Loops

While loops

Loops

Loops execute code repeatedly.

```
int i=0;

while (i < 5) {
     println(i);
     i++;
}
println(“Done”);
```

While the condition in green is true...

```
int i=0;

while (i < 5) {
     println(i);
     i++;
}
println(“Done”);
```

Execute the code ‘in’ the loop (between { and } repeatedly.

```
int i=0;

while (i < 5) {
     println(i);
     i++;
}
println(“Done”);
```

One line of code between the braces is i++;
The variable i starts at zero, and increases with each execution of the loop.

```
int i=0;

while (i < 5) {
     println(i);
     i++;  ←
}
println(“Done”);
```

```
int i=0;
       0
while (i < 5) {
     println(i);
     i++;
}
println(“Done”);
```

Output:

```
int i=0;
       0
while (i < 5) {
    println(i);
    i++;
}
println(“Done”);
```

Output:

```
0
```

```
int i=0;
       1
while (i < 5) {
     println(i);
     i++;
}
println(“Done”);
```

```
Output:

0
```

```
int i=0;
       1
while (i < 5) {
    println(i);
    i++;
}
println(“Done”);
```

Output:

```
0
```

```
int i=0;
       1
while (i < 5) {
    println(i);
    i++;
}
println(“Done”);
```

Output:

```
0
1
```

```
int i=0;
       2
while (i < 5) {
     println(i);
     i++;
}
println(“Done”);
```

```
Output:

0
1
```

```
int i=0;
       2
while (i < 5) {
    println(i);
    i++;
}
println(“Done”);
```

```
Output:

0
1
```

```
int i=0;
       0
while (i < 5) {
    println(i);
    i++;
}
println(“Done”);
```

```
Output:

0
1
2
```

```
int i=0;
       3
while (i < 5) {
    println(i);
    i++;
}
println(“Done”);
```

Output:

```
0
1
2
```

```
int i=0;
       3
while (i < 5) {
     println(i);
     i++;
}
println(“Done”);
```

```
Output:

0
1
2
```

```
int i=0;
       3
while (i < 5) {
     println(i);
     i++;
}
println(“Done”);
```

```
Output:

0
1
2
3
```

```
int i=0;
       4
while (i < 5) {
    println(i);
    i++;
}
println(“Done”);
```

```
Output:

0
1
2
3
```

```
int i=0;
       4
while (i < 5) {
    println(i);
    i++;
}
println(“Done”);
```

```
Output:

0
1
2
3
```

```
int i=0;
      4
while (i < 5) {
    println(i);
    i++;
}
println(“Done”);
```

```
Output:

0
1
2
3
4
```

```
int i=0;
       5
while (i < 5) {
    println(i);
    i++;
}
println(“Done”);
```

```
Output:

0
1
2
3
4
```

```
int i=0;
      5
while (i < 5) {
    println(i);
    i++;
}
println(“Done”);
```

```
Output:

0
1
2
3
4
```

```
int i=0;
       5
while (i < 5) {
    println(i);
    i++;
}
println(“Done”);
```

```
Output:

0
1
2
3
4
```

```
int i=0;
       5
while (i < 5) {
    println(i);
    i++;
}
println(“Done”);
```

```
Output:

0
1
2
3
4
Done
```

If i does not increase then i never becomes greater than 5. This creates an infinite loop.

```
int i=0;

while (i < 5) {
    println(i);
    // i++;
}
println(“Done”);
```

For loops

A for loop deals with this situation in a more compact way, and avoids the infinite loop.

```
int i=0;

while (i < 5) {
    println(i);
    i++;
}
println(“Done”);
```

```
for (int i=0; i < 5; i++) {
    println(i);
}
println(“Done”);
```

```
int i=0;

while (i < 5) {
     println(i);
     i++;
}
println(“Done”);
```

Start End

```
for (int i=0; i < 5; i++) {
     println(i);
}
println(“Done”);
```

Start End

```
for (int i=0; i < 5; i++) {
     println(i);
}
println(“Done”);
```

Output:

Start —— {

} —— End

true

```
for (int i=0; i < 5; i++) {
     println(i);
}
println(“Done”);
```

Output:

0

```
for (int i=0; i < 5; i++) {
    println(i);
}
println(“Done”);
```

Output:

```
         0     true

for (int i=0; i < 5; i++) {
    println(i);
}
println(“Done”);
```

Output:

0

```
for (int i=0; i < 5; i++) {
     println(i);
}
println(“Done”);
```

Output:

```
0
```

1

```
for (int i=0; i < 5; i++) {
     println(i);
}
println(“Done”);
```

Output:

```
0
```

```
         1     true

for (int i=0; i < 5; i++) {
    println(i);
}
println(“Done”);
```

Output:

2

```
for (int i=0; i < 5; i++) {
     println(i);
}
println("Done");
```

Output:

```
0
1
```

```
         2     true

for (int i=0; i < 5; i++) {
     println(i);
}
println(“Done”);
```

Output:

```
0
1
```

```
         3

for (int i=0; i < 5; i++) {
     println(i);
}
println(“Done”);
```

```
Output:

0
1
2
```

3 true

Output:

```
for (int i=0; i < 5; i++) {
    println(i);
}
println(“Done”);
```

3

```
for (int i=0; i < 5; i++) {
     println(i);
}
println(“Done”);
```

```
Output:

0
1
2
3
```

4

```
for (int i=0; i < 5; i++) {
     println(i);
}
println(“Done”);
```

Output:

```
0
1
2
3
```

```
         4    true

for (int i=0; i < 5; i++) {
    println(i);
}
println(“Done”);
```

```
Output:

0
1
2
3
```

4

```
for (int i=0; i < 5; i++) {
    println(i);
}
println(“Done”);
```

```
Output:

0
1
2
3
4
```

5

```
for (int i=0; i < 5; i++) {
     println(i);
}
println(“Done”);
```

Output:

```
0
1
2
3
4
```

```
         5    false

for (int i=0; i < 5; i++) {
    println(i);
}
println(“Done”);
```

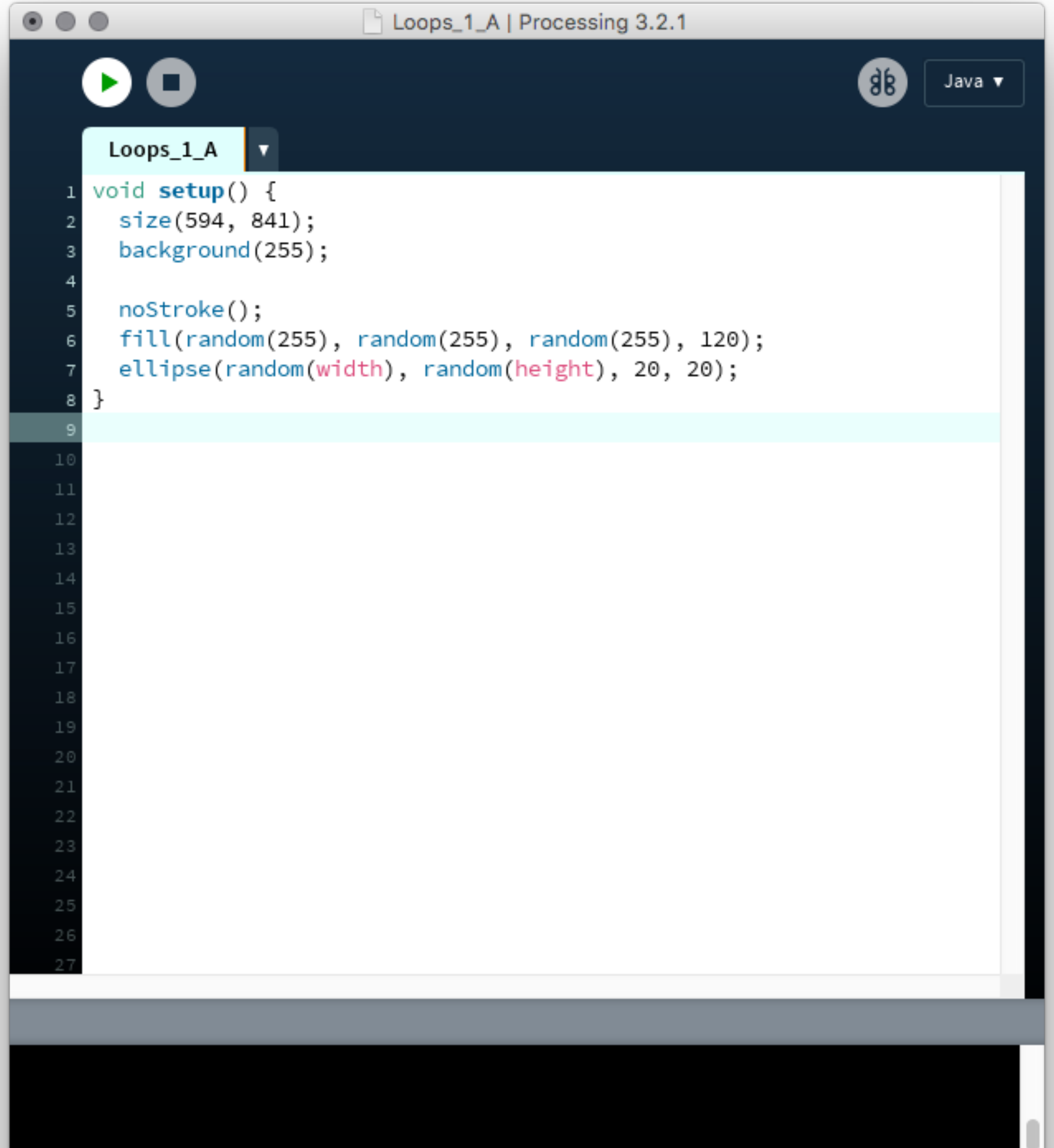
Output:

```
0
1
2
3
4
```

```
         5    false

for (int i=0; i < 5; i++) {
     println(i);
}
println(“Done”);
```

