CS174A Assignment 1 - Part 1
Written Section: Transformations

1. a. In coordinate system A,

$p = 2i_A - j_A + O_A$ in homogeneous coordinates $[2\ -1\ 1]^T$

In coordinate system B,

$p = 3i_B + j_B + O_B$ in homogeneous coordinates $[3\ 1\ 1]^T$

In coordinate system C,

$p = -3i_C + \frac{7}{3}j_C + O_C$ in homogeneous coordinates $[-3\ \frac{7}{3}\ 1]^T$

b. since vector = point1 − point2, we need 2 points to describe a vector.

In coordinate system A,

$\vec{v} = [1\ 4\ 1]^T - [-1\ 5\ 1]^T = [2\ -1\ 0]^T$

In coordinate system B,

$\vec{v} = [4\ 4\ 1]^T - [6\ \frac{11}{2}\ 1]^T = [-2\ \frac{3}{-2}\ 0]^T$

In coordinate system C,

$\vec{v} = [1\ -\frac{2}{3}\ 1]^T - [0\ -\frac{2}{3}\ 1]^T = [1\ 0\ 0]^T$

c. According to lecture 03 Slides, "Coordinate Frame defined by the matrix: $[\vec{a}\ \vec{b}\ \vec{c}\ \vec{O}]$" where $\vec{a}\ \vec{b}\ \vec{c}$ are the basis vectors and $\vec{O}$ is origin.

d. $M_A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix}$ $M_B = \begin{bmatrix} -1 & 0 & 6 \\ -1 & 2 & 2 \\ 0 & 0 & 1 \end{bmatrix}$ $M_C = \begin{bmatrix} 2 & 3 & 2 \\ -1 & -3 & 5 \\ 0 & 0 & 1 \end{bmatrix}$

e. According to lecture03 slides, " $P = [\vec{a}\ \vec{b}\ \vec{c}\ \vec{o}]\begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ 1 \end{bmatrix}$

where $P = \vec{O} + P_1\vec{a} + P_2\vec{b} + P_3\vec{c}$ "

thus for coordinate system A,

$$M_A \cdot P_A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \\ 1 \end{bmatrix}$$ which is $p = 0 + 3\vec{i} + 1\vec{j}$.

The same goes for coordinate system B,

$$M_B \cdot P_B = \begin{bmatrix} -1 & 0 & 6 \\ -1 & 2 & 2 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 3 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \\ 1 \end{bmatrix}$$

$$M_C \cdot P_C = \begin{bmatrix} 2 & 3 & 2 \\ -1 & -3 & 5 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} -3 \\ \frac{7}{3} \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \\ 1 \end{bmatrix}$$

which proves that $P_{world} = M_i \cdot P_i$, where $i = A, B, C$ coordinate systems.

2. The scaling matrix is $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}$ with no translation.

Thus the affine matrix is $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

3. One solution is that, we can scale x, y, z with (1, 1, 2) respectively and then translate x, y, z with (1, 1, 1)

For any 4x4 Matrix A, we shall have M such that $A_{new} = M A_{old}$

$$= M_t (M_s A_{old})$$

Thus the OpenGL Shader Commands to generate M ~~a~~ are below:

```
modelMatrix.setAsIdentity();
modelMatrix = modelMatrix * ~~model~~ Translate(1,1,1);
modelMatrix = modelMatrix * Scale(1,1,2);
```

4. Point $P = [2, 10, 8, 4]^T$

which $w=4$. divided by w.

$P = [1/2, 5/2, 2, 1]^T$ thus $P = (1/2, 5/2, 2)$ in 3D.

5. At first, modelMatrix $M = \begin{bmatrix} 1&0&0&0 \\ 0&1&0&0 \\ 0&0&1&0 \\ 0&0&0&1 \end{bmatrix}$ is an Identity.

After translation,

$$A = M * \begin{bmatrix} 1&0&0&2 \\ 0&1&0&3 \\ 0&0&1&0 \\ 0&0&0&1 \end{bmatrix} = \begin{bmatrix} 1&0&0&2 \\ 0&1&0&3 \\ 0&0&1&0 \\ 0&0&0&1 \end{bmatrix}$$

After Rotation,

$$B = A * \begin{bmatrix} 0&0&1&0 \\ 0&1&0&0 \\ -1&0&0&0 \\ 0&0&0&1 \end{bmatrix} = \begin{bmatrix} 0&0&1&2 \\ 0&1&0&3 \\ -1&0&0&0 \\ 0&0&0&1 \end{bmatrix}$$

And then $B = \begin{bmatrix} 0&0&1&2 \\ 0&1&0&3 \\ -1&0&0&0 \\ 0&0&0&1 \end{bmatrix}$ is push to the stack

