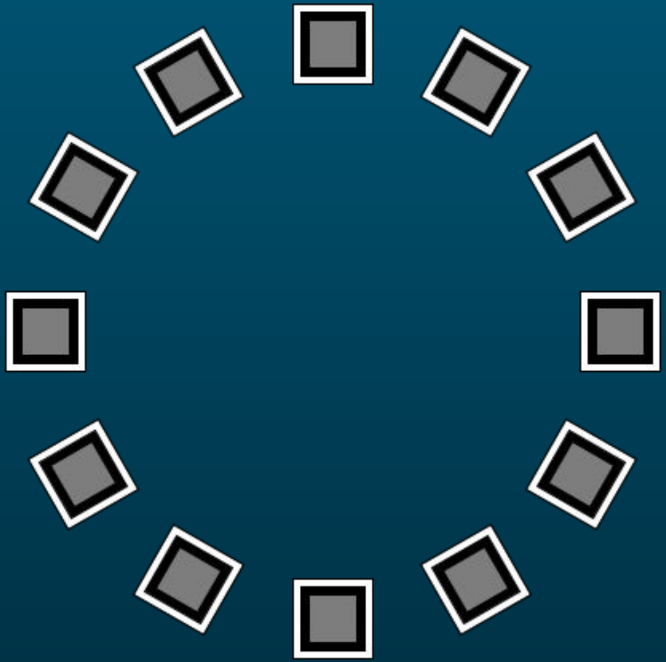
Week 06
Move, Rotate & SCALE !

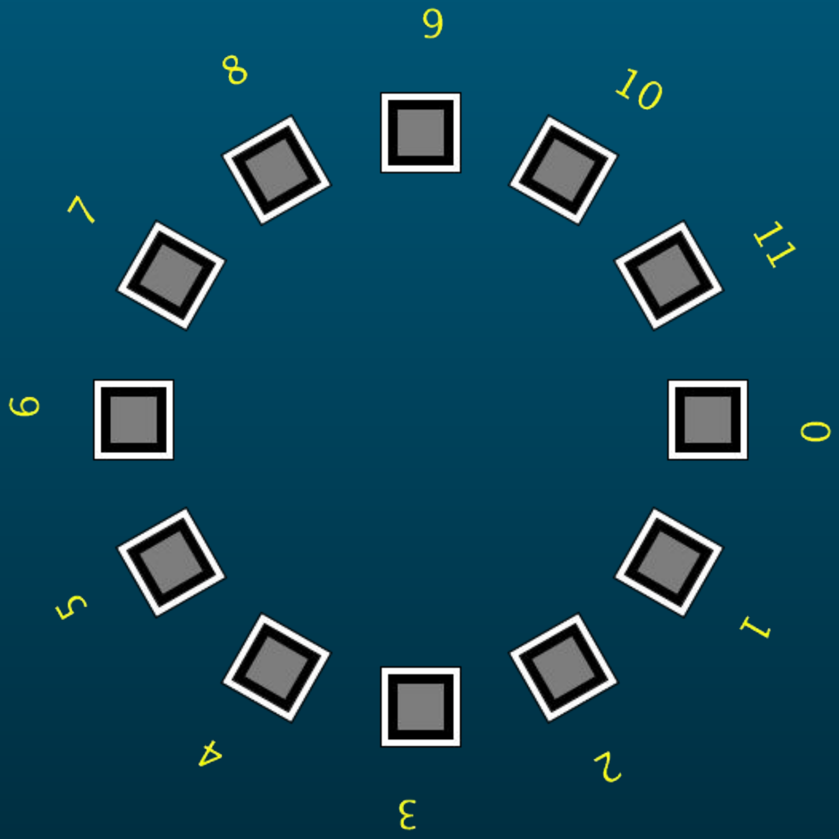
by Mike Wong. School of Creative Media

How to draw this ?