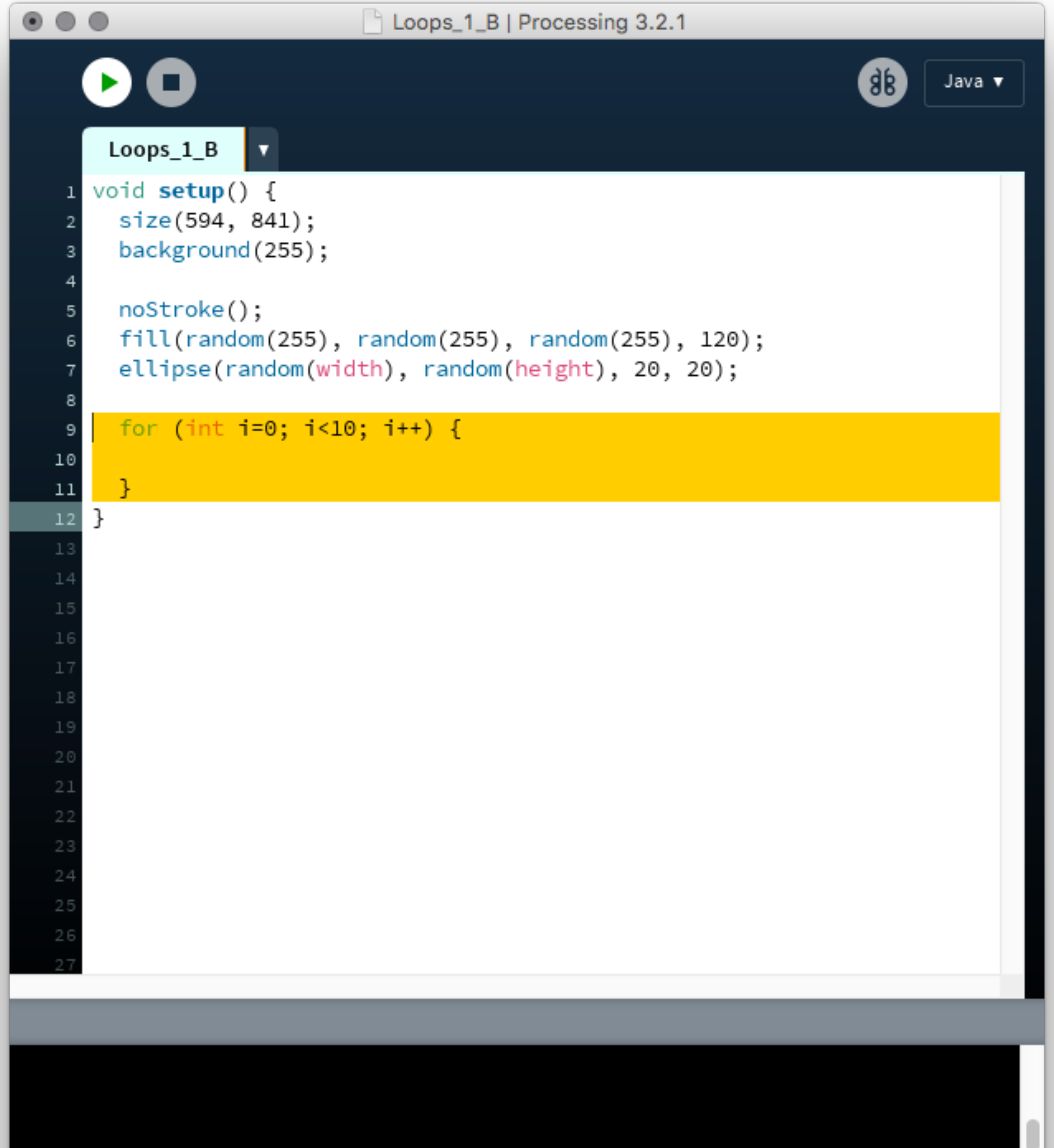
Output:

```
0
1
2
3
4
Done
```



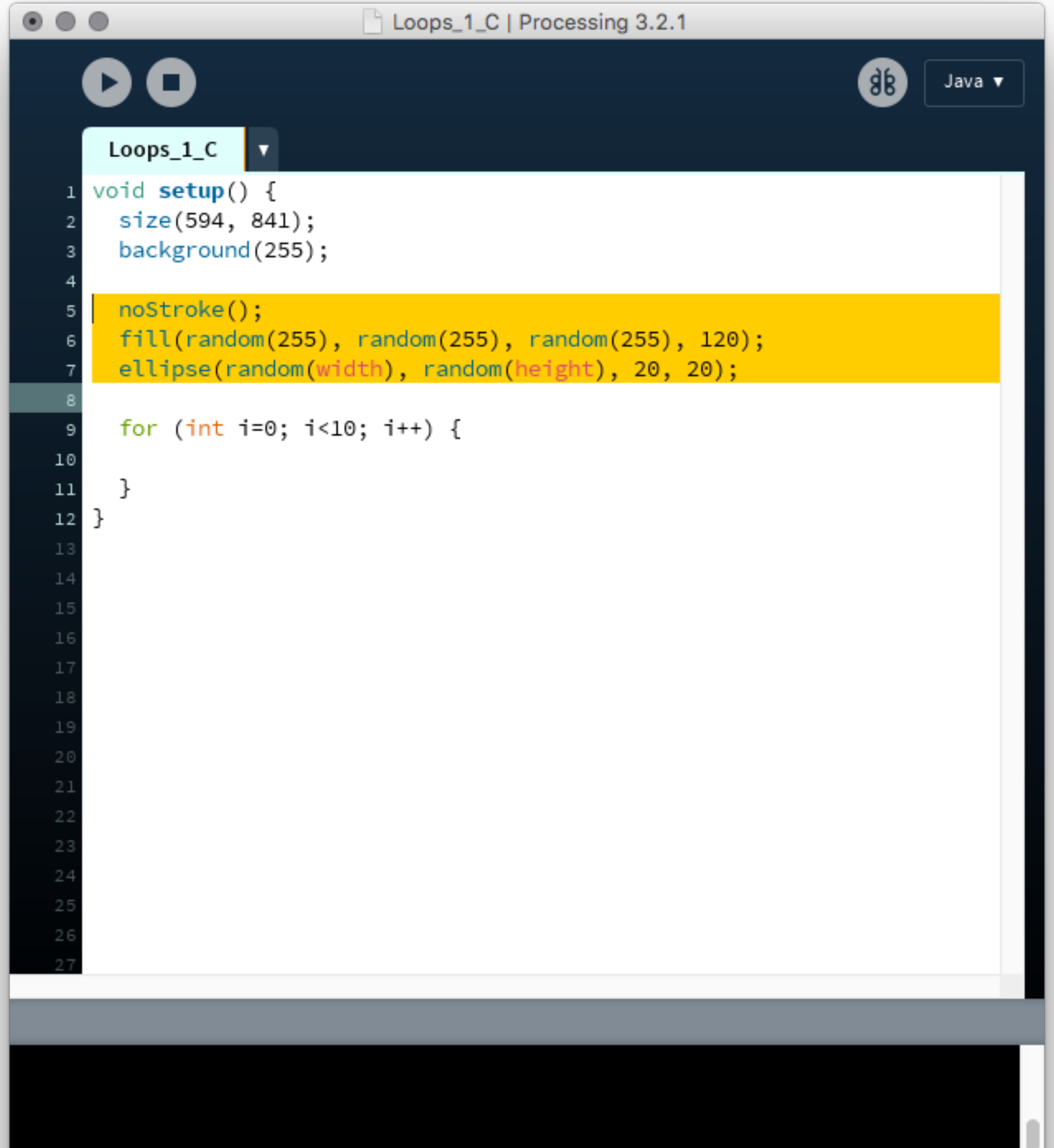
```
void setup() {
  size(594, 841);
  background(255);

  noStroke();
  fill(random(255), random(255), random(255), 120);
  ellipse(random(width), random(height), 20, 20);
}
```

Start with a single randomly colored, randomly positioned ellipse.



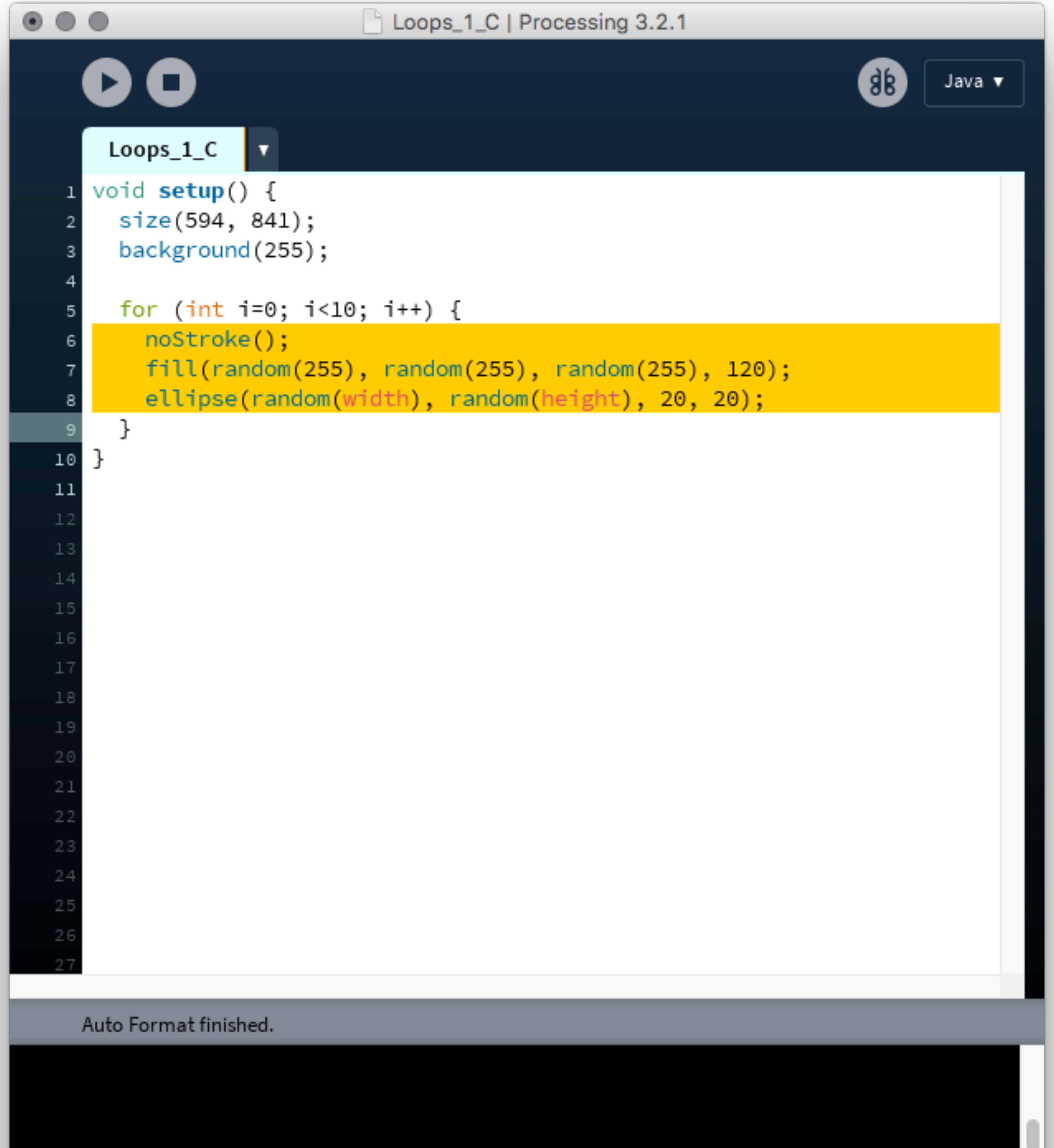
```
void setup() {
  size(594, 841);
  background(255);

  noStroke();
  fill(random(255), random(255), random(255), 120);
  ellipse(random(width), random(height), 20, 20);

  for (int i=0; i<10; i++) {

  }
}
```

Add a for loop that executes 10 times



```
void setup() {
  size(594, 841);
  background(255);

  noStroke();
  fill(random(255), random(255), random(255), 120);
  ellipse(random(width), random(height), 20, 20);

  for (int i=0; i<10; i++) {

  }
}
```

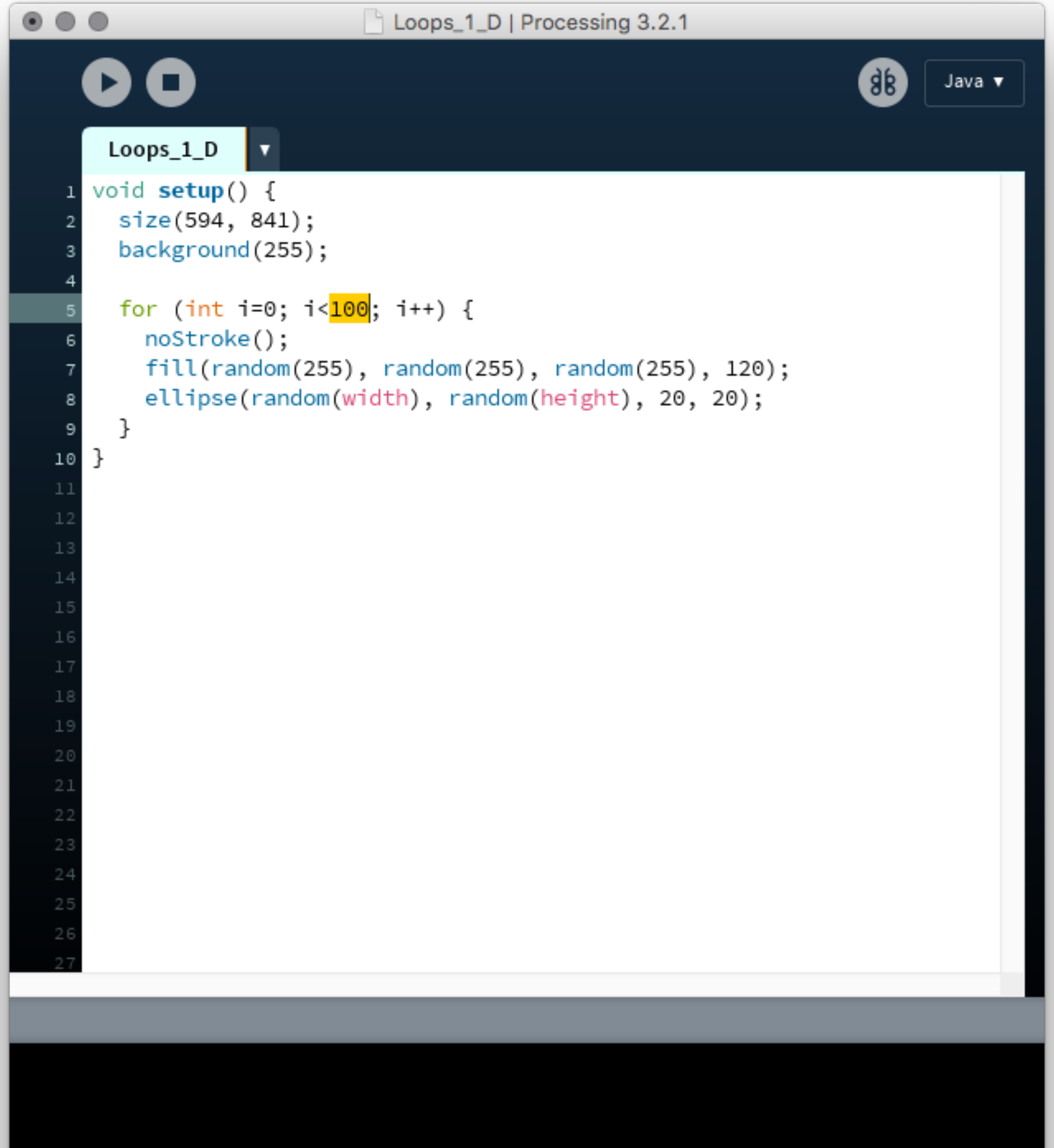
Take the block of code that draws the ellipse…



```
void setup() {
  size(594, 841);
  background(255);

  for (int i=0; i<10; i++) {
    noStroke();
    fill(random(255), random(255), random(255), 120);
    ellipse(random(width), random(height), 20, 20);
  }
}
```

Auto Format finished.

And move it ‘into’ the for loop.



```
void setup() {
  size(594, 841);
  background(255);

  for (int i=0; i<100; i++) {
    noStroke();
    fill(random(255), random(255), random(255), 120);
    ellipse(random(width), random(height), 20, 20);
  }
}
```

Change the number of time the loop executes.

Functions

setup and draw are functions. You decide what they do, but processing decides when they are called.

```
void setup() {

}


void draw() {

}
```

random() and ellipse() are functions. You decide when they're called, but processing decides what they do.

We are going create our own functions. We will decide what they do, AND when they are called.

```
random(100);

ellipse(10, 10, 20, 20);
```

functions have types. Random always ‘returns’ a float.

ellipse does not return anything. It ‘does’ something.

float

```
float x = random(100); // 36.751217

          ellipse(10, 10, 20, 20);
```

void

Variable types

```
int

float

boolean

color

char

String
```

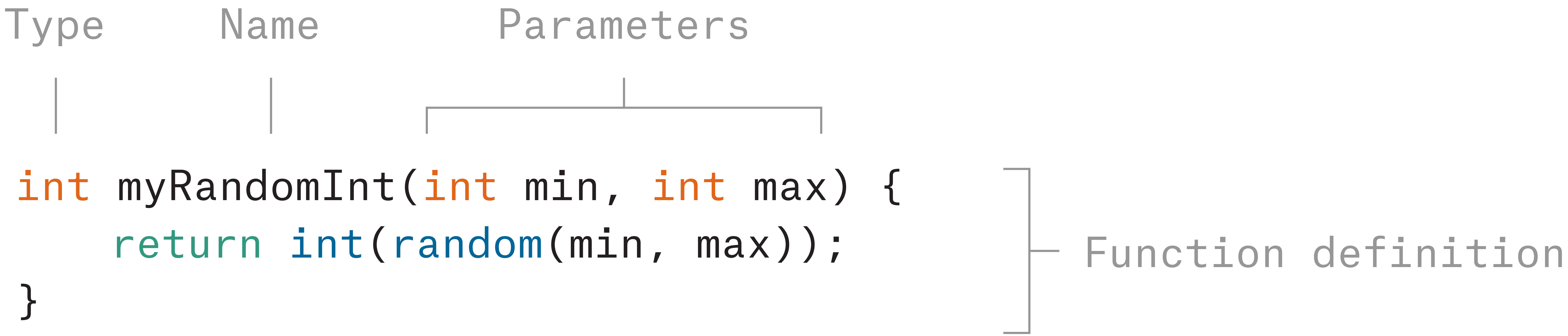
So do functions.