After scaling, $B1 = B * Scale(1, 0.5, 1)$

$$= \begin{bmatrix} 0 & 0 & 1 & 2 \\ 0 & 1 & 0 & 3 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 2 \\ 0 & 0.5 & 0 & 3 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

After translation, $C = B1 * Translate(1, 1, 0)$

$$= \begin{bmatrix} 0 & 0 & 1 & 2 \\ 0 & 0.5 & 0 & 3 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 2 \\ 0 & 0.5 & 0 & 3.5 \\ -1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

After poping from stack,

$$\text{model-matrix} = B = \begin{bmatrix} 0 & 0 & 1 & 2 \\ 0 & 1 & 0 & 3 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

After scaling,

$D = B * Scale(2, 1, 1)$

$$= \begin{bmatrix} 0 & 0 & 1 & 2 \\ 0 & 1 & 0 & 3 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 2 \\ 0 & 1 & 0 & 3 \\ -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

6. According to the professor, the tilted line crosses $(0, -1)$
but θ angle is unknow.
We have to first translate $(-1, 0)$
then rotate $(-\theta)$
and do the scaling $(1 -1)$
and rotate back (θ)
translate back $(+1, 0)$

Composite all of steps above using 2D matrix, (with reverse order)

$$M = \begin{bmatrix} 1 & 0 & +1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

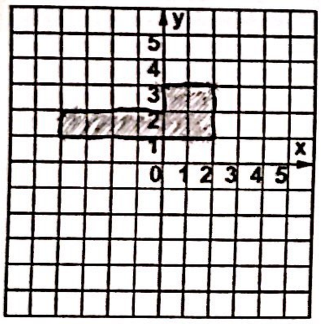
$$\text{thus } M = \begin{bmatrix} \cos2\theta & \sin2\theta & \overset{=1-\cos2\theta}{-2\sin^2\theta} \\ \sin2\theta & -\cos2\theta & -\sin2\theta \\ 0 & 0 & 1 \end{bmatrix}$$

Using similar code to problem5, and suppose theta is a given variable.

```
modelM.setAsIdentity();
modelM = modelM * Translate(1,0,0);
modelM = modelM * RotateZ(theta);
modelM = modelM * Scale(1,-1,1); //(Z is 0)
modelM = modelM * RotateZ(-theta);
modelM = modelM * Translate(-1,0,0);
```

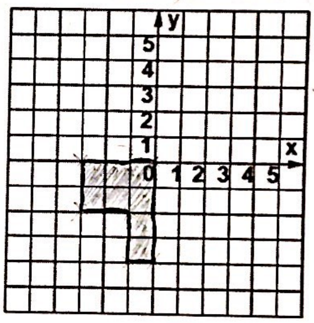
7.

a) L' = ABC L



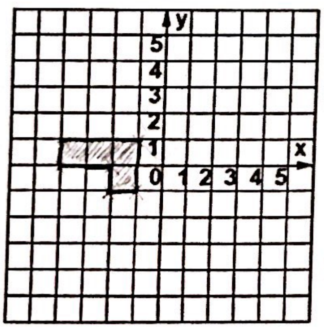

```
modelMatrix.setAsIdentify();
modelMatrix=modelMatrix *
    Scale(2,1,1);
modelMatrix=modelMatrix *
    Translate(1,1,0);
modelMatrix = modelMatrix *
    RotateZ(90);
drawL();
```

b) L' = CAD L



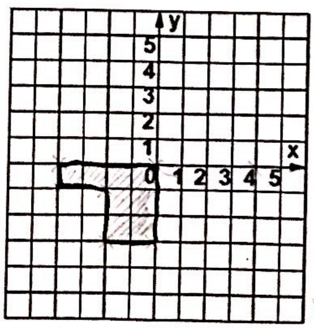

```
modelMatrix.setAsIdentity();
modelMatrix = modelMatrix *
    RotateZ(90);
modelMatrix= modelMatrix *
    Scale(2,1,1);
modelMatrix= modelMatrix *
    Scale(-1,1,1);
drawL();
```

c) L' = CBD L




```
modelMatrix.setAsIadentity();
modelMatrix =modelMatrix
    * RotateZ(90);
modelMatrix = modelMatrix
    * Translate(1,1,0);
modelMatrix = modelMatrix
    * Scale(-1,1,1);
drawL();
```

d) L' = DCCAD L




```
modelMatrix.setAsIdentity();
modelMatrix =modelMatrix *
    Scale(-1,1,1);
modelMatrix= modelMatrix *
    RotateZ(90);
modelMatrix= modelMatrix *
    RotateZ(90);
modelMatrix= modelMatrix *
    Scale(2,1,1);
modelMatrix = modelMatrix *
    Scale(-1,1,1);
drawL();
```