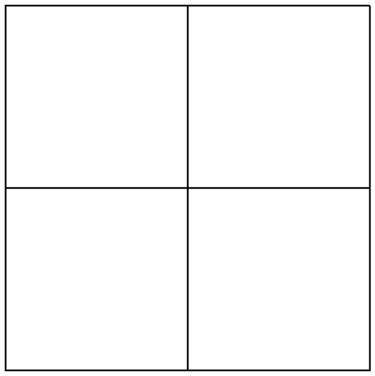
1. OpenGL has ten types of geometric primitives. Give the OpenGL name and a brief description of five geometric primitives.

2. Draw a sketch of what the following code would draw (using the `gluOrtho2D` two dimensional viewing projection). In your sketch, label the appropriate points in the order p0 to p9

```
glBegin(GL_TRIANGLE_FAN);
   glVertex2i( 0,  0);  // p0
   glVertex2i( 0,  1);
   glVertex2i( 1,  1);
   glVertex2i( 1,  0);
   glVertex2i( 1, -1);
   glVertex2i( 0, -1);
   glVertex2i(-1, -1);
   glVertex2i(-1,  0);
   glVertex2i(-1,  1);
   glVertex2i( 0,  1);  // p9
glEnd();
```

3. Use appropriate OpenGL primitives to draw the following figure in such a way that you send at most eight vertices into the graphics pipeline (if you send the same vertex twice into the pipeline, that counts as two vertices). You can assume that this figure is centered at the origin and the sides have length two.



4. Give one important reason for using homogeneous coordinates in a graphics system.

5. When is it preferable to use orthographic viewing transformations? When is it preferable to use perspective transformations? Explain why.

6. (a) Compute the 16 number that make up the 4×4 homogeneous matrix that OpenGL would compute after executing the follow lines of code.

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(3.0, 0.0, 0.0);
glScalef(3.0, 1.0, 1.0);
```

(b) Compute the 16 number that make up the 4×4 homogeneous matrix that OpenGL would compute after executing the follow lines of code.

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glScalef(3.0, 1.0, 1.0);
glTranslatef(3.0, 0.0, 0.0);
```

(c) Compute the 16 number that make up the 4×4 homogeneous matrix that OpenGL would compute after executing the follow lines of code.

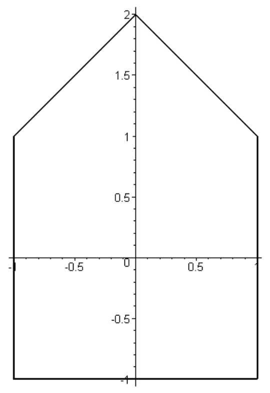
```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glScalef(3.0, 1.0, 1.0);
glTranslatef(1.0, 0.0, 0.0);
```

(d) How would you explain to someone, without doing the actual calculations, why the matrices in parts (a) and (b) are different, but the matrices in parts (a) and (c) are the same?

7. What would the following code draw in a two-dimensional world? Assume that the function `drawUnitCircle()` draws a circle with radius 1 centered at the origin of the coordinate system.

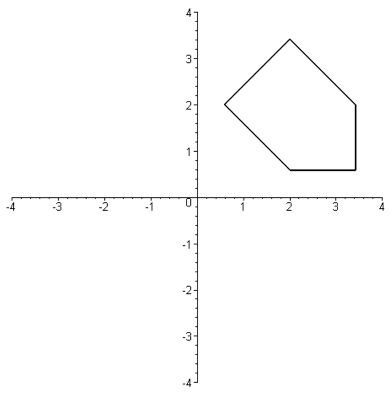
```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glPushMatrix();
   glScalef(2.0, 2.0, 1.0);
   drawUnitCircle();
glPopMatrix
glPushMatrix();
  glRotatef(theta1, 0, 0, 1);
  glTranslatef(2.0, 0.0, 0.0);
  glScalef(0.5, 0.5, 1.0);
  drawUnitCircle();
glPopMatrix();
glPushMatrix();
  glRotatef(theta2, 0, 0, 1);
  glTranslatef(3.0, 0.0, 0.0);
  drawUnitCircle();
glPopMatrix();
glScalef(0.25, 0.25, 1.0);
drawUnitCircle();
```

8. Suppose we have defined a function `drawModel()` that pushes the appropriate geometric primitives into the OpenGL pipeline to draw the following image (using the `gluOrtho2D()` two dimensional viewing projection). Draw a sketch of the scene that the following code would produce. For each of the three model images, give the coordinates of the model's "origin" point.



```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(1.0, 0.0, 0.0);
glPushMatrix();
  glTranslatef(0.0, 3.0, 0.0);
  glRotatef(45.0, 0, 0, 1);
  drawModel();
glPopMatrix();
glTranslatef(1.0, 0.0, 0.0);
drawModel();
glLoadIdentity();
glTranslatef(-2.0, -2.0, 0.0);
glRotatef(-90.0, 0, 0, 1);
drawModel();
```

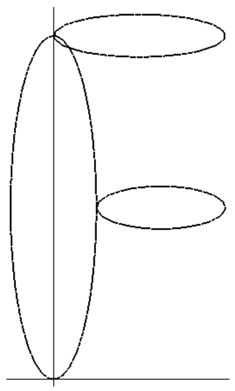
9. Using the same `drawModel()` function from the previous problem, write OpenGL code that would generate the following image. Do this two different ways, one way with a translation preceding a rotation, and the second way with a rotation preceding a translation.



(a)

(b)

10. Suppose you have available to you a function `drawUnitCircle()` that draws a circle of radius one centered at the origin. Use this function, and any of the OpenGL functions `glLoadIdentity()`, `glTranslatef()`, `glRotatef()`, `glScalef()`, `glPushMatrix()`, and `glPopMatrix()`, to draw the following "F-shape". The vertical ellipse has its bottom at the point $(0, 0)$ and its top at the point $(0, 4)$, and at its widest, it is 1 unit across. The upper horizontal ellipse is 2 units long. The lower horizontal ellipse is 1.5 units long. Its right hand end point has coordinates $(2, 2)$. The two horizontal ellipses are 0.25 units tall.



```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
```

11. Demonstrate, using algebra or a picture, that order matters when composing two transformations.

12. Suppose you have a program window that is 600 pixels horizontally and 400 pixels vertically. Suppose you want to render the part of the (2D) world coordinate system with $0 \le x \le 5$ and $-1 \le y \le 1$ in a viewport that does not distort the contents of the world window. The viewport should be centered within the right half of the program window, and the viewport should be as large as it can be, given these conditions. Write the lines of OpenGL code needed to do this. Your code should include appropriate `gluOrtho2D()` and `glViewport()` calls.

13. Describe what the following combination of `display()` and `reshape()` functions will draw in the program's screen window. How will the drawing change when the user resizes the screen window? Explain why.

```
void display()
{
   glClear(GL_COLOR_BUFFER_BIT);  /* clear the window */
   glColor3f(0.0, 0.0, 0.0);      /* draw in black */

   glMatrixMode(GL_MODELVIEW);
   glLoadIdentity();

   glScalef(100.0, 100.0, 1.0);
   drawUnitCircle();
   glFlush();
}


void reshape(int width, int height)
{
   glMatrixMode(GL_PROJECTION);
   glLoadIdentity();
   gluOrtho2D(0, width, 0, height); /* left, right, bottom, top */

   glViewport(0, height/2, width, height/2); /* left, bottom, width, height */
   display();
}
```

14. A real-time graphics program can use a single frame buffer by rendering polygons, clearing the buffer, and repeating. Why do we usually use two frame buffers instead?