Variable types	Return types
int	int
float	float
boolean	boolean
color	color
char	char
String	String
	void

This function calculates something and ‘returns’ it. It generates a random number, rounds it down, and returns it as an integer.

```
int myRandomInt(int min, int max) {
    return int(random(min, max));
}
```

Function definition



Type

Name

Parameters

```
int myRandomInt(int min, int max) {
    return int(random(min, max));
}
```

Function definition

When you call it, it returns an int, and you can store that number in a variable of type int.

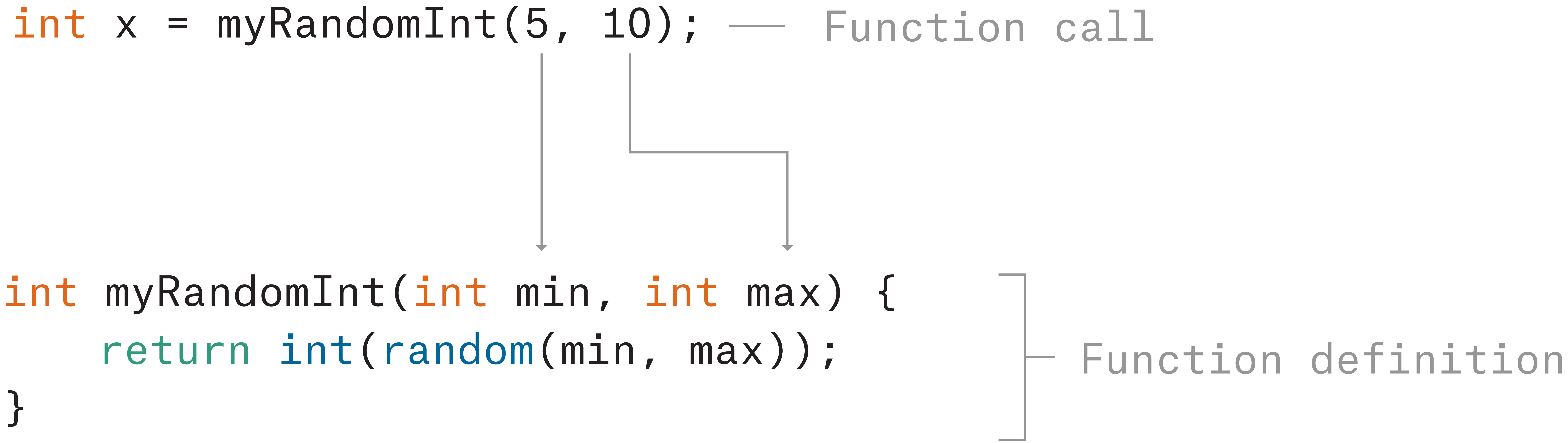
```
int x = myRandomInt(5, 10);
```

Function call

```
int myRandomInt(int min, int max) {
    return int(random(min, max));
}
```

Function definition

Functions accept parameters that determine their behavior.



```
int x = myRandomInt(5, 10);
```

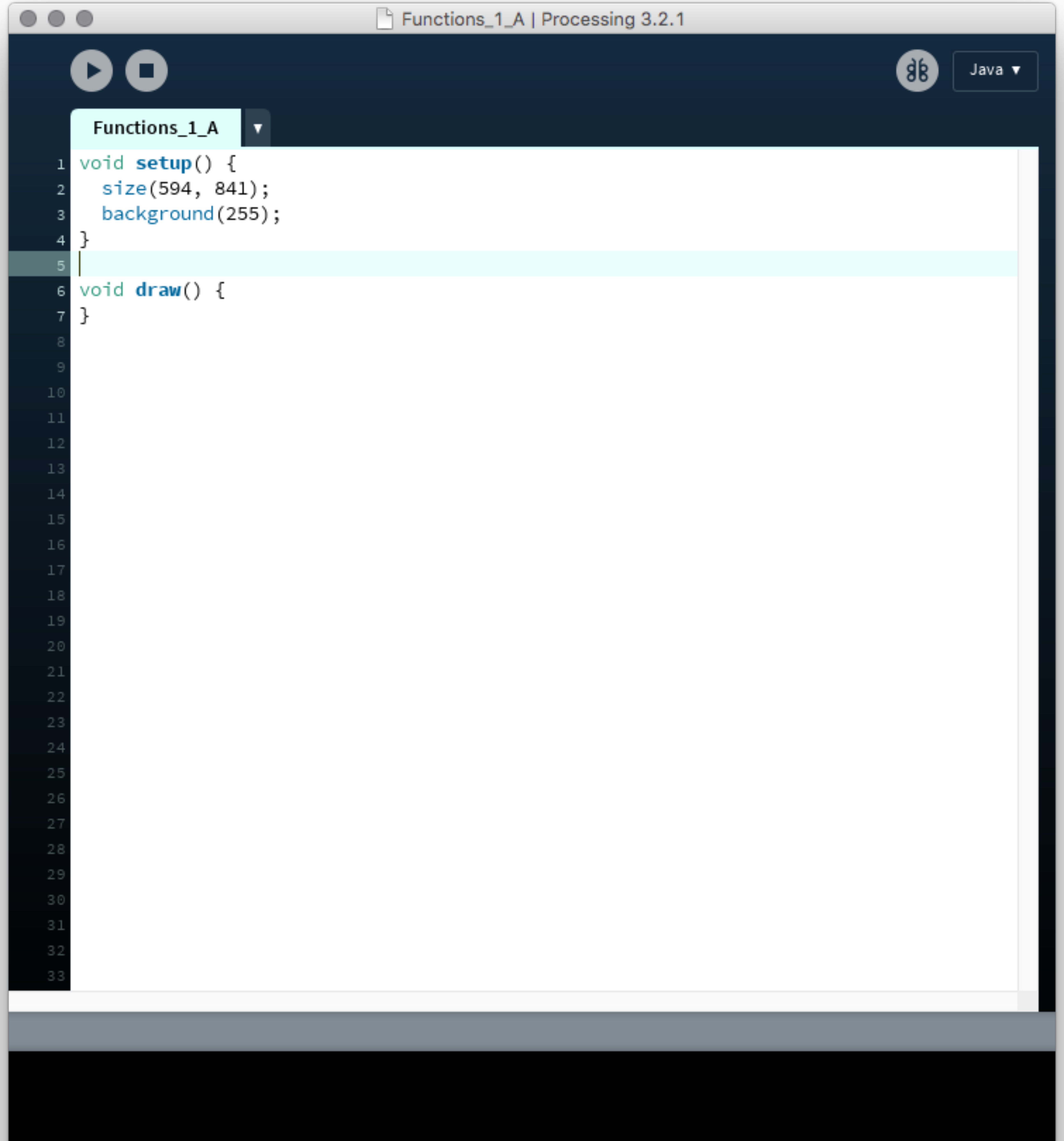
Function call

```
int myRandomInt(int min, int max) {
    return int(random(min, max));
}
```

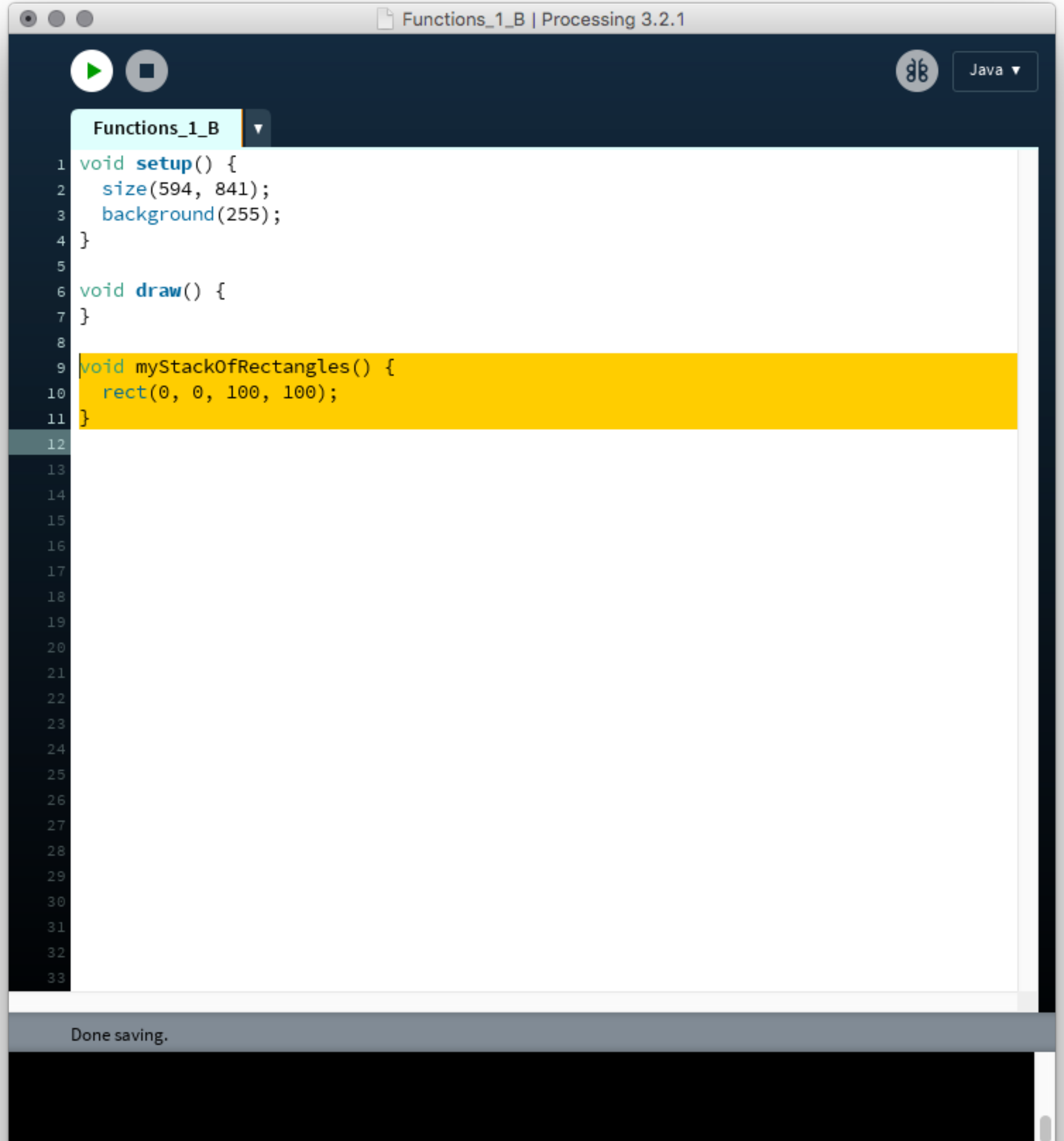
Function definition

Some function do not calculate anything / do not return anything. A function that draws something to the canvas has no return value. These are type void.

```
void myCoolDrawingFunction() {
    ~~return ???;~~
    // draws something
}
```



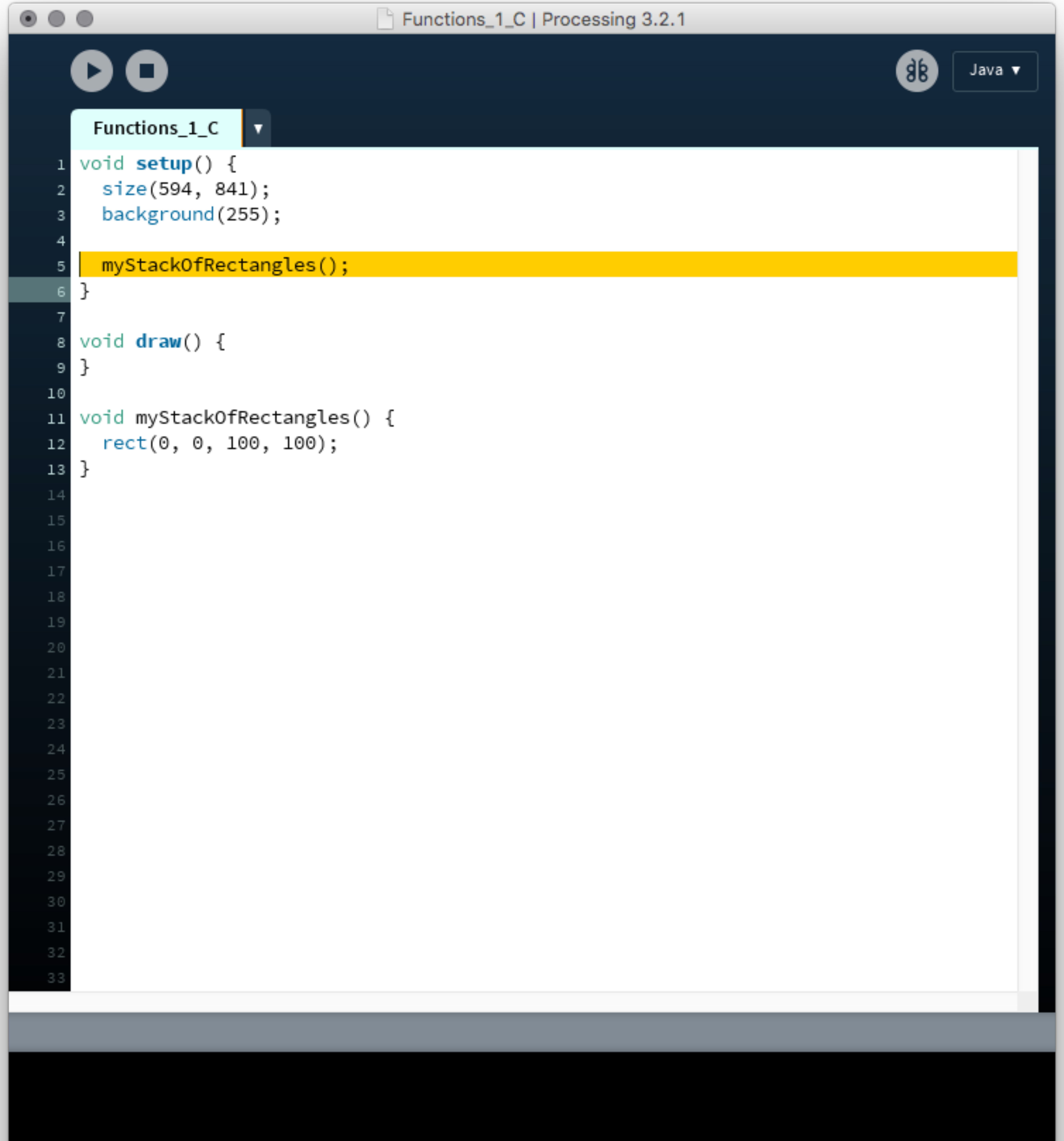
```
void setup() {
  size(594, 841);
  background(255);
}

void draw() {
}
```



```
void setup() {
  size(594, 841);
  background(255);
}

void draw() {
}

void mystackOfRectangles() {
  rect(0, 0, 100, 100);
}
```

