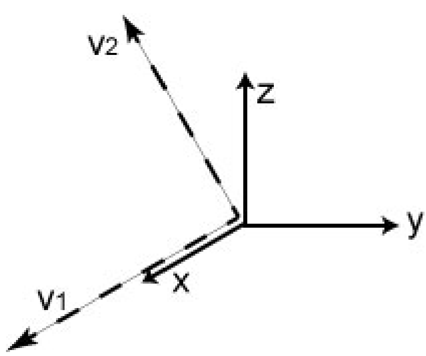
CS445 Midterm

Fall 2017

1.	(max = 9)	5.	(max = 16)
2.	(max = 10)	6.	(max = 10)
3.	(max = 11)	7.	(max = 12)
4.	(max = 12)	8.	(max = 20)
Final Score:		(max=100)	

Please try to write legibly. The instructor's complete failure to decipher what you write will be considered an incorrect answer.

1. (9 pts) Coordinate Systems: Suppose you want to compute 3 basis vectors (xyz) for a coordinate system. You are given 2 vectors v_1 and v_2 and told that the x direction points along v_1, and v_2 lies in the x-z plane. Show below how to compute the xyz basis vectors for your coordinate system in terms of v_1 and v_2. The basis vectors x,y,z should be perpendicular to each other and of length 1.
   .
   Assume you are using a ***right handed*** coordinate system as shown.



x = v1/||v1||

y = (v2 X v1) / || v2 X v1||

z = x X y

2. (2 pts each, 10 pts total) Multiple Choice – Circle the correct answer:

a) The cross product (0, 0, 1) X (0, 1,0) is
   i) (1,0,0)
   ii) (0,1, 1)
   iii) (-1, 0, 0)
   iv) None of the above

b) The cross product (0, 0, 1) X (0, 0,1) is
   i) (0,1,0)
   ii) (1,1, 0)
   iii) (-1, 0, 0)
   iv) None of the above

c) The dot product of two vectors has the largest magnitude when the vectors are
   i) perpendicular
   ii) parallel
   iii) at a 45 degree angle
   iv) none of the above

d) In homogeneous coordinates, the expression (x,y,z,w) with w = 1 represents
   i) A vector
   ii) A point
   iii) A scalar
   iv) None of the above.

e) If a polygon mesh is rotated using Euler rotations in the rotation order RxRzRy, then gimbal lock can occur when which axis is rotated by 90 degrees?
   i) X axis
   ii) Y axis
   iii) Z axis
   iv) None of the above

3. (11 pts total) Suppose you have that:
- **M** is the modeling transformation matrix
- **V** is the view matrix.
- **Ry(θ)** is the rotation by θ about the y axis
- **T(x,y,z)** is a translation by (x,y,z).
- **p** is a vertex in the *object coordinate system*.

In terms of above, what is the sequence of transformations that will:

i) (2 pts) Transform p from object to camera coordinates? V M

ii) (3 pts) Transform p from object to camera coordinates while also rotating p by θ about the **world** y-axis. V Ry(θ) M

iii) (3 pts) Transform p from object to camera coordinates while also rotating p by θ about the **camera** y-axis. . Ry(θ) V M

iv) (3 pts) Transform p from object to camera coordinates while also rotating **p** around an axis that both intersects a point **q** and is parallel to the **camera's y axis**. Assume **q** is expressed in camera coordinates. T(q) Ry(θ) T(-q) V M

4. (12 pts total) Concepts:
   a) (6 pts) Explain what the z-buffer is and how it works.

      Z buffer is
      - a 2D buffer the size of the window
      - each pixel location (i,j) contains the smallest z value of the fragment seen so far at that location
      - if a new fragment at a given location has a smaller z value then that fragment replaces the current fragment at that location in the image (display buffer)
      - if the new fragment at a given location has a larger z value then the new fragment does not replace the current fragment at that location in the image

   b) (6 pts) What is the purpose of the projection transform? For example, what does it do and why does it do it?