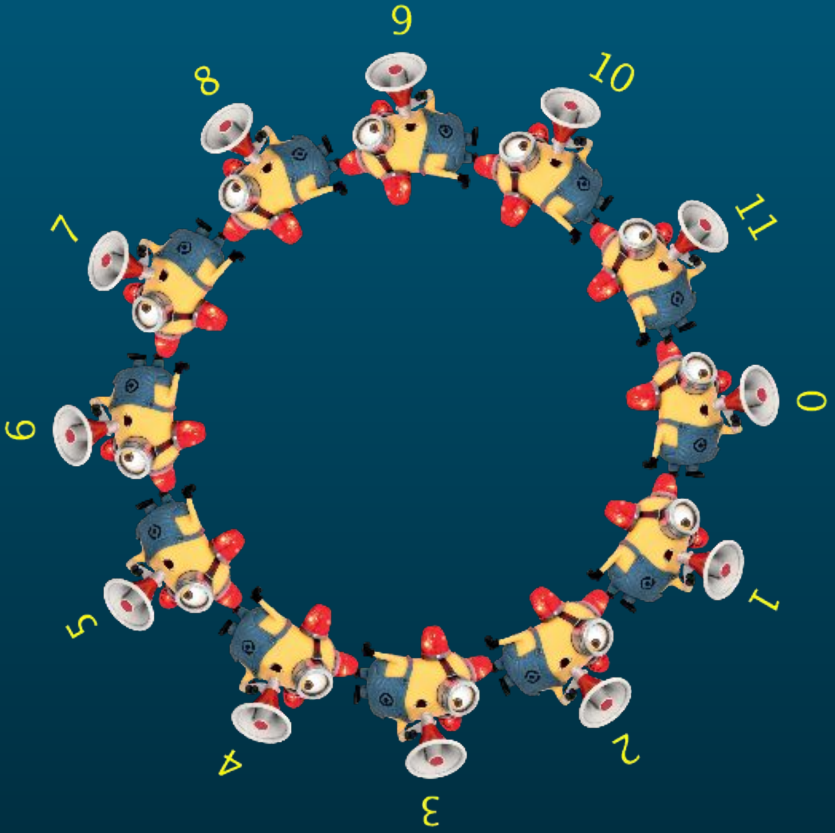


by Mike Wong. School of Creative Media

How to draw this ?

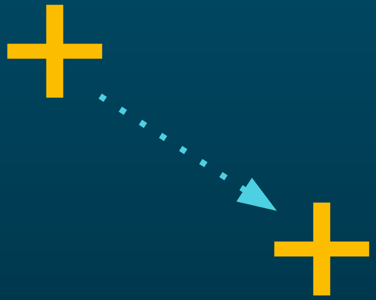


How to draw this ?

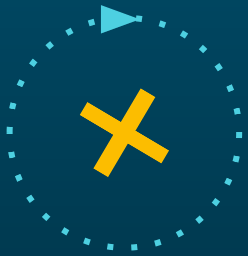


2D Transformations in Processing

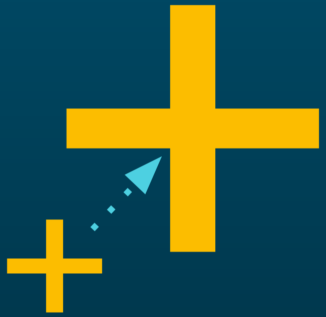
2D Transformations refer to THREE common operations used in drawing in computer graphics. They are:



Translate (Move)