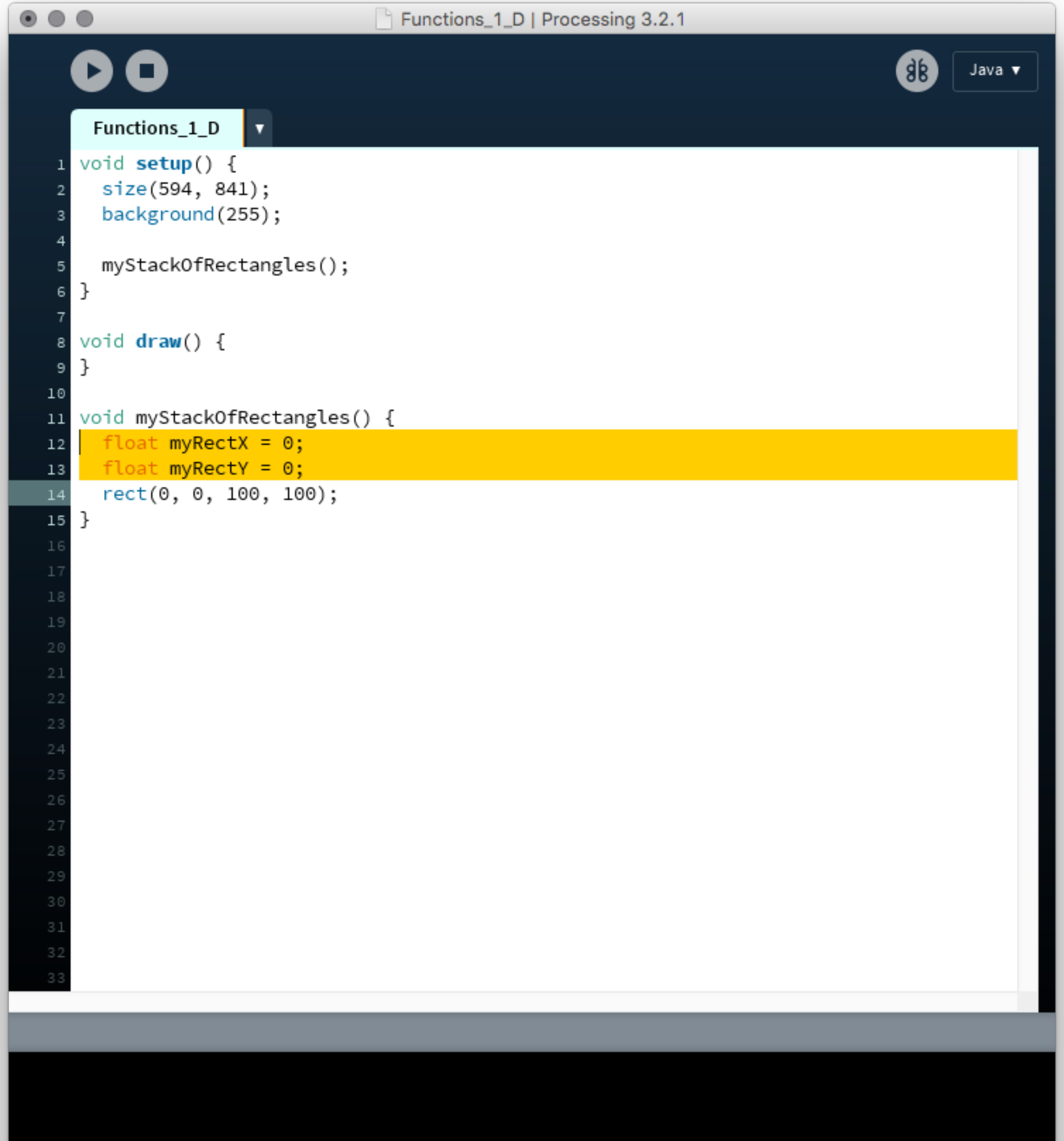
Done saving.

Create a function.



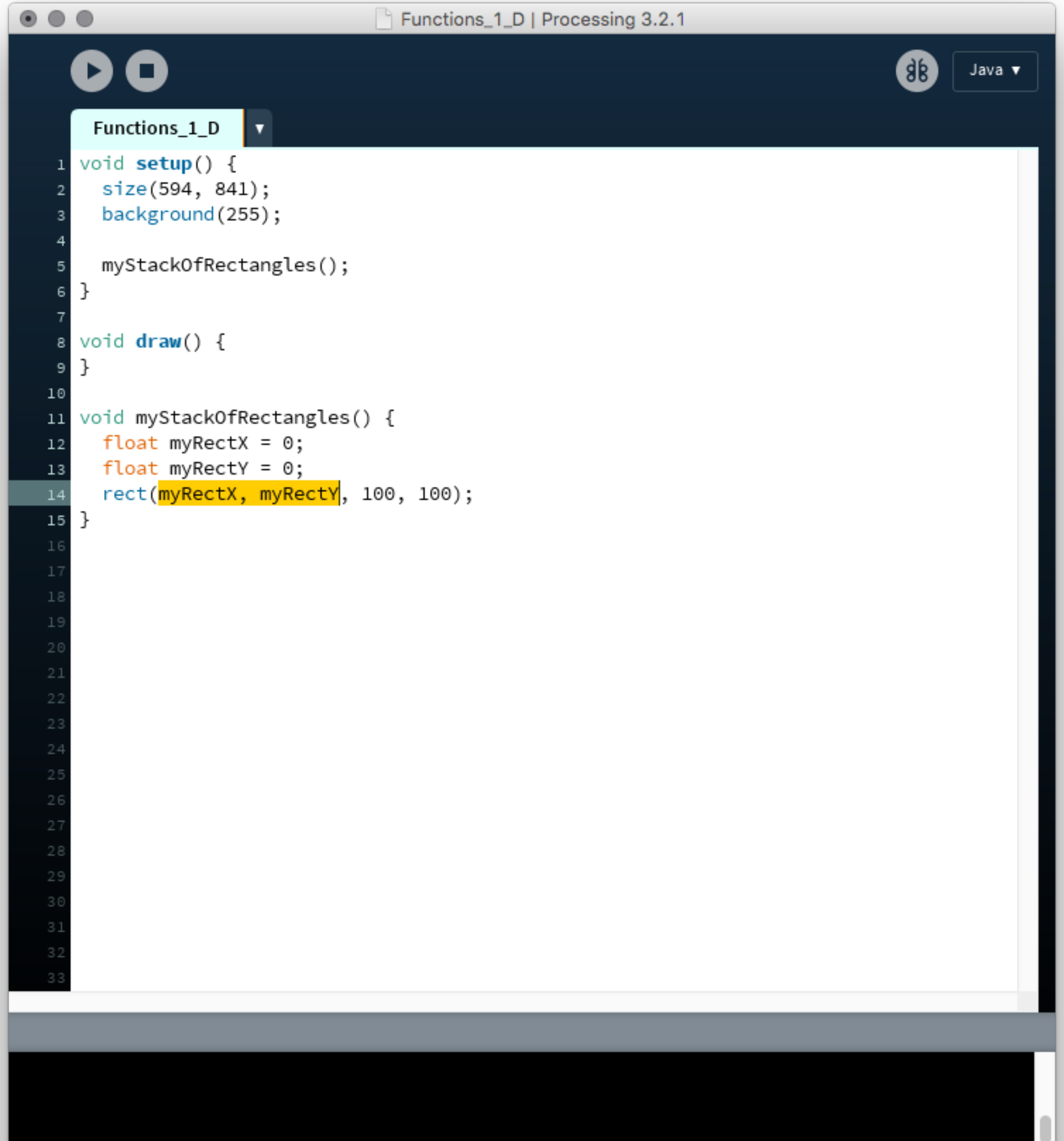
```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles();
}

void draw() {
}

void myStackOfRectangles() {
  rect(0, 0, 100, 100);
}
```

Call the function.



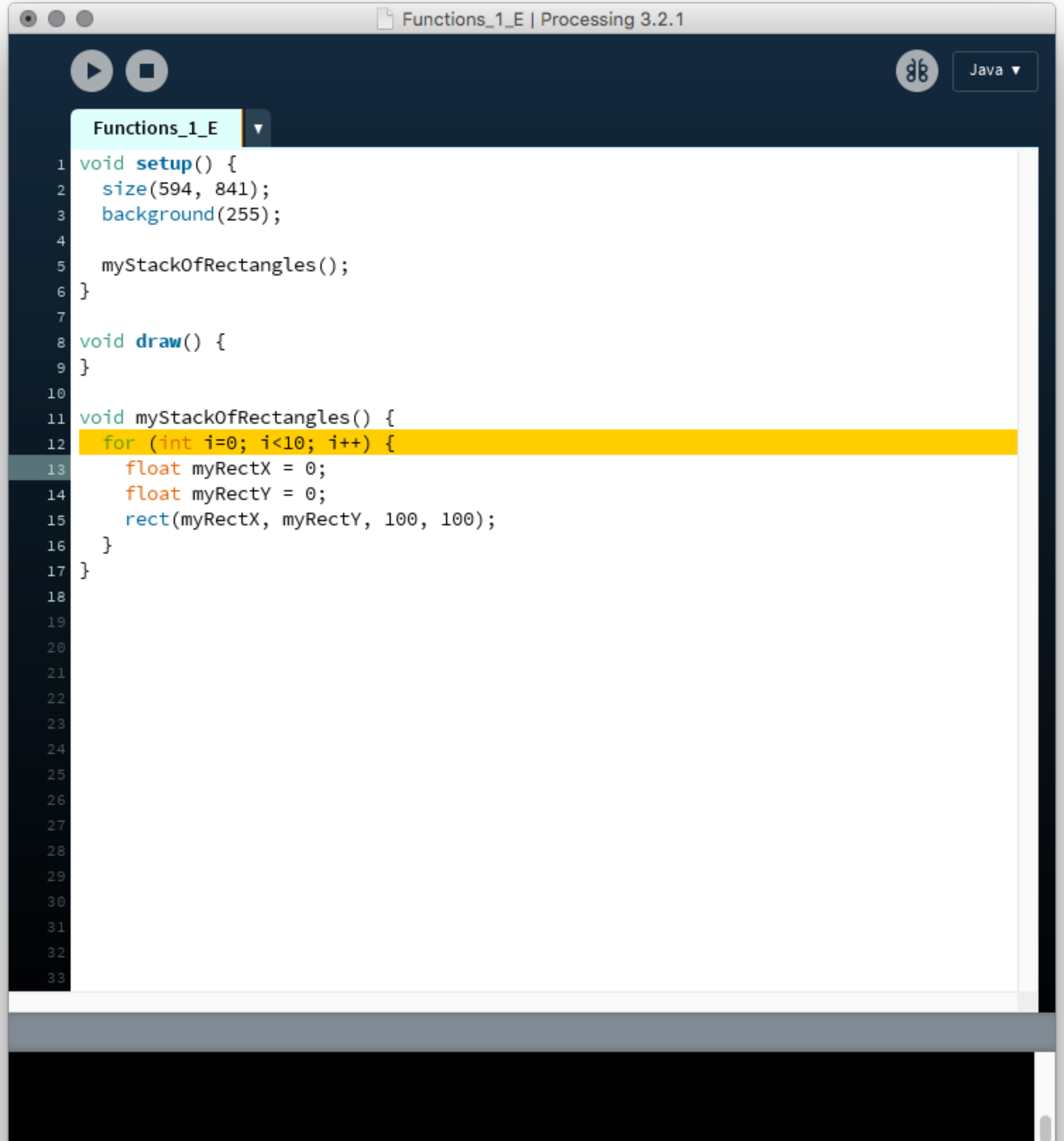
```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles();
}

void draw() {
}

void myStackOfRectangles() {
  float myRectX = 0;
  float myRectY = 0;
  rect(0, 0, 100, 100);
}
```

Create variables to store the x and y coordinates.