The projection transform transforms the view frustum into a cube centered at the origin and of size 2x2x2

Why does it do this:
- Makes it easy to clip polygons that extend outside of the frustum
- Makes it easy to project the volume down to a projection screen
- Makes the transformation to screen coordinates easier to do.

5. (4 pts each, 16 pts total) **3D Transforms:** What is the 4x4 matrix transform (in homogeneous coordinates) for the 3D transformations below. *Also give the inverse*.

a) A rotation of 30 degrees about the x axis:

The transform:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(30) & -\sin(30) & 0 \\ 0 & \sin(30) & \cos(30) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The inverse:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(-30) & -\sin(-30) & 0 \\ 0 & \sin(-30) & \cos(-30) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \text{ or } \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(30) & \sin(30) & 0 \\ 0 & -\sin(30) & \cos(30) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

b) A scale by 100 about the origin and along the z axis.

The transform:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The inverse:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/100 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

c) A translation by -2 along x and 7 along z.

The transform:

$$\begin{pmatrix} 1 & 0 & 0 & -2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 7 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The inverse:

$$\begin{pmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -7 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

d) A reflection through the x-axis.

The transform:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The inverse:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

6. (5 each, 10 pts total) **Composition of 3D Transforms:** What is the sequence of transformations needed to achieve the operations given below. Also, include the corresponding inverse. You do not need to write out the 4x4 matrices. Instead, make use of the syntax:

   Scale: $S(s_x,s_y,s_z)$
   Translation: $T(t_x,t_y,t_z)$
   Rotation: $R_x(\Theta)$, $R_y(\Theta)$, $R_z(\Theta)$.

   a) A uniform scale by 100 about the point (1,2,4).

      The transforms:

      T(1,2,4)   S(100,100,100) T(-1,-2,-4)

      The inverse:

      T(1,2,4)   S(1/100,1/100,1/100) T(-1,-2,-4)

   b) A rotation by 70 degrees about an axis that is parallel to the y axis and intersects the point (8,0,3).

      The transforms:

      T(8, 0, 3)   Ry(70)   T(-8 0, -3)

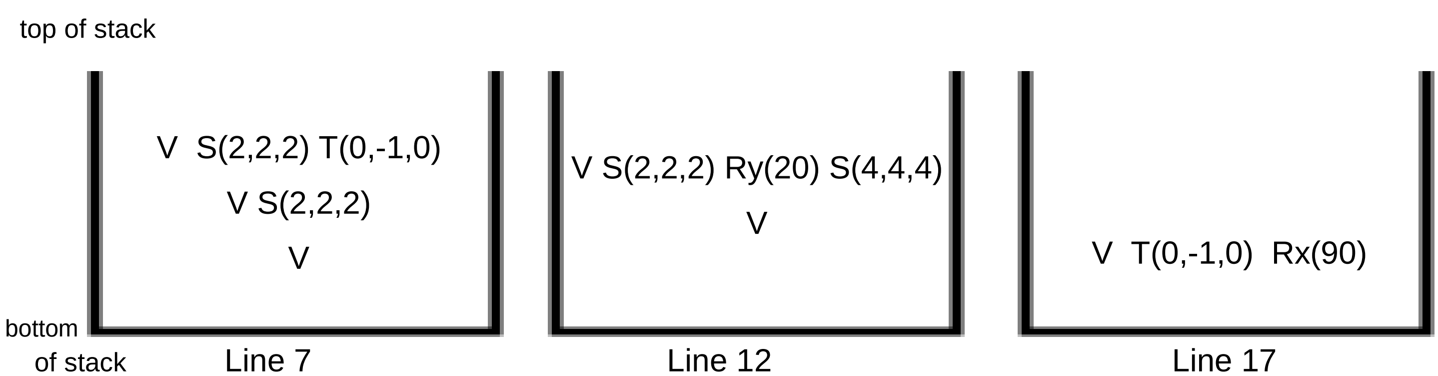
      The inverse:

