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?

Positioning Part #1 With Respect to Ground

1. Rotate by $\theta_1$
2. Translate by $T_{1G}$

Write it

$$[M_{1/G}] = [T_{1/G}] * [R_{\theta_1}]$$

Say it

Forward Kinematics: 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:

$$[x'] = [R_{\theta_1}] * [T_{1/G}] * [M_{1/G}]$$

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 Θ₂
2. Translate the length of part 1
3. Rotate by Θ₁
4. Translate by \( T_{1G} \)

\[
\begin{align*}
[M_{2/G}] &= [T_{1G}] \cdot [R_{\theta_1}] \cdot [T_{21}] \cdot [R_{\theta_2}] \\
[M_{2/G}] &= [M_{1/G}] \cdot [M_{21}] \\
\end{align*}
\]

Sample Program

```
Sample Program
```

1. Rotate by Θ₃
2. Translate the length of part 2
3. Rotate by Θ₂
4. Translate the length of part 1
5. Rotate by Θ₁
6. Translate by \( T_{1G} \)

\[
\begin{align*}
[M_{3/G}] &= [T_{1G}] \cdot [R_{\theta_1}] \cdot [T_{21}] \cdot [R_{\theta_2}] \cdot [T_{32}] \cdot [R_{\theta_3}] \\
[M_{3/G}] &= [M_{1/G}] \cdot [M_{21}] \cdot [M_{32}] \\
\end{align*}
\]

Sample Program, using OpenGL's Automatic Transformation Concatenation

```
Sample Program
```

```
Sample Program
```

```
Sample Program
```

Where in the window to display (pixels)

Viewing Info:
- Field of view: angle, x:y aspect ratio, near, far
- Whatever interaction is being used to get the eye position
- DrawMechanism( \( \theta_1, \theta_2, \theta_3 \) );
- \( \text{glViewport} (100, 100, 500, 500) \);
- \( \text{glMatrixMode} (\text{GL_PROJECTION}) \);
- \( \text{glLoadIdentity} () \);
- \( \text{gluPerspective} (90., 1.0, 1., 10.) \);
- \( \text{glMatrixMode} (\text{GL_MODELVIEW}) \);
- \( \text{glLoadIdentity} () \);
- \( \text{done} = \text{FALSE} \);
- \( \text{while} (! \text{done}) \) {
  \( \text{\langle \langle \text{Determine} \theta_1, \theta_2, \theta_3 \rangle \rangle} \)
  \( \text{glMatrixMode} (\text{GL_MODELVIEW}) \);
  \( \text{glLoadIdentity} () \);
  \( \text{gluLookAt} (\text{eyex}, \text{eyey}, \text{eyez}, \text{centerx}, \text{centery}, \text{centerz}, \upx, \upy, \upz) \);
  \( \text{glColor3f} (1., 0., 0.) \); \( // \text{red, green blue} \)
  \( \text{glBegin} (\text{GL_QUADS}) \);
  \( \text{glVertex2f} (-\text{BUTT}, -\text{THICKNESS}/2) \);
  \( \text{glVertex2f} (\text{LENGTH}_1, -\text{THICKNESS}/2) \);
  \( \text{glVertex2f} (\text{LENGTH}_1, \text{THICKNESS}/2) \);
  \( \text{glVertex2f} (-\text{BUTT}, \text{THICKNESS}/2) \);
  \( \text{glEnd} () \);
  \( \text{glPushMatrix} () \);
  \( \text{glTranslatef} (\text{LENGTH}_2, 0., 0.) \);
  \( \text{glRotatef} (\theta_2, 0., 0., 1.) \);
  \( \text{glColor3f} (0., 1., 0.) \);
  \( \text{DrawLinkTwo} () \);
  \( \text{glPopMatrix} () \);
\}
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.