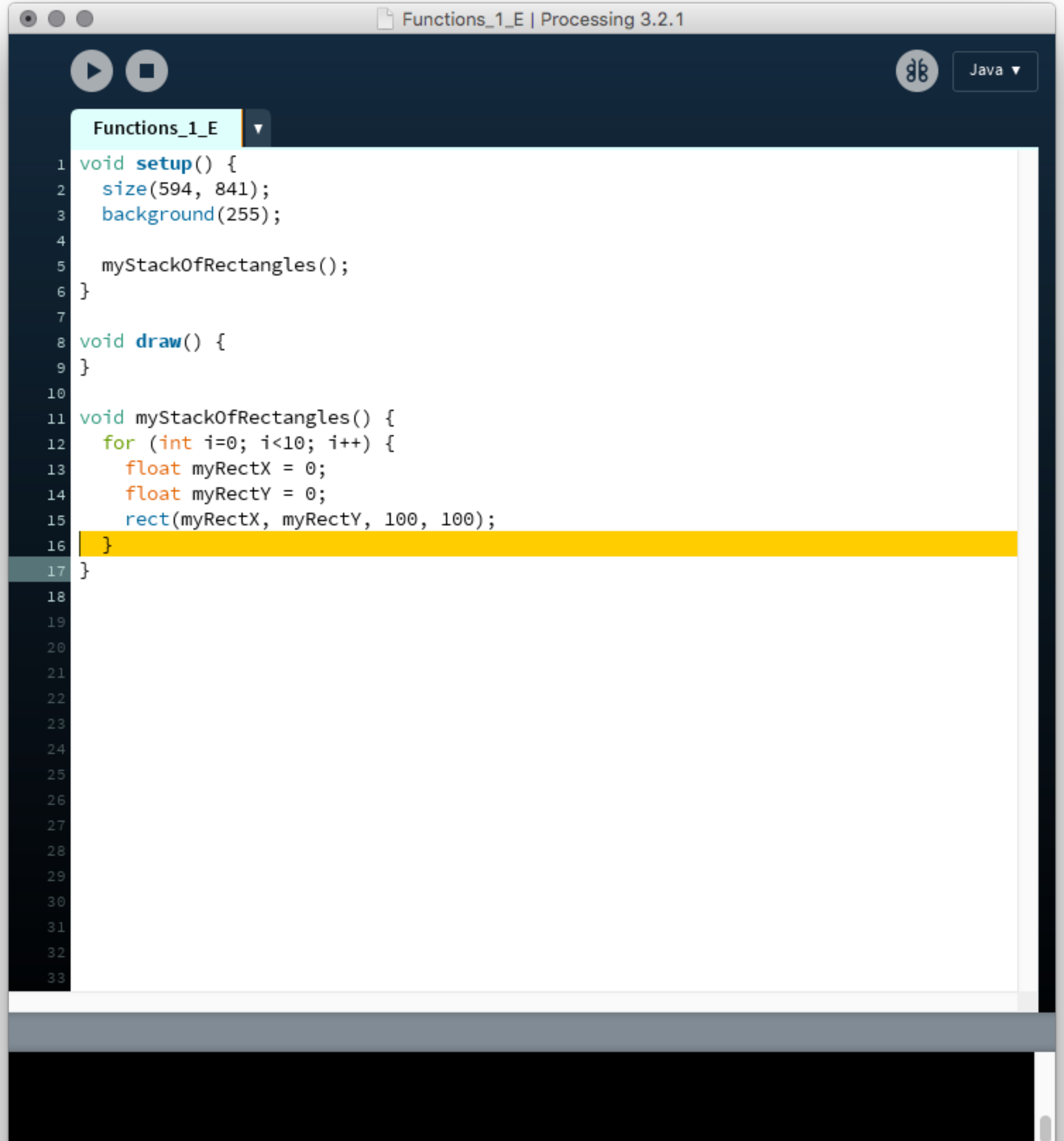


```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles();
}

void draw() {
}

void myStackOfRectangles() {
  float myRectX = 0;
  float myRectY = 0;
  rect(myRectX, myRectY, 100, 100);
}
```

And plug these into the rect function.

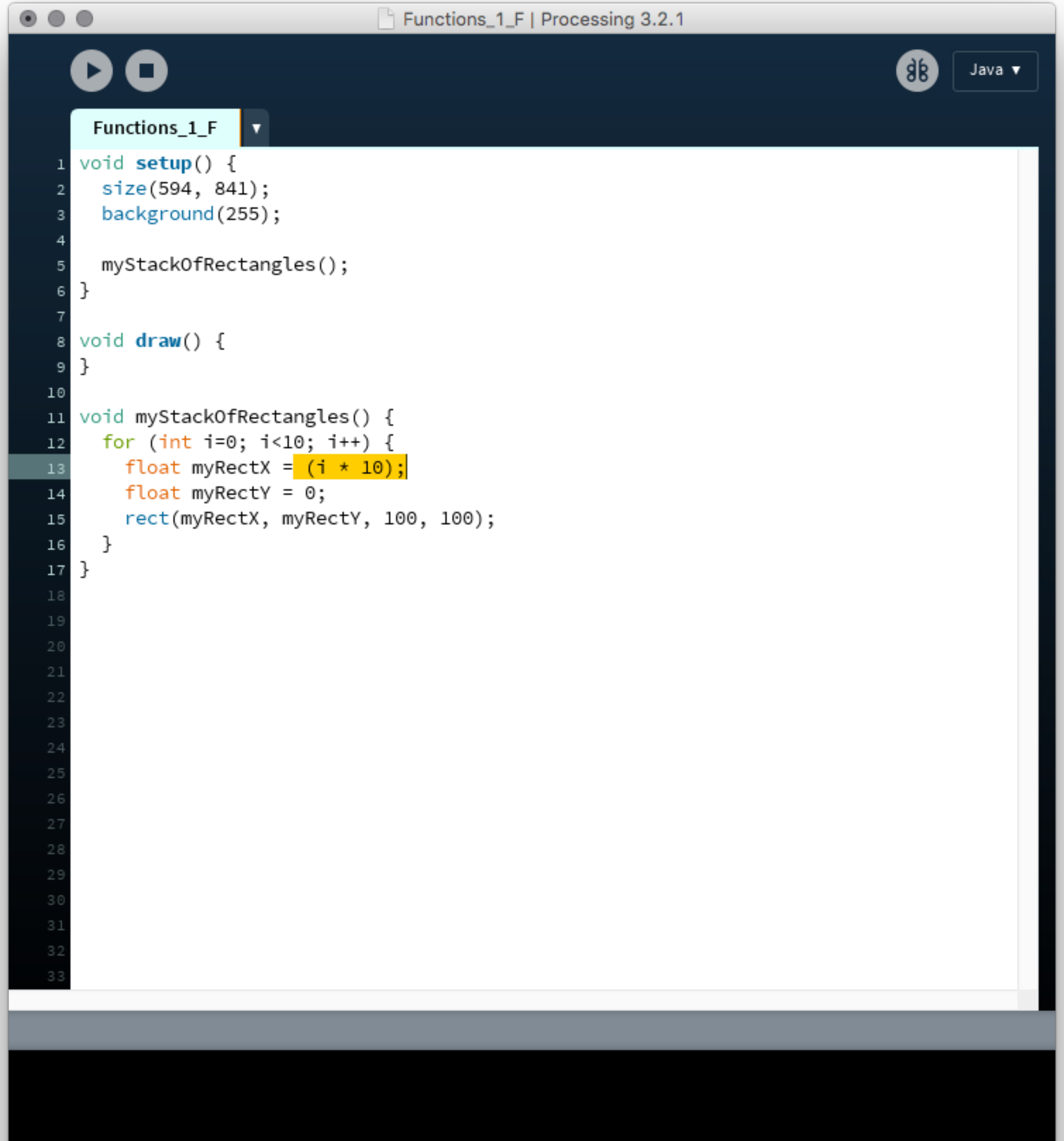


```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles();
}

void draw() {
}

void myStackOfRectangles() {
  for (int i=0; i<10; i++) {
    float myRectX = 0;
    float myRectY = 0;
    rect(myRectX, myRectY, 100, 100);
  }
}
```



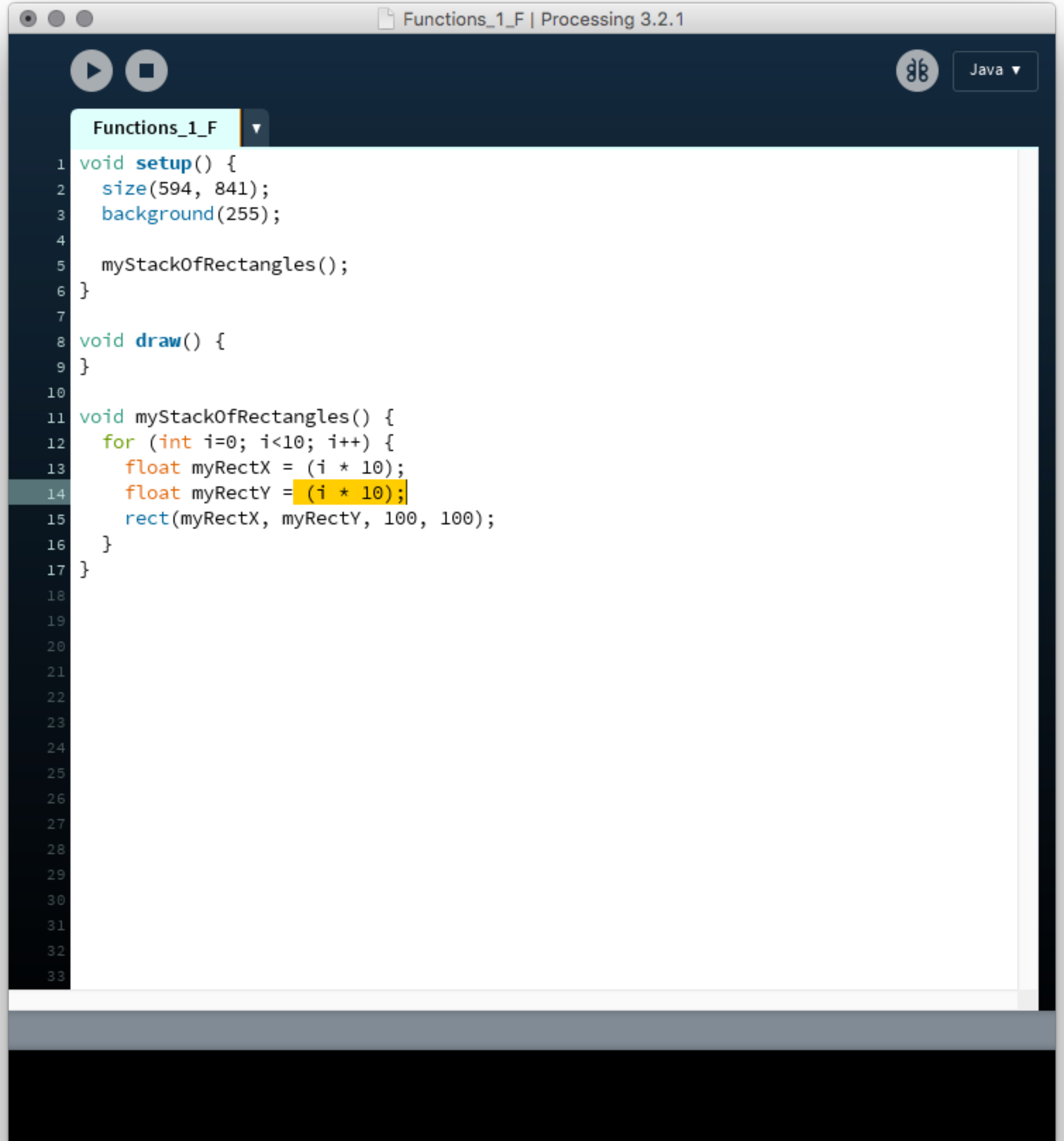
```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles();
}

void draw() {
}

void myStackOfRectangles() {
  for (int i=0; i<10; i++) {
    float myRectX = 0;
    float myRectY = 0;
    rect(myRectX, myRectY, 100, 100);
  }
}
```

Wrap this code in a for loop.

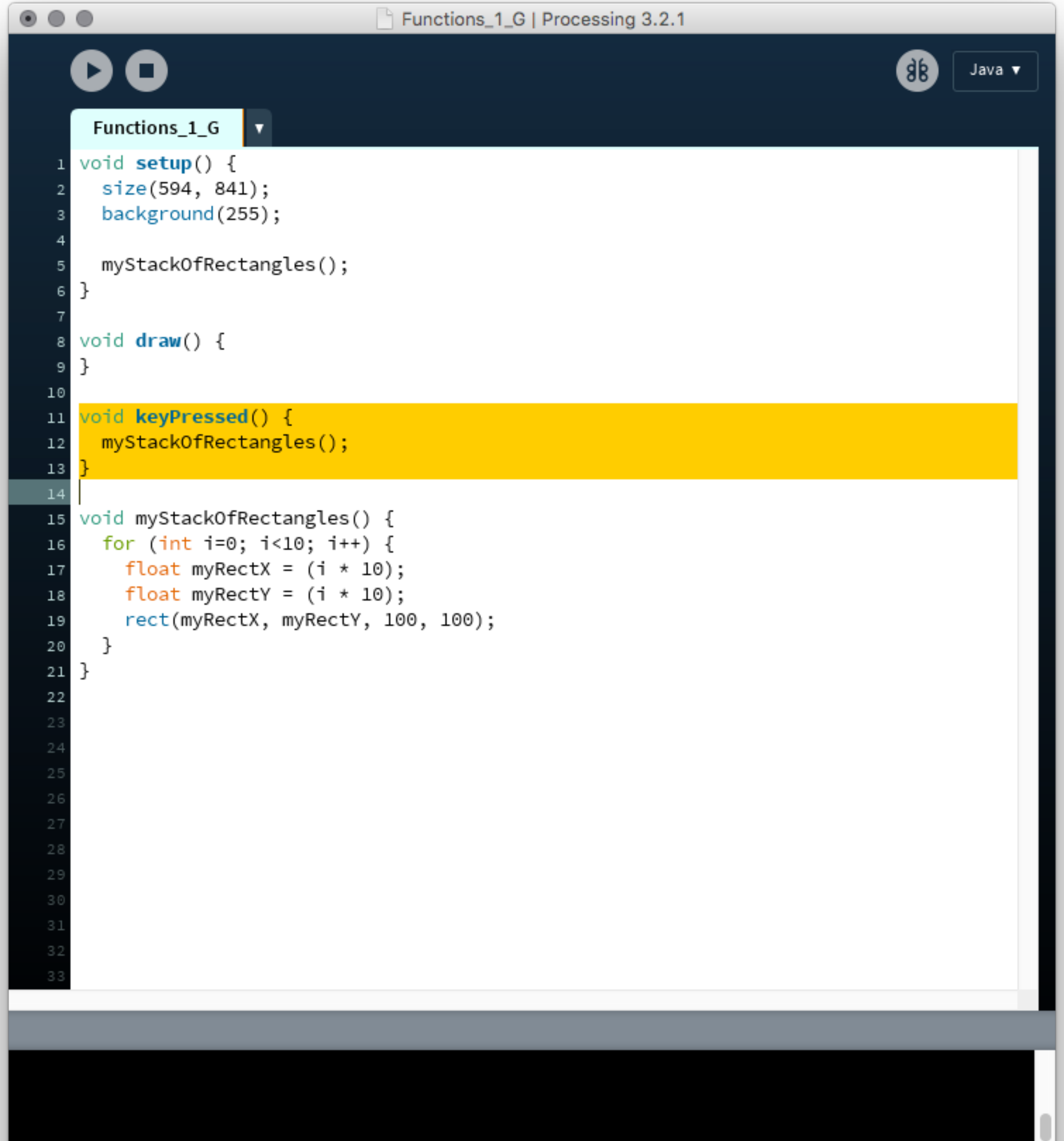


```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles();
}

void draw() {
}

void myStackOfRectangles() {
  for (int i=0; i<10; i++) {
    float myRectX = (i * 10);
    float myRectY = 0;
    rect(myRectX, myRectY, 100, 100);
  }
}
```

And use i to modify the position.