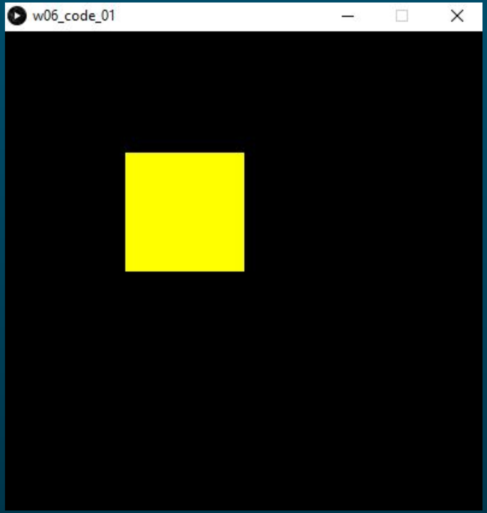
Rotate



Scale

(X,Y) Coordinate System

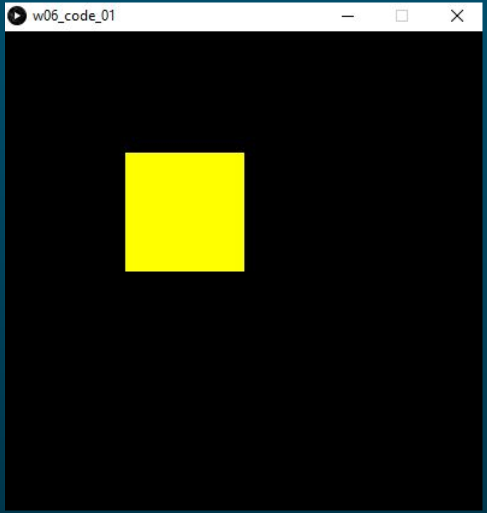
```
rect(100,100, 100,100);
```



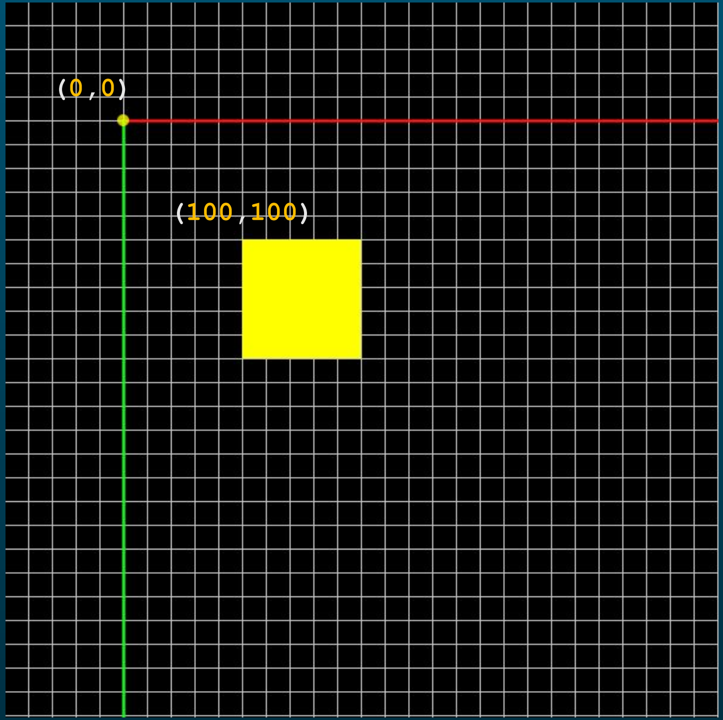
Our Processing Display

(X,Y) Coordinate System

```
rect(100,100, 100,100);
```



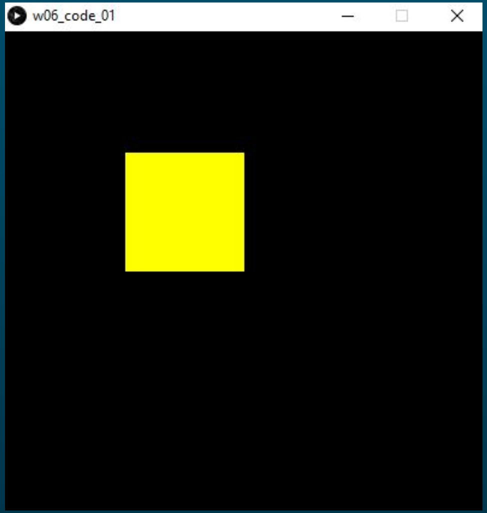
Our Processing Display



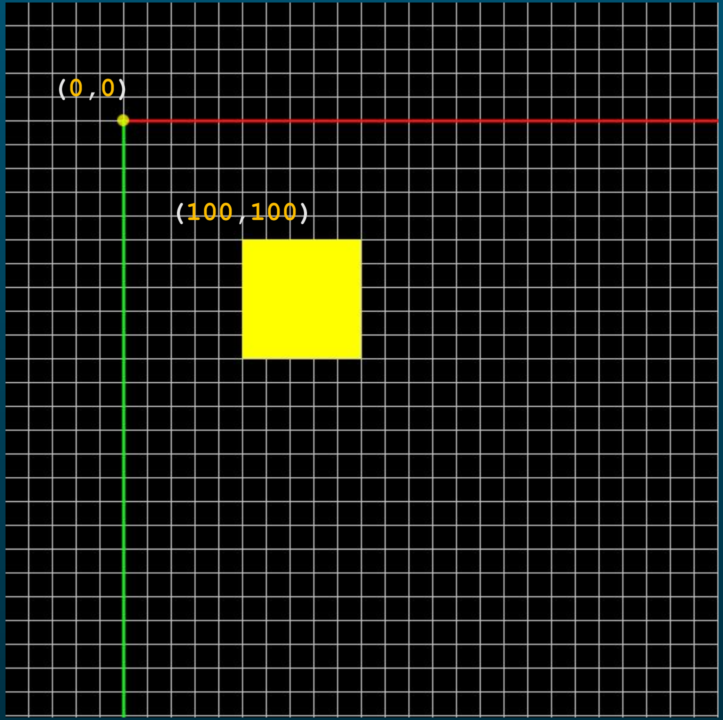
The underlying coordinate system

(X,Y) Coordinate System

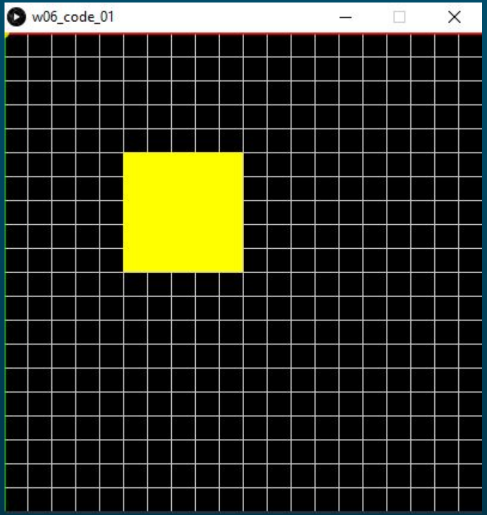
```
rect(100,100, 100,100);
```



Our Processing Display



The underlying coordinate system



Today, we will always visualize the underlying coordinate system to help us understand.

