In most programming environments, the origin (0,0) is at the upper left corner of the window! This may seem confusing at first but it actually makes our coordinate math simpler:

- we don’t have to worry about negatives
- there’s always a fixed point when we stretch a window from the bottom right corner

**Draw and label** the points (0,0), (5,5), (3,6) and (6,3) on the Processing axes below:

To get Processing to draw a shape, we use the name of the shape followed by some information about the specific shape we want. Here is the basic structure of a Processing function call (a command):

```
functionName(value1, value2, value3, value4);
```

- The function name tells Processing which command to execute. It can’t contain spaces.
- Parentheses surround a list of values that specify how the function is executed.
- A semicolon tells Processing that we’re done with our line of code.
Processing Drawing Challenges

1. The following Processing program contains several errors. Can you find and fix them all?
   
   ```java
   size 300, 300;
   rect(100 200 40 50);
   lines(100, 200, 140, 250)
   ```

2. Write the Processing code that would draw the image shown below. You may find it helpful to start by labeling important coordinates.

![Diagram of a geometric figure with labeled coordinates](image)

3. If you typed in the code you got for challenge 2 into Processing, you’d find that the picture would be very small. How could you make it bigger?

4. Start planning out your face program.
   - What kind of face is it? Animal? Robot? Zombie?
   - What kinds of accessories will it have? Hat? Bowtie?
   - What colors will you use?
   - What will be changed by sound (we’ll do it later, but plan it now)?
Functions for Drawing Shapes in Processing

```plaintext
line(startX, startY, endX, endY)
Draws a line from (startX, startY) to (endX, endY).
```
```
line(0, 0, 5, 5);  line(2, 2, 7, 7);  line(4, 7, 1, 2);
```
```
rect(startX, startY, width, height)
Draws a width x height rectangle with upper left corner at point (startX, startY).
```
```
rect(0, 0, 5, 5);  rect(2, 2, 5, 5);  rect(1, 2, 3, 5);
```
```
ellipse(centerX, centerY, width, height)
Draws an ellipse of the given width and height centered at point (centerX, centerY).
```
```
ellipse(2, 2, 4, 4);  ellipse(4, 3, 4, 4);  ellipse(2, 4, 2, 6);
```
```
triangle(x1, y1, x2, y2, x3, y3)
Draws a triangle between points (x1, y1), (x2, y2) and (x3, y3).
```
```
triangle(0, 0, 0, 5, 5, 5);  triangle(2, 2, 7, 7, 2, 7);  triangle(4, 2, 6, 5, 2, 5);
```
```
text(text, startX, startY)
Displays text with bottom left anchored at (startX, startY).
```
```
text("hello", 1, 2);
```
```
You may want to investigate `textSize` and `textFont`.
```
More Drawing Functions

size(width, height)
Sets the size of the drawing window to width, height.

noStroke()
All future shapes will be drawn without an outline.

strokeWeight(weight)
All future outlines will be drawn with the specified thickness.

noFill()
All future shapes will be drawn without being filled out.

smooth()
All future shapes will be drawn smoothly (may slow down drawing speed).

Functions for Using Color in Processing

In Processing, colors are represented by 3 numbers that specify the amount of red, green and blue to mix together. These numbers range from 0 - 255.

In Processing, colors mix like light. For example, 255 of red, 255 of green and 255 of blue makes white. Here are more examples:

255, 255, 0 - yellow
255, 0, 255 - magenta
88, 0, 88 - dark purple
0, 0, 0 - black

background(red, green, blue)
Changes the background color.

fill(red, green, blue)
Changes the current fill color. All future filled shapes will be of this color.

stroke(red, green, blue)
Changes the current outline color. All future outlined shapes will be of this color.

Transparency

The fill and stroke commands can also take in a 4th value between 0-255 to represent transparency. 255 means fully opaque (not transparent at all) and 0 is fully transparent (invisible!).