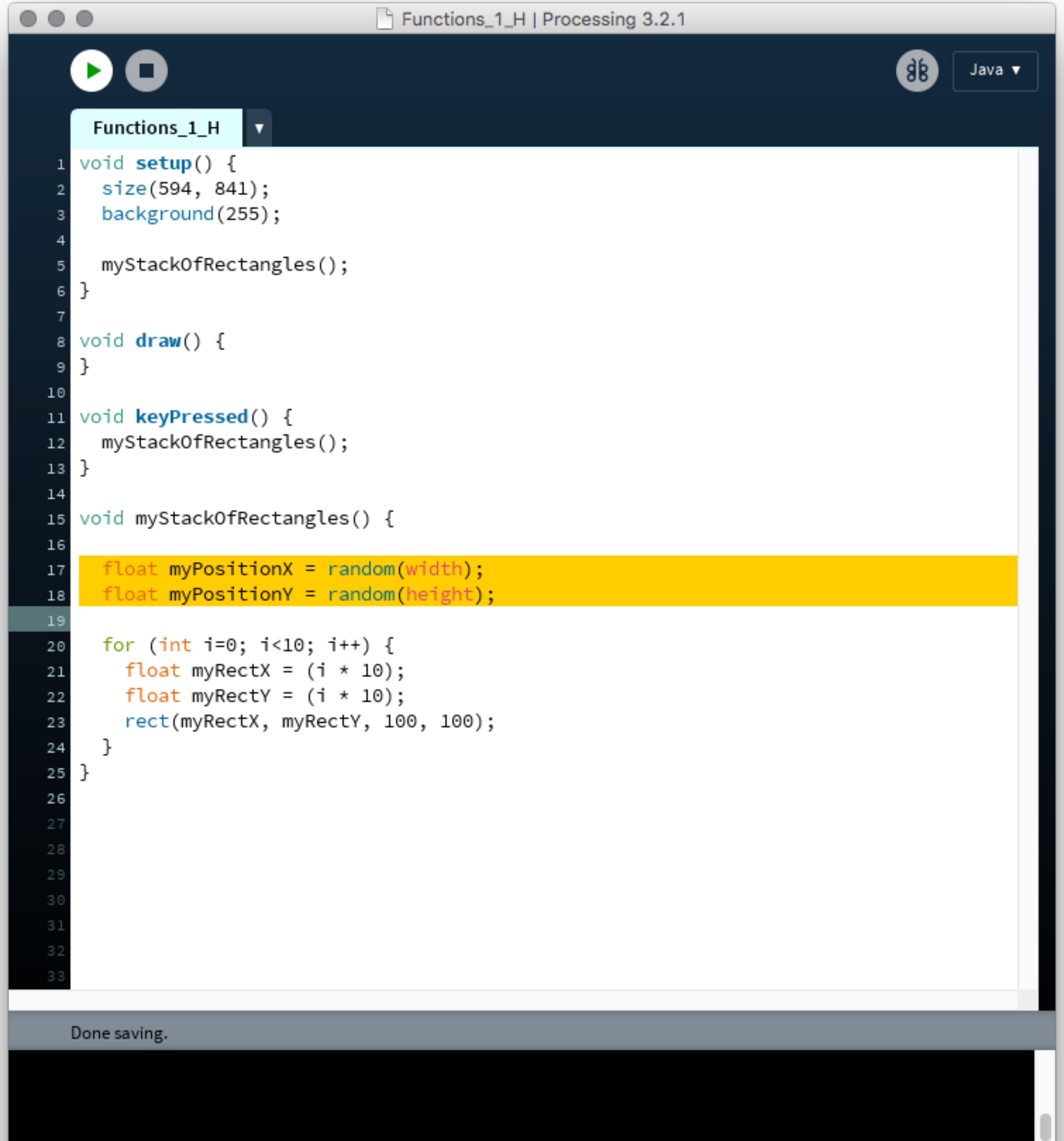


```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles();
}

void draw() {
}

void myStackOfRectangles() {
  for (int i=0; i<10; i++) {
    float myRectX = (i * 10);
    float myRectY = (i * 10);
    rect(myRectX, myRectY, 100, 100);
  }
}
```



```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles();
}

void draw() {
}

void keyPressed() {
  myStackOfRectangles();
}

void myStackOfRectangles() {
  for (int i=0; i<10; i++) {
    float myRectX = (i * 10);
    float myRectY = (i * 10);
    rect(myRectX, myRectY, 100, 100);
  }
}
```

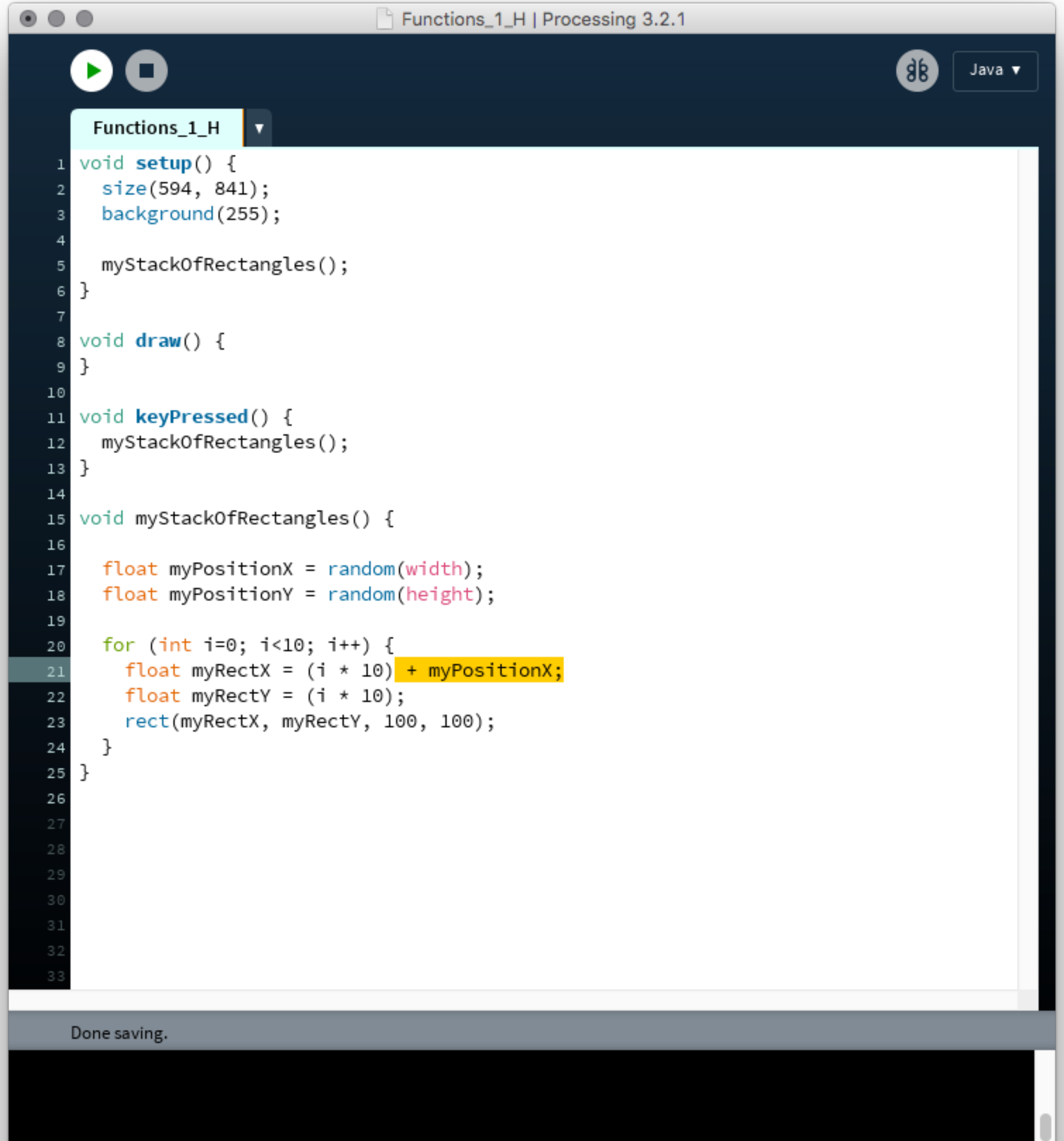
Also call this function whenever a key is pressed.



```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles();
}

void draw() {
}

void keyPressed() {
  myStackOfRectangles();
}

void myStackOfRectangles() {

  float myPositionX = random(width);
  float myPositionY = random(height);

  for (int i=0; i<10; i++) {
    float myRectX = (i * 10);
    float myRectY = (i * 10);
    rect(myRectX, myRectY, 100, 100);
  }
}
```

Done saving.

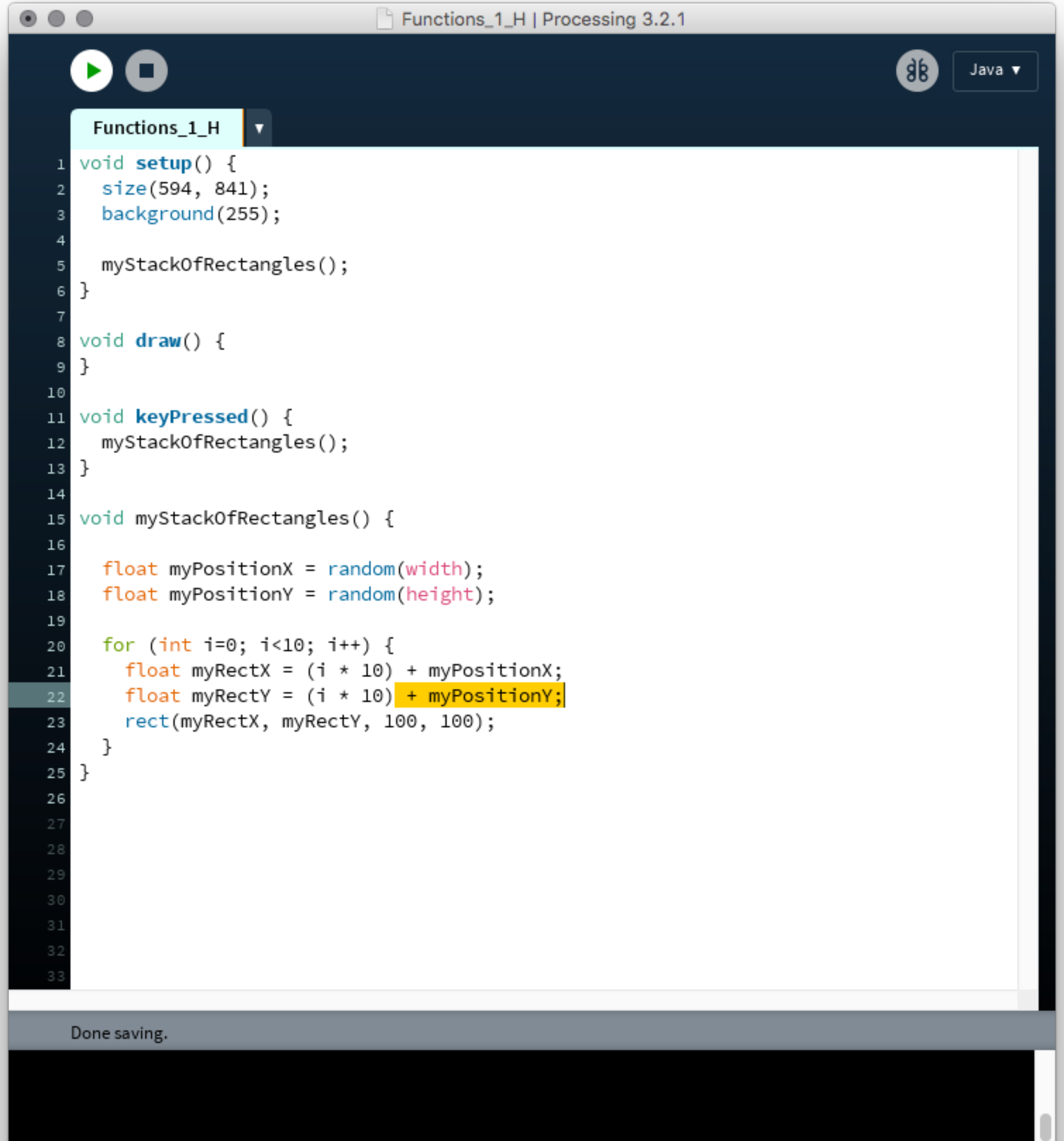
Add some variables for the x and y coordinate of the whole stack.



```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles();
}

void draw() {
}

void keyPressed() {
  myStackOfRectangles();
}

void myStackOfRectangles() {

  float myPositionX = random(width);
  float myPositionY = random(height);

  for (int i=0; i<10; i++) {
    float myRectX = (i * 10) + myPositionX;
    float myRectY = (i * 10);
    rect(myRectX, myRectY, 100, 100);
  }
}
```