2D Transformations in Processing

Functions	Description
`translate(x,y)`	Moves the coordinate system to (x,y)
`rotate(angle)`	Rotates the coordinate system by `angle.` Please notice that this angle is measured in Radians. If one wants to supply the `angle` parameter in **degree**, we may use for example: `rotate(radians(60));`

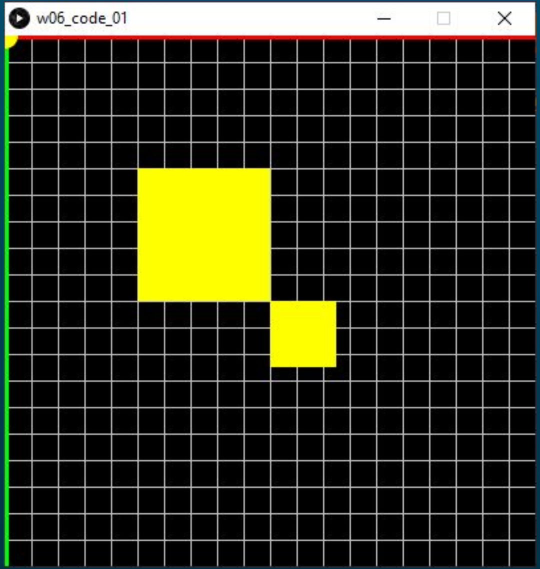
`scale(s) or`
`scale(sx,sy)`
	Scales everything uniformly by `s` or
	scales by `sx` and `sy` for the **X-axis** and **Y-axis** respectively.

`pushMatrix()`	**Save** the current **Transformation** state.
`popMatrix()`	**Reset** the **Transformation** to its **last saved state**.

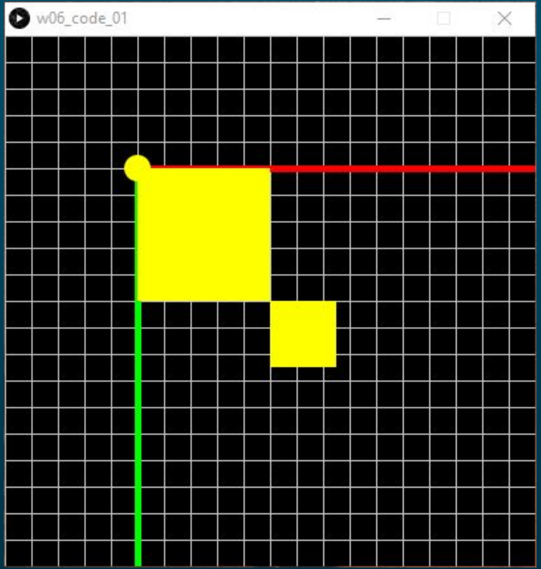
translate()

Example 1

translate(x,y) moves the coordinate system to (x,y).



```
rect(100,100, 100,100);
rect(200,200, 50,50);
```

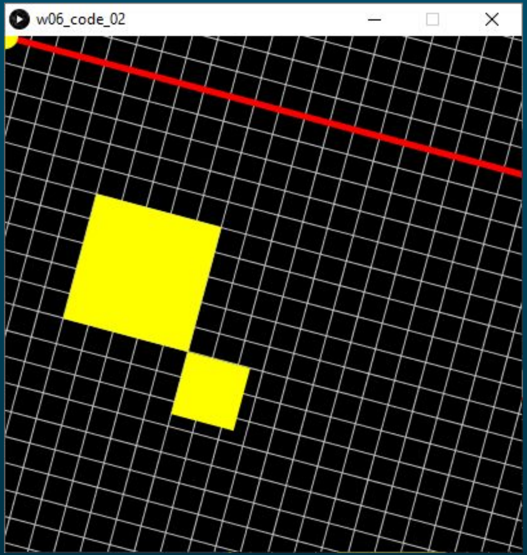


```
translate(100,100);
rect(0,0, 100,100);
rect(100,100, 50,50);
```

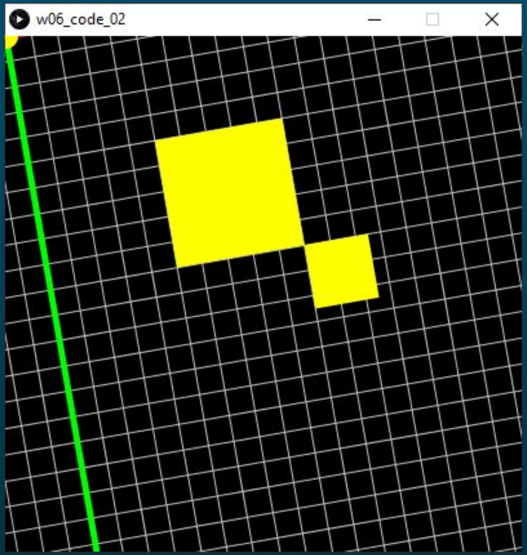
rotate()

Example 2

rotate(angle) rotates the coordinate system by **angle (in radians)**.



```
rotate(radians(15));
rect(100,100, 100,100);
rect(200,200, 50,50);
```

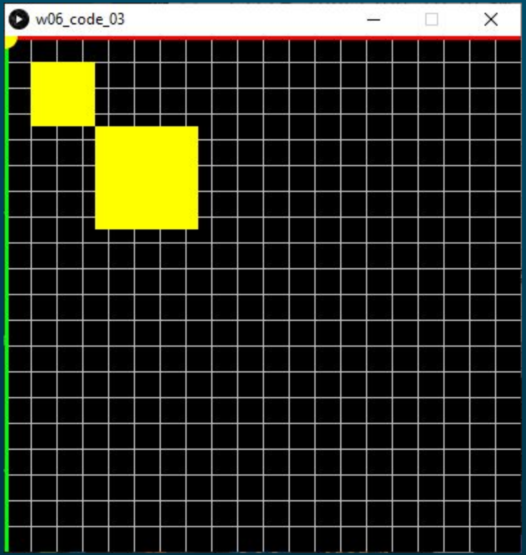


```
rotate(radians(-10));
rect(100,100, 100,100);
rect(200,200, 50,50);
```

