1. (21 pts) The point coordinate P can be expressed as $P = 3i + 0j$, where $i$ and $j$ are basis vectors of unit length along the x and y axes, respectively. Describe the point P in terms of the 3 other coordinate systems given below.

   ![Diagram]

   a) $P = (-2, 2)$ b) $P = (3, 0.5)$ c) $P = (2, -1)$

2. (3 pts) Write down the 4x4 matrix for scaling an object by 2 in y and 3 in Z.

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

3. (10 pts) Give the OpenGL commands required to encode $M$. You may assume the matrix stack has been initialized with `glIdentity()`.

   $$\begin{bmatrix}
   1 & 0 & 0 & 1 \\
   0 & 1 & 0 & 1 \\
   0 & 0 & 2 & 1 \\
   0 & 0 & 0 & 1 \\
   \end{bmatrix}$$

   `glTranslatef(1.0, 1.0, 1.0); glScalef(1.0, 1.0, 2.0);`

4. (6 pts) Homogenize the point (2,10,8,4).

   $(.5, 2.5, 2, 1)$
5. (20 pts) Give the 4x4 OpenGL modelview matrix at the four lines A, B, C, and D below.

```c
glLoadIdentity();
glTranslatef(2,3,0);
A
glRotate(90, 0,1,0);
B
glPushMatrix();
glScale(1,.5,1);
glTranslatef(1,1,0);
C
glPopMatrix();
glScale(2,1,1);
D
```

A

```
| 1 0 0 2 |
| 0 1 0 3 |
| 0 0 1 0 |
| 0 0 0 1 |
```

B

```
| 1 0 0 2 | | 0 0 1 0 | | 0 0 1 2 |
| 0 1 0 3 | * | 0 1 0 0 | = | 0 1 0 3 |
| 0 0 1 0 | | -1 0 0 0 | | -1 0 0 0 |
| 0 0 0 1 | | 0 0 0 1 | | 0 0 0 1 |
```

C

```
| 0 0 1 2 | | 1 0 0 0 | | 1 0 0 1 | | 0 0 1 2 |
| 0 1 0 3 | * | 0 0.5 0 0 | * | 0 1 0 1 | = | 0 0.5 0 3.5 |
| -1 0 0 0 | | 0 0 1 0 | | 0 0 1 0 | | -1 0 0 -1 |
| 0 0 0 1 | | 0 0 0 1 | | 0 0 0 1 | | 0 0 0 1 |
```

D (multiply by matrix from line B)

```
| 0 0 1 2 | | 2 0 0 0 | | 0 0 1 2 |
| 0 1 0 3 | * | 0 1 0 0 | = | 0 1 0 3 |
| -1 0 0 0 | | 0 0 1 0 | | -2 0 0 0 |
| 0 0 0 1 | | 0 0 0 1 | | 0 0 0 1 |
```

6. (40 pts) For each equation below, sketch the new location L' of the L shape on the grid and provide the OpenGL sequence needed to carry out those operations. Use the function `drawL()`. which draws an L shape with the lower left corner at the current origin as shown below. You may assume the matrix mode is `GL_MODELVIEW` and that the stack has been initialized with `glLoadIdentity()`. For reference, the OpenGL command syntax is `glTranslatef(angle, x, y, z), glScalef(x, y, z)`, `glRotatef(angle, x, y, z)`.

```c
drawL();
```
Observe that...
A -> translate by 1 along X axis -> `glTranslate[<d|f>]( 1, 0, 0 )`
B -> rotate by 3/2 pi about Z axis -> `glRotate[<d|f>]( 3 / 2 * PI, 0, 0, 1 )`
C -> translate by -1 along X axis -> `glTranslate[<d|f>]( -1, 0, 0 )`
D -> scale by 2 along X axis -> `glScale[<d|f>]( 2, 1, 1 )`

a)
```latex
\begin{align*}
glTranslate[<d|f>]( 1, 0, 0 ); \\
glRotate[<d|f>]( 3 / 2 \times PI, 0, 0, 1 ); \\
glTranslate[<d|f>]( -1, 0, 0 ); \\
drawL();
\end{align*}
```

b)
```latex
\begin{align*}
glTranslate[<d|f>]( 1, 0, 0 ); \\
glTranslate[<d|f>]( -1, 0, 0 ); \\
glScale[<d|f>]( 2, 1, 1 ); \\
drawL();
\end{align*}
```

c)
```latex
\begin{align*}
glRotate[<d|f>]( 3 / 2 \times PI, 0, 0, 1 ); \\
glTranslate[<d|f>]( 1, 0, 0 ); \\
glRotate[<d|f>]( 3 / 2 \times PI, 0, 0, 1 ); \\
drawL();
\end{align*}
```

d)
```latex
\begin{align*}
glTranslate[<d|f>]( -1, 0, 0 ); \\
glRotate[<d|f>]( 3 / 2 \times PI, 0, 0, 1 ); \\
glTranslate[<d|f>]( 1, 0, 0 ); \\
glScale[<d|f>]( 2, 1, 1 ); \\
drawL();
\end{align*}
```

a) $L' = ABC L$

![Diagram a)](image-a)

b) $L' = ACD L$

![Diagram b)](image-b)

c) $L' = BAB L$

![Diagram c)](image-c)

d) $L' = CBAD L$

![Diagram d)](image-d)