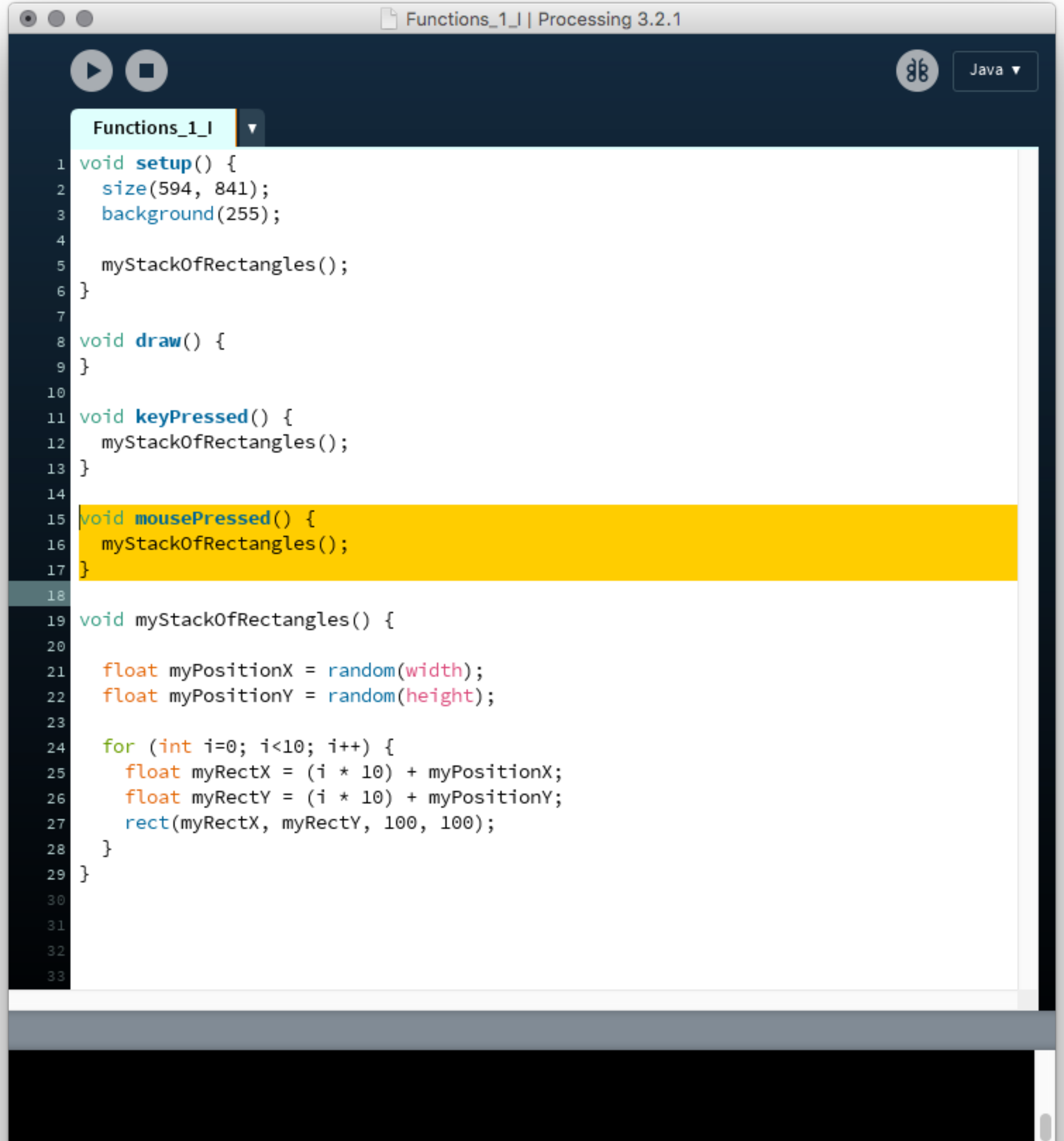
Done saving.

Add these numbers to the x and y coordinates of each rectangle.



```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles();
}

void draw() {
}

void keyPressed() {
  myStackOfRectangles();
}

void myStackOfRectangles() {

  float myPositionX = random(width);
  float myPositionY = random(height);

  for (int i=0; i<10; i++) {
    float myRectX = (i * 10) + myPositionX;
    float myRectY = (i * 10) + myPositionY;
    rect(myRectX, myRectY, 100, 100);
  }
}
```

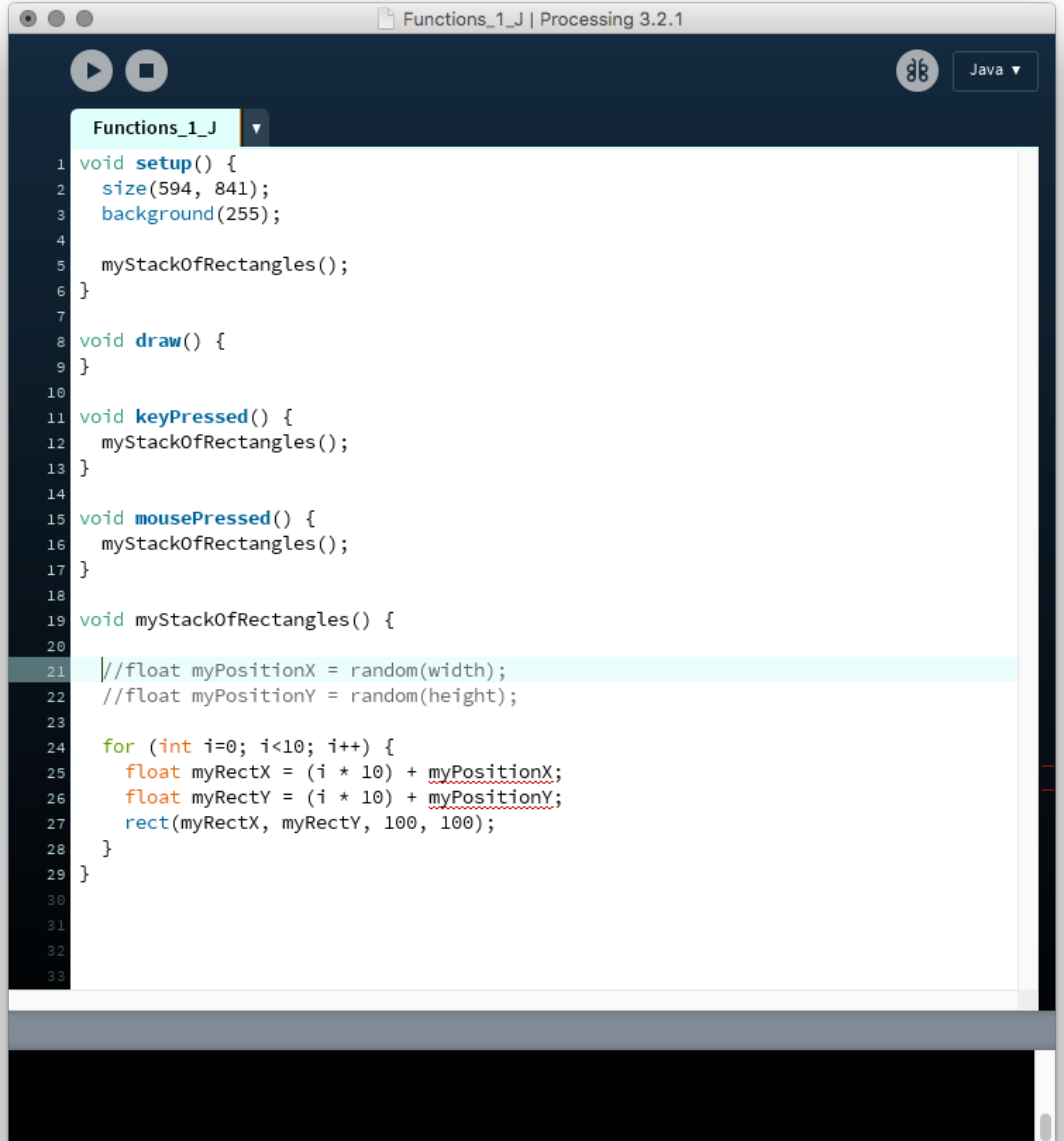
Done saving.



```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles();
}

void draw() {
}

void keyPressed() {
  myStackOfRectangles();
}

void mousePressed() {
  myStackOfRectangles();
}

void myStackOfRectangles() {

  float myPositionX = random(width);
  float myPositionY = random(height);

  for (int i=0; i<10; i++) {
    float myRectX = (i * 10) + myPositionX;
    float myRectY = (i * 10) + myPositionY;
    rect(myRectX, myRectY, 100, 100);
  }
}
```

Call the function whenever the mouse is pressed as well.

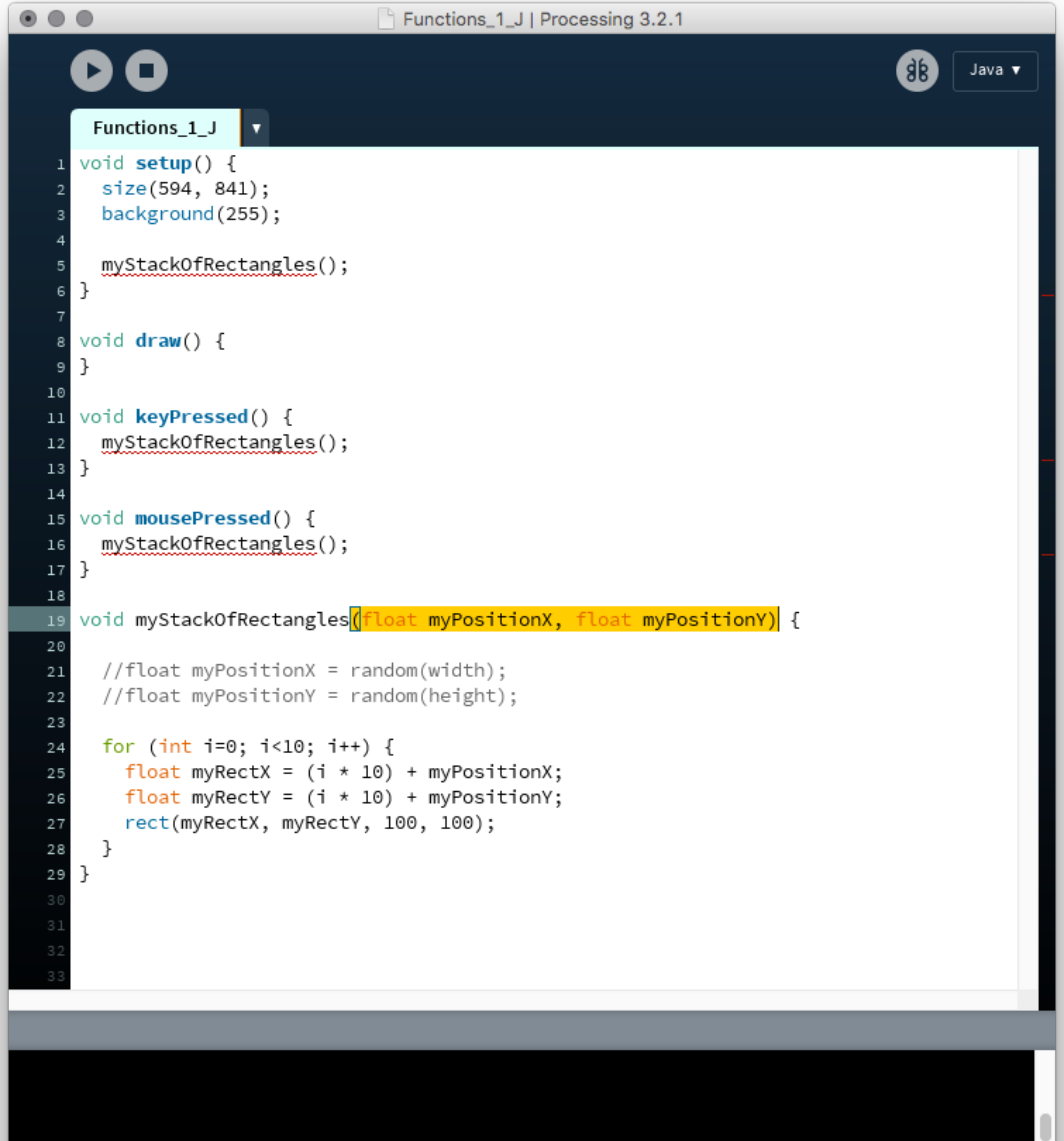
Now the function is called in 3 ways:
1. In setup() when the program starts
2. in keyPressed() whenever a key is pressed
3. in mousePressed() whenever the mouse is pressed.



```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles();
}

void draw() {
}

void keyPressed() {
  myStackOfRectangles();
}

void mousePressed() {
  myStackOfRectangles();
}

void myStackOfRectangles() {

  //float myPositionX = random(width);
  //float myPositionY = random(height);

  for (int i=0; i<10; i++) {
    float myRectX = (i * 10) + myPositionX;
    float myRectY = (i * 10) + myPositionY;
    rect(myRectX, myRectY, 100, 100);
  }
}
```

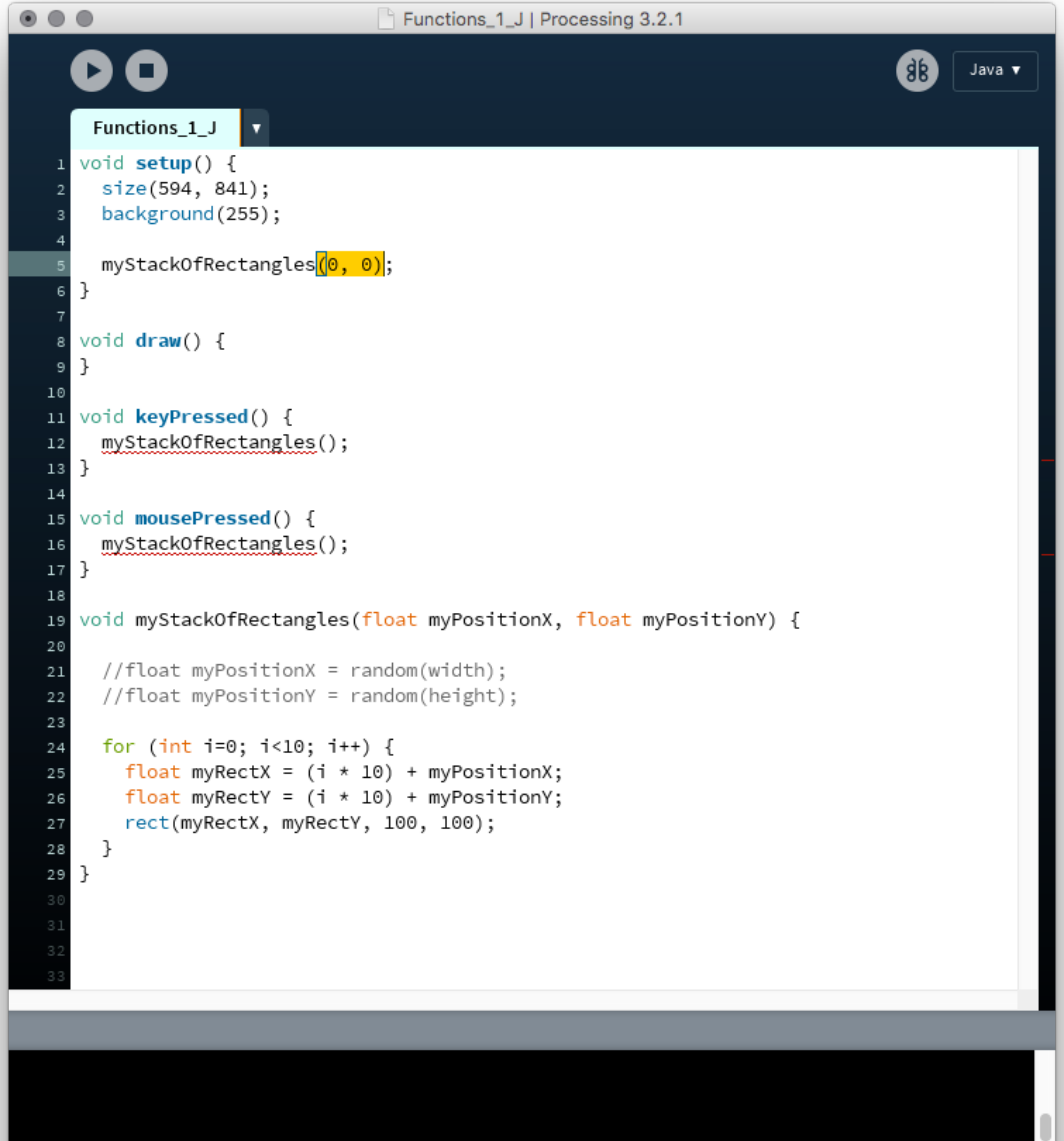
The position of the stack is determined by myPositionX and myPositionY.