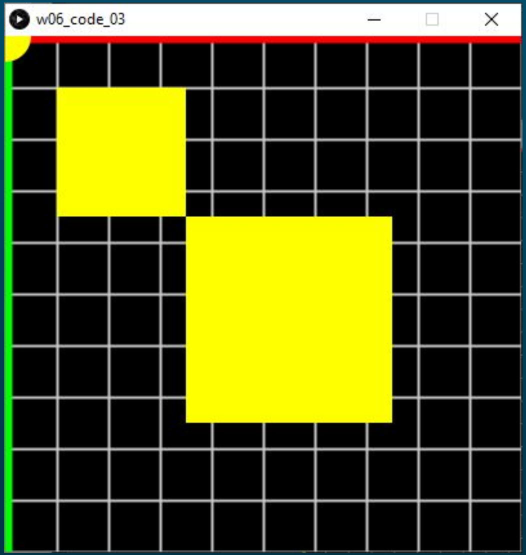
scale()

Example 3

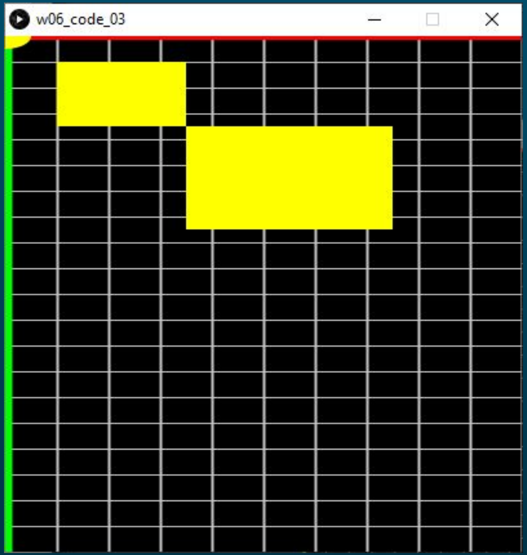
`scale(s)` scales the coordinate system by `s` or `(sx,sy)`.



```
rect(20,20, 50,50);
rect(70,70, 80,80);
```



```
scale(2.0);
rect(20,20, 50,50);
rect(70,70, 80,80);
```

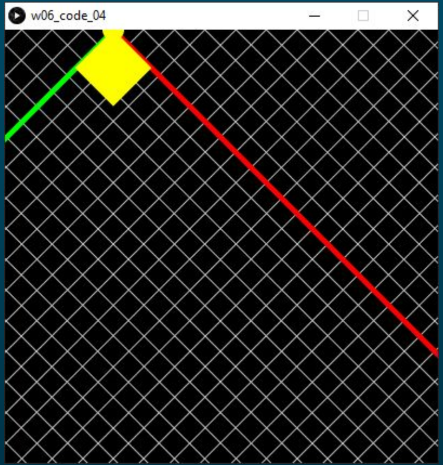


```
scale(2.0, 1.0);
rect(20,20, 50,50);
rect(70,70, 80,80);
```

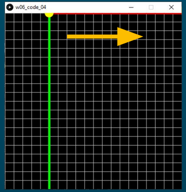
Multiple Transformations

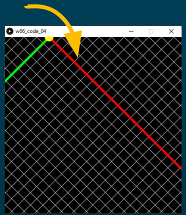
Example 4

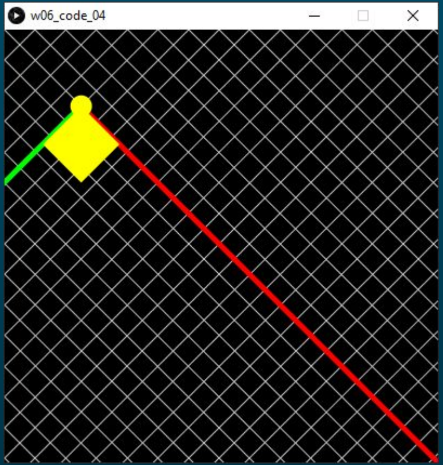
Multiple transformations may be applied **BUT**
Order of Transformation makes a HUGE DIFFERENCE.