      T(8, 0, 3)   Ry(-70)   T(-8 0, -3)

7. (12 pts total) Given the following sequence of commands in webGL:

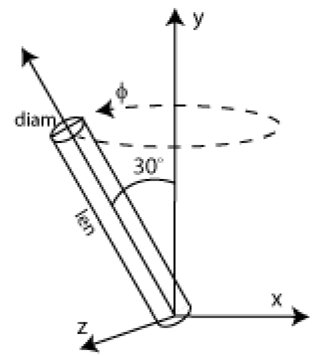
```
Line  0: stack.clear(); // places identity on stack
Line  1: stack.multiply(V); // V is the view matrix
Line  2: stack.push();
Line  3: stack.multiply(scalem(2, 2, 2));
Line  4: stack.push();
Line  5: stack.multiply(translate(0, -1, 0));
Line  6: gl.uniformMatrix4fv(uModel_view, false,
flatten(stack.top()));
Line  7: Shapes.drawPrimitive(Shapes.cube);

Line  8: stack.pop();
Line  9: stack.multiply(rotateY(20));
Line 10: stack.multiply(scalem(4, 4, 4));
Line 11: gl.uniformMatrix4fv(uModel_view, false,
                                  flatten(stack.top()));
Line 12: Shapes.drawPrimitive(Shapes.disk);

Line 13: stack.pop();
Line 14: stack.multiply(translate(0, -1, 0));
Line 15: stack.multiply(rotateX(90));
Line 16: gl.uniformMatrix4fv(uModel_view, false,
                                  flatten(stack.top()));
Line 17: Shapes.drawPrimitive(Shapes.cone);
```

Show what is stored on the matrix *stack* at these lines?



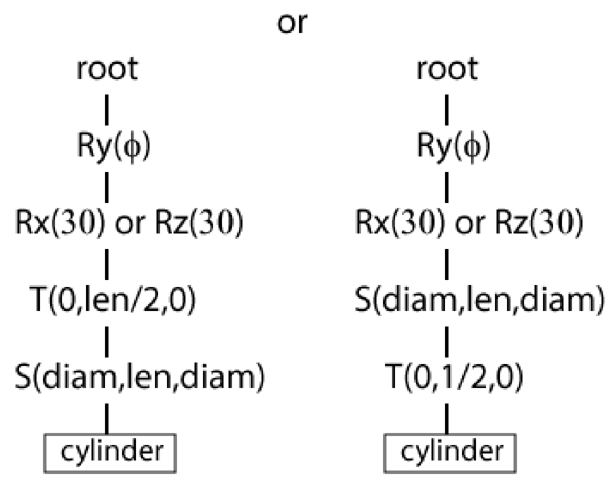
8. (20 pts total) Scene Graphs.

a) (10 pts) Suppose you have a rod tilted by a <u>fixed</u> 30 degrees from the y-axis. It can rotate about the y-axis (fixed at origin) by angle ϕ as shown. The rod is a cylinder with length `len` and diameter `diam`, as indicated.



Draw the <u>scene graph</u> for this rod: Assume that you have access to a cylinder primitive with diameter 1, height 1, centered at the origin, aligned with the y-axis.

Scale transformations should be indicated as **S(sx, sy, sz)** where you fill in specific values for s_x, s_y, and s_z. Similarly, translations and rotations should have the form **T(tx, ty, tz)**, **Rx(angle)**, **Ry(angle)**, and **Rz(angle)**.



b) (10 pts) We now place a second rod of the same dimensions on top of the first, forming a T shape. *The first rod moves exactly as before.* The second rod moves with the first rod and also rotates by θ about the axis of the first rod, as shown.

Draw the scene graph for the entire object. It should build on the scene graph in part a). If you want, you can just draw the graph for the second rod, and then draw an arrow to show where it would attach to the scene graph in part a). Make sure it is very clear where this part connects to the above scene graph. *Include push and pops.*