Forward Kinematics:
You Start with Separate Pieces, all Defined in their Own Local Coordinate System

Hook the Pieces Together, Change Parameters, Things Move
(All Children Understand This)

Forward Kinematics: Where do the Pieces Move To?
Locations?

Forward Kinematics: Positioning Part #1 With Respect to Ground

1. Rotate by $\Theta_1$
2. Translate by $T_{1G}$

Write it
\[
\begin{bmatrix} M_{1/G} \end{bmatrix} = \begin{bmatrix} T_{1/G} \end{bmatrix} * \begin{bmatrix} R_{\Theta_1} \end{bmatrix}
\]

Say it

Why Do We Say it Right-to-Left?

It's because in the matrix notes, we adopted the convention that the coordinates are multiplied on the right side of the matrix:

So the right-most transformation in the sequence multiplies the (x,y,z,1) first and the left-most transformation multiplies it last.
Positioning Part #2 With Respect to Ground

1. Rotate by $\Theta_2$
2. Translate the length of part 1
3. Rotate by $\Theta_1$
4. Translate by $T_{1/G}$

$$[\mathbf{M}_{2/G}] = [\mathbf{T}_{1/G}] \cdot [\mathbf{R}_{\Theta_1}] \cdot [\mathbf{T}_{2/G}] \cdot [\mathbf{R}_{\Theta_2}]$$

Positioning Part #3 With Respect to Ground

1. Rotate by $\Theta_3$
2. Translate the length of part 2
3. Rotate by $\Theta_2$
4. Translate the length of part 1
5. Rotate by $\Theta_1$
6. Translate by $T_{1/G}$

$$[\mathbf{M}_{3/G}] = [\mathbf{T}_{1/G}] \cdot [\mathbf{R}_{\Theta_1}] \cdot [\mathbf{T}_{2/G}] \cdot [\mathbf{R}_{\Theta_2}] \cdot [\mathbf{T}_{3/G}] \cdot [\mathbf{R}_{\Theta_3}]$$

Sample Program

```c
DrawLinkOne( )
{
    glColor3f( 1., 0., 0. ); // red, green blue
    glBegin( GL_QUADS );
    glVertex2f(    -BUTT, -THICKNESS/2 );
    glVertex2f( LENGTH_1, -THICKNESS/2 );
    glVertex2f( LENGTH_1,  THICKNESS/2 );
    glVertex2f(    -BUTT,  THICKNESS/2 );
    glEnd( );
}
```

Sample Program, using OpenGL's Automatic Transformation Concatenation

```c
Sample Program

DrawMechanism( float $\Theta_1$, float $\Theta_2$, float $\Theta_3$ )
{
    glPushMatrix( );
    glRotatef( $\Theta_1$, 0., 0., 1. );
    glColor3f( 1., 0., 0. );
    DrawLinkOne( );
    glTranslatef( LENGTH_1, 0., 0. );
    glRotatef( $\Theta_2$, 0., 0., 1. );
    glColor3f( 0., 1., 0. );
    DrawLinkTwo( );
    glTranslatef( LENGTH_2, 0., 0. );
    glRotatef( $\Theta_3$, 0., 0., 1. );
    glColor3f( 0., 0., 1. );
    glPopMatrix( );
    DrawLinkThree( );
}
```

Sample Program

```c
glViewport( 100, 100,   500, 500 );
glMatrixMode( GL_PROJECTION );
glLoadIdentity( );
gluPerspective( 90., 1.0, 1., 10. );
```

Whatever interaction is being used to get the eye position

```c
glMatrixMode( GL_MODELVIEW );
gluLookAt( eyex, eyey, eyez,
centerx, centery, centerz,
upx, upy, upz );
```

Set the eye position

```c
[\mathbf{D}_{\text{light}}] = [\mathbf{T}_{\text{light}}] \cdot [\mathbf{R}_{\text{light}}]$

[\mathbf{D}_{\text{camera}}] = [\mathbf{T}_{\text{camera}}] \cdot [\mathbf{R}_{\text{camera}}] \cdot [\mathbf{T}_{\text{light}}] \cdot [\mathbf{R}_{\text{light}}]$

Where in the window to display (pixels)

Viewing Info:

Field of view angle, x:y aspect ratio, near, far

```c
done = FALSE;
while( ! done )
{
    << Determine $\Theta_1$, $\Theta_2$, $\Theta_3$ >>
    glPushMatrix();
    glLoadIdentity( );
    glRotatef();
    DrawMechanism( $\Theta_1$, $\Theta_2$, $\Theta_3$ );
    glPopMatrix();
}```
In Project #4, you won't be able to do this. You will need to create each full matrix separately using your own Matrix class methods.