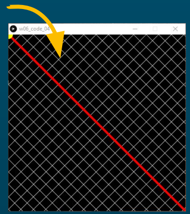
```
translate(100,0);
rotate(radians(45));
```

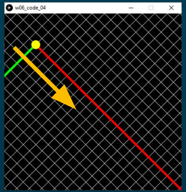

1. translate


2. rotate



```
rotate(radians(45));
translate(100,0);
```


1. rotate

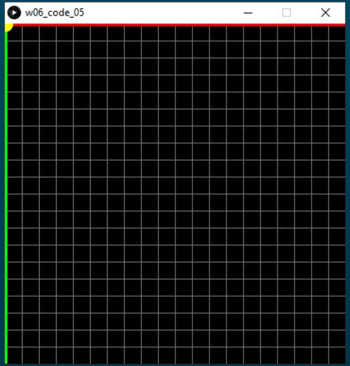

2. translate

Save & Restore Transformations

Example 5

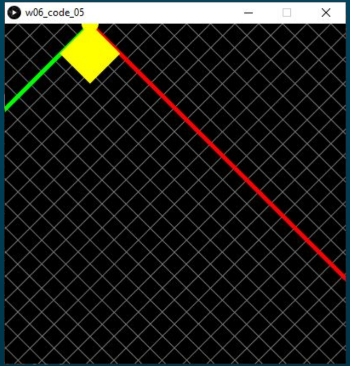
Transformation state may be saved and restored using `pushMatrix()` and `popMatrix()` respectively.

1. Save initial state



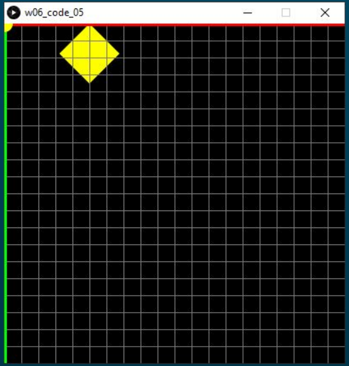
```
pushMatrix();
```

2. Apply transformation



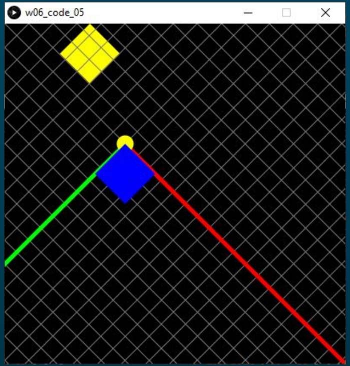
```
translate(100,0);
rotate(radians(45));
```

3. Restore initial state



```
popMatrix();
```

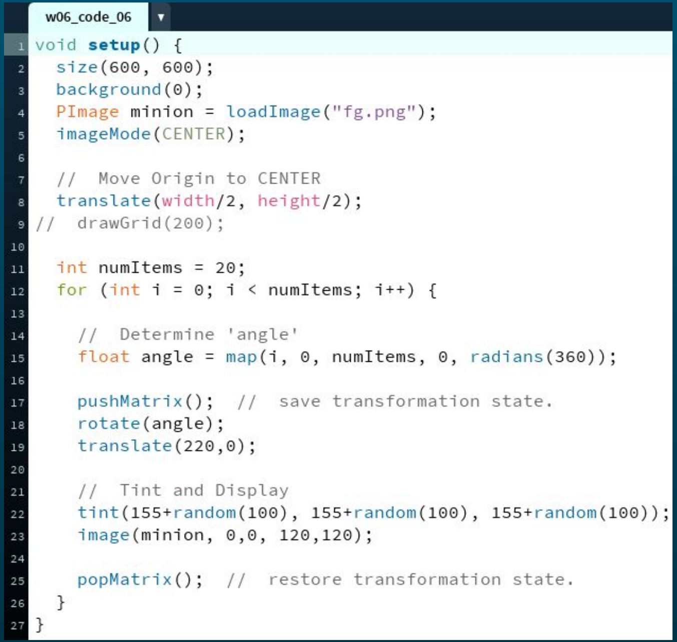
4. Apply transformation



```
rotate(radians(45));
translate(200,0);
```

