Now let's do two simple projections:

Orthogonal projection onto the xy plane:

$$x' = x \text{ and } y' = y \text{ and } z' = 0$$

really simple, eh?

The OpenGL default is equivalent to:

```
glMatrixMode(GL_PROJECTION);
glLoadIdentity( );
glOrtho( xwmin, xwmax, ywmin, ywmax, dnear, dfar );
```

with

```
xwmin = -1.0;   xwmax = 1.0; //x-coods of clipping window
ywmin = -1.0;   ywmax = 1.0; //y-coods of clipping window
dnear = -1.0;   dfar  = 1.0; //NEAR and FAR DISTANCES..
//NEAR clipping plane is z=(-dnear)= 1.0
//FAR clipping plane is z=(-dfar)= -1.0
```

```
gluOrtho2D(xwm, xwM, ywm, ywM);
//is equivalent to:
glOrtho( xwm, xwM, ywm, ywM, -1.0,  1.0 );
```

Let's implement `glOrtho` — Use the math of your HW3 programming exercise:

Given (x, y, z) (viewing coordinates), how should say x be transformed?

$$x \to \left(x - \frac{\text{xwmin} + \text{xwmax}}{2}\right)$$

is the signed distance of x from middle of the desired view volume.

$$\left(x - \frac{\text{xwmin} + \text{xwmax}}{2}\right) \to 2 \cdot \frac{\left(x - \frac{\text{xwmin}+\text{xwmax}}{2}\right)}{\text{xwmax} - \text{xwmin}}$$

normalizes it to the $[-1, +1]$ range (the length of this range is 2.0.)

Ditto for Y. BUT for Z, use `-dnear` and `-dfar`

So, larger z-buffer values correspond to larger distances from the viewer.

From the RedBook, **glOrtho**(l, r, t, n, f) generates

$$\begin{bmatrix} \frac{2}{r-l} & 0 & 0 & \frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & \frac{t+b}{t-b} \\ 0 & 0 & \frac{-2}{f-n} & \frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

when $n = -1$ and $f = +1$, the lower-right-hand 2×2 matrix

$$\begin{bmatrix} \frac{-2}{f-n} & \frac{f+n}{f-n} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} z \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{-2}{1-(-1)} & \frac{0}{1-(-1)} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} z \\ 1 \end{bmatrix} = \begin{bmatrix} -z \\ 1 \end{bmatrix}$$

next, when the (x', y', z', h') from the above used to transform $(x, y, z, 1)$ is perspectively divided at last, the result is

$$\frac{-z}{1} = -z$$

So, points in the near clipping plane $z = +1$ get transformed to normalized device coordinates with $z_{\text{ndc}} = -1$ (smallest Z value.)