Instead of letting these be assigned a random number in the function...



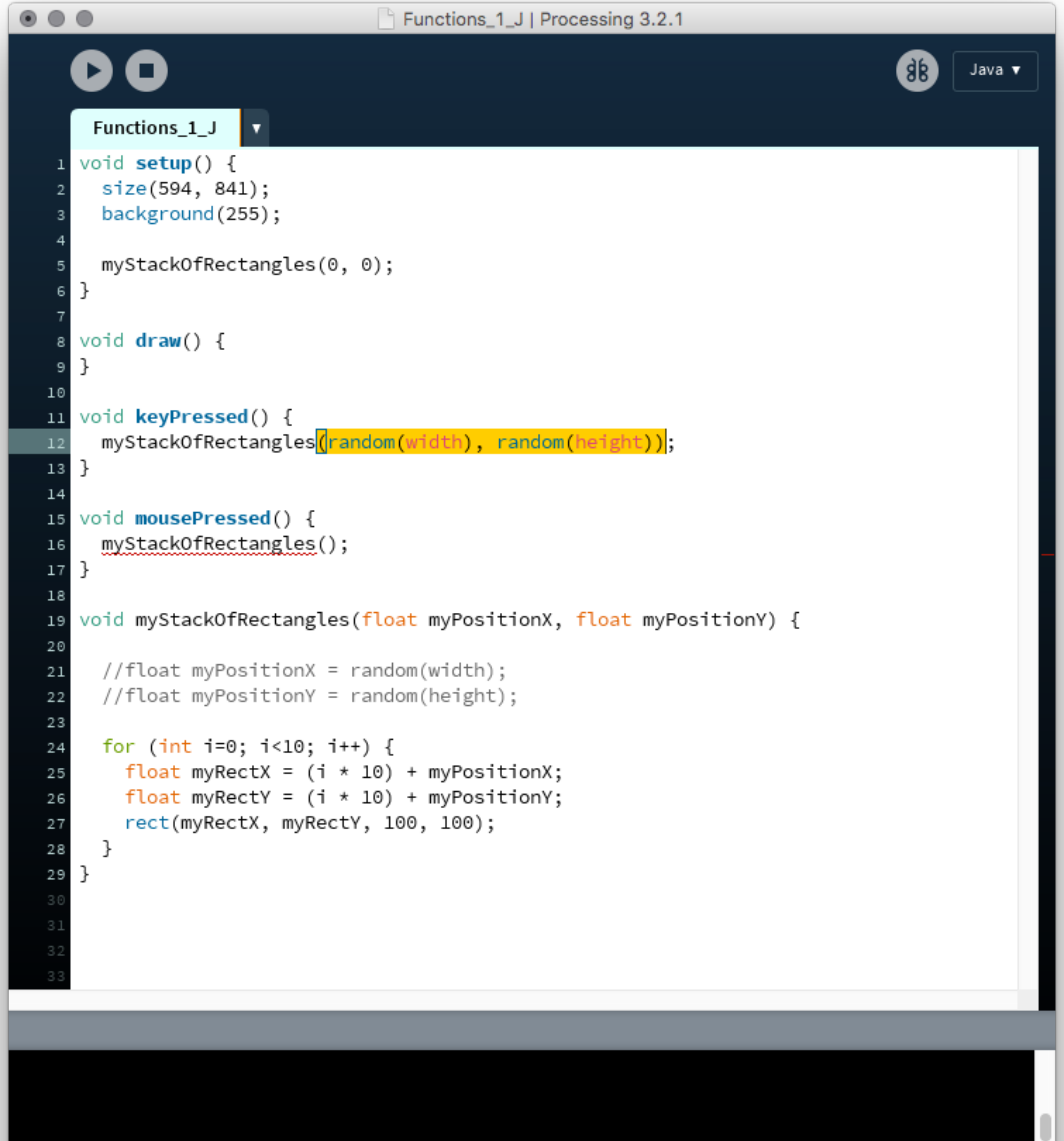
```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles();
}

void draw() {
}

void keyPressed() {
  myStackOfRectangles();
}

void mousePressed() {
  myStackOfRectangles();
}

void myStackOfRectangles(float myPositionX, float myPositionY) {

  //float myPositionX = random(width);
  //float myPositionY = random(height);

  for (int i=0; i<10; i++) {
    float myRectX = (i * 10) + myPositionX;
    float myRectY = (i * 10) + myPositionY;
    rect(myRectX, myRectY, 100, 100);
  }
}
```

We can have the function accept these variables as parameters.



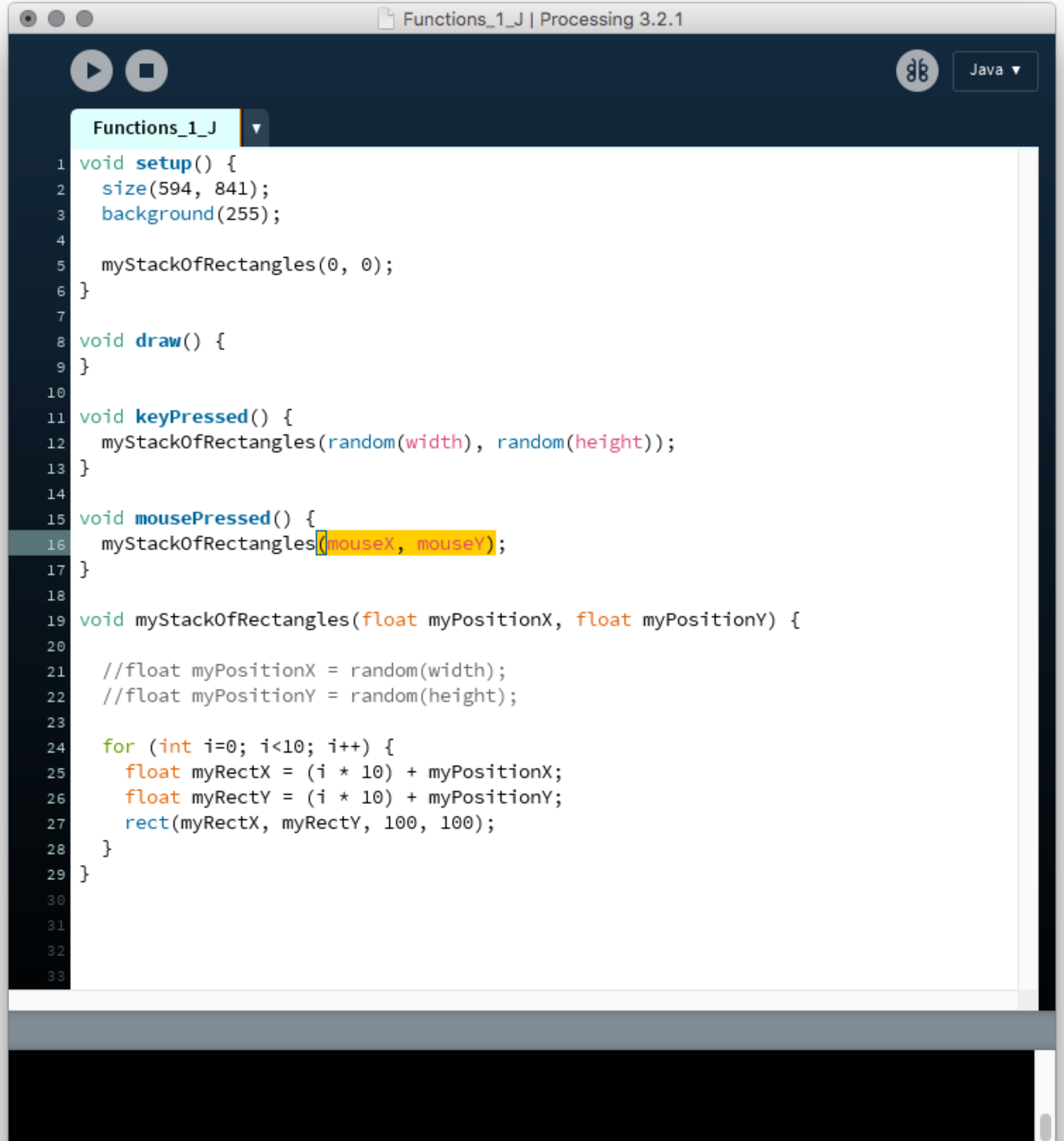
```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles(0, 0);
}

void draw() {
}

void keyPressed() {
  myStackOfRectangles();
}

void mousePressed() {
  myStackOfRectangles();
}

void myStackOfRectangles(float myPositionX, float myPositionY) {

  //float myPositionX = random(width);
  //float myPositionY = random(height);

  for (int i=0; i<10; i++) {
    float myRectX = (i * 10) + myPositionX;
    float myRectY = (i * 10) + myPositionY;
    rect(myRectX, myRectY, 100, 100);
  }
}
```

In setup we pass 0 for mousePositionX and 0 for mousePositionY.



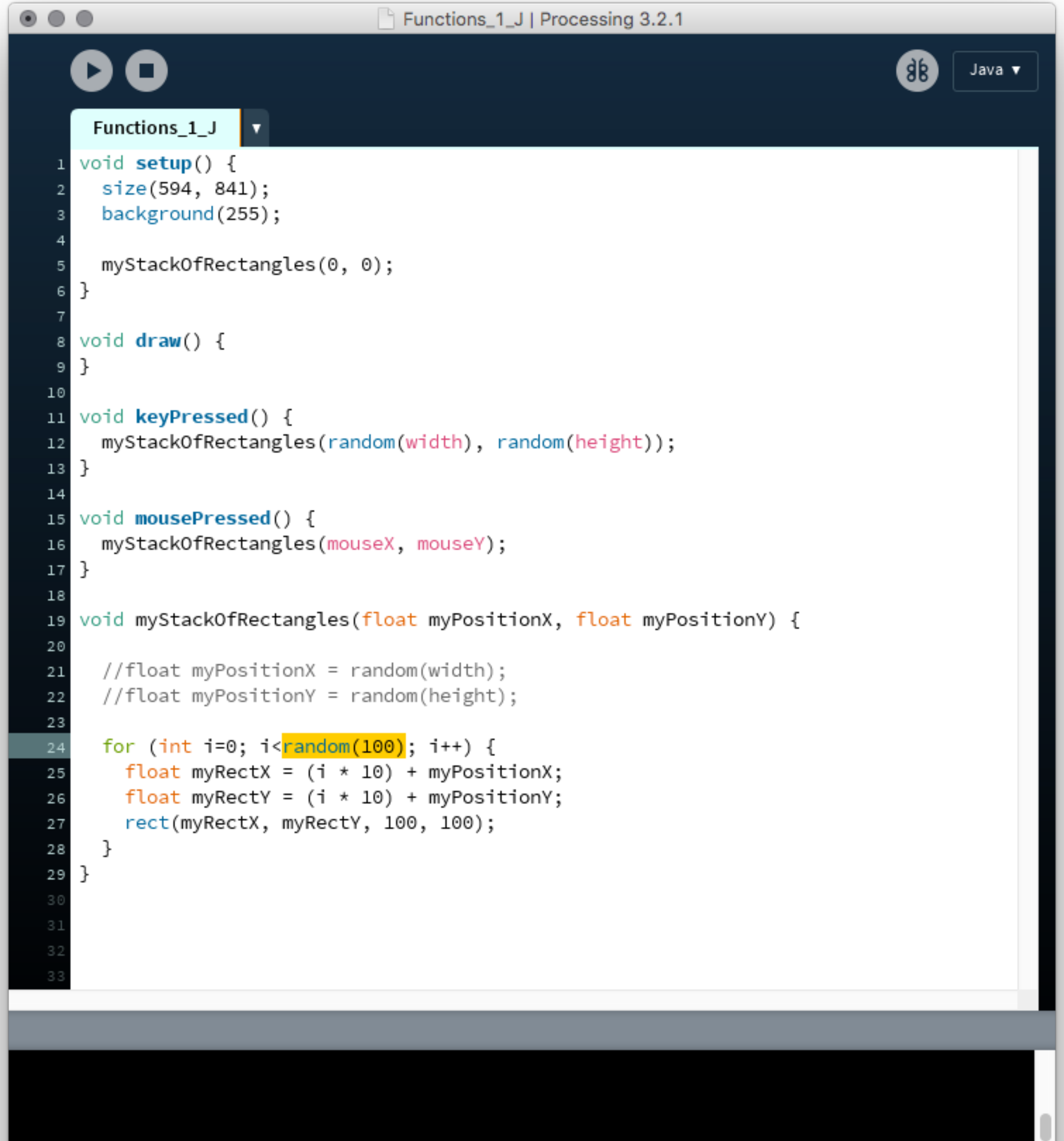
```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles(0, 0);
}

void draw() {
}

void keyPressed() {
  myStackOfRectangles(random(width), random(height));
}

void mousePressed() {
  myStackOfRectangles();
}

void myStackOfRectangles(float myPositionX, float myPositionY) {

  //float myPositionX = random(width);
  //float myPositionY = random(height);

  for (int i=0; i<10; i++) {
    float myRectX = (i * 10) + myPositionX;
    float myRectY = (i * 10) + myPositionY;
    rect(myRectX, myRectY, 100, 100);
  }
}
```

When a key is pressed, we pass random numbers for these values.