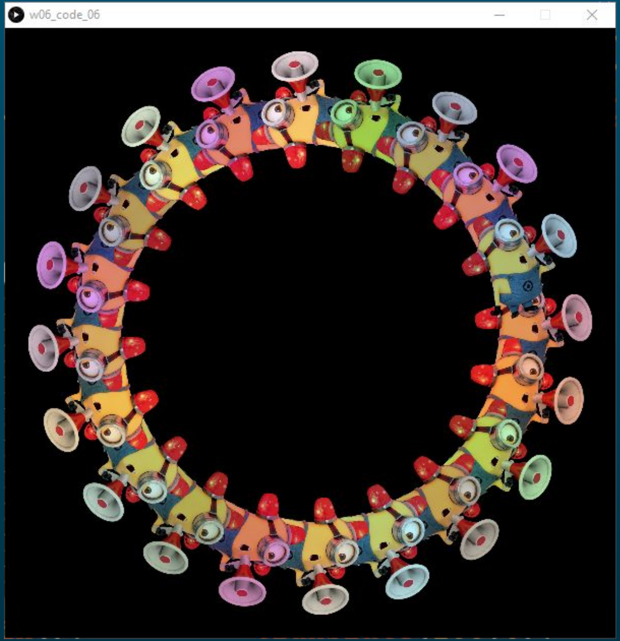
Example 6 - Ring of images

Example 6



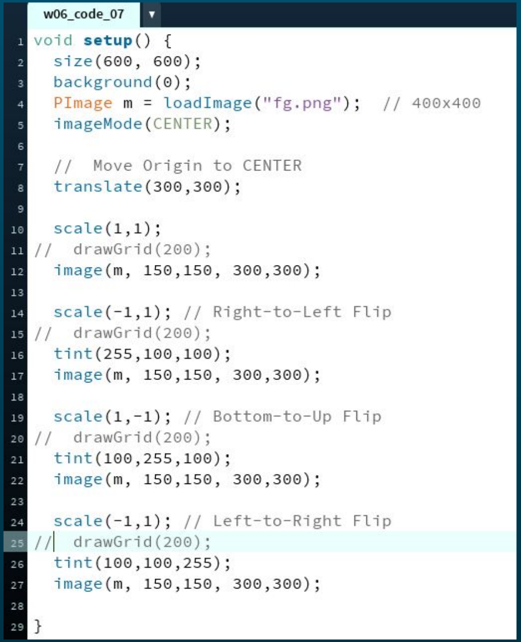
w06_code_06

```
void setup() {
  size(600, 600);
  background(0);
  PImage minion = loadImage("fg.png");
  imageMode(CENTER);

  //  Move Origin to CENTER
  translate(width/2, height/2);
//  drawGrid(200);

  int numItems = 20;
  for (int i = 0; i < numItems; i++) {

    //  Determine 'angle'
    float angle = map(i, 0, numItems, 0, radians(360));

    pushMatrix();  //  save transformation state.
    rotate(angle);
    translate(220,0);

    //  Tint and Display
    tint(155+random(100), 155+random(100), 155+random(100));
    image(minion, 0,0, 120,120);

    popMatrix();  //  restore transformation state.
  }
}
```

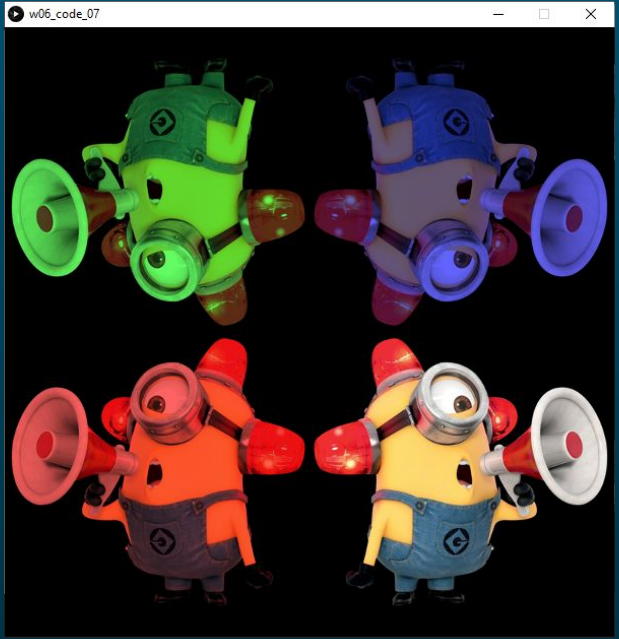


Example 7 - Reflections using scale()

Example 7

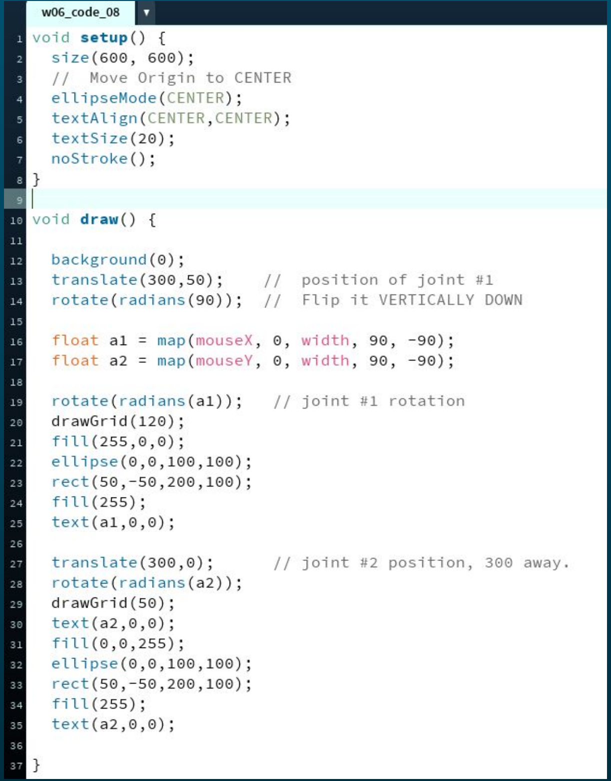


```
void setup() {
  size(600, 600);
  background(0);
  PImage m = loadImage("fg.png");  // 400x400
  imageMode(CENTER);

  //  Move Origin to CENTER
  translate(300,300);

  scale(1,1);
//  drawGrid(200);
  image(m, 150,150, 300,300);

  scale(-1,1); // Right-to-Left Flip
//  drawGrid(200);
  tint(255,100,100);
  image(m, 150,150, 300,300);

  scale(1,-1); // Bottom-to-Up Flip
//  drawGrid(200);
  tint(100,255,100);
  image(m, 150,150, 300,300);

  scale(-1,1); // Left-to-Right Flip
//  drawGrid(200);
  tint(100,100,255);
  image(m, 150,150, 300,300);

}
```

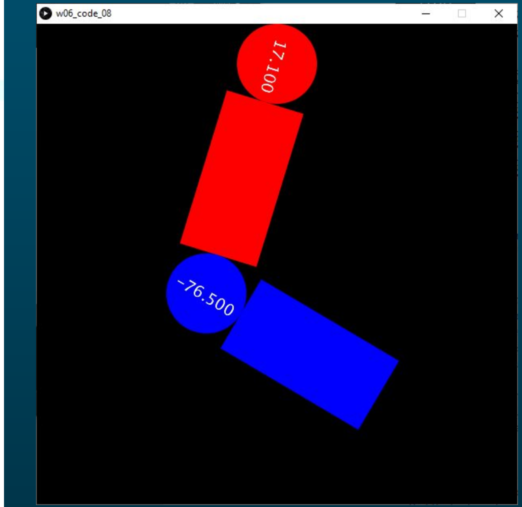


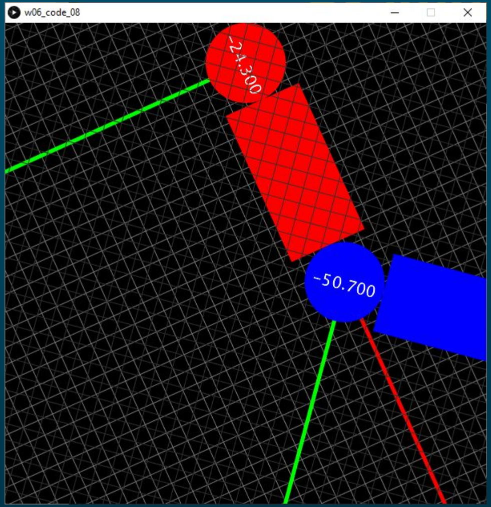
Example 8 - Hierarchy

Example 7

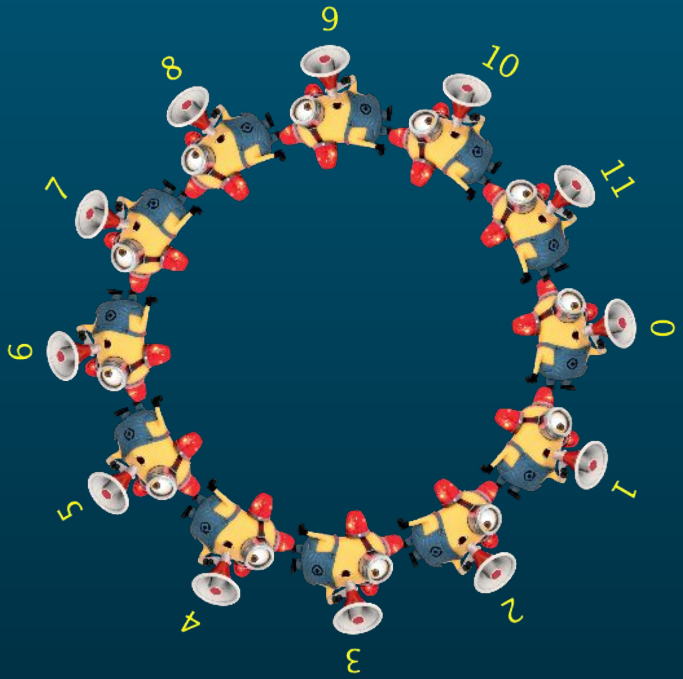


```
void setup() {
  size(600, 600);
  //  Move Origin to CENTER
  ellipseMode(CENTER);
  textAlign(CENTER,CENTER);
  textSize(20);
  noStroke();
}

void draw() {

  background(0);
  translate(300,50);    //  position of joint #1
  rotate(radians(90));  //  Flip it VERTICALLY DOWN

  float a1 = map(mouseX, 0, width, 90, -90);
  float a2 = map(mouseY, 0, width, 90, -90);

  rotate(radians(a1));   // joint #1 rotation
  drawGrid(120);
  fill(255,0,0);
  ellipse(0,0,100,100);
  rect(50,-50,200,100);
  fill(255);
  text(a1,0,0);

  translate(300,0);      // joint #2 position, 300 away.
  rotate(radians(a2));
  drawGrid(50);
  text(a2,0,0);
  fill(0,0,255);
  ellipse(0,0,100,100);
  rect(50,-50,200,100);
  fill(255);
  text(a2,0,0);

}
```





In-class exercise



By using the transformation techniques we have introduced today, draw a shape similar to the left one with a bitmap of your choice. In addition, a number has to printed along with each image.

Hint: It takes an additional transformation to print the number like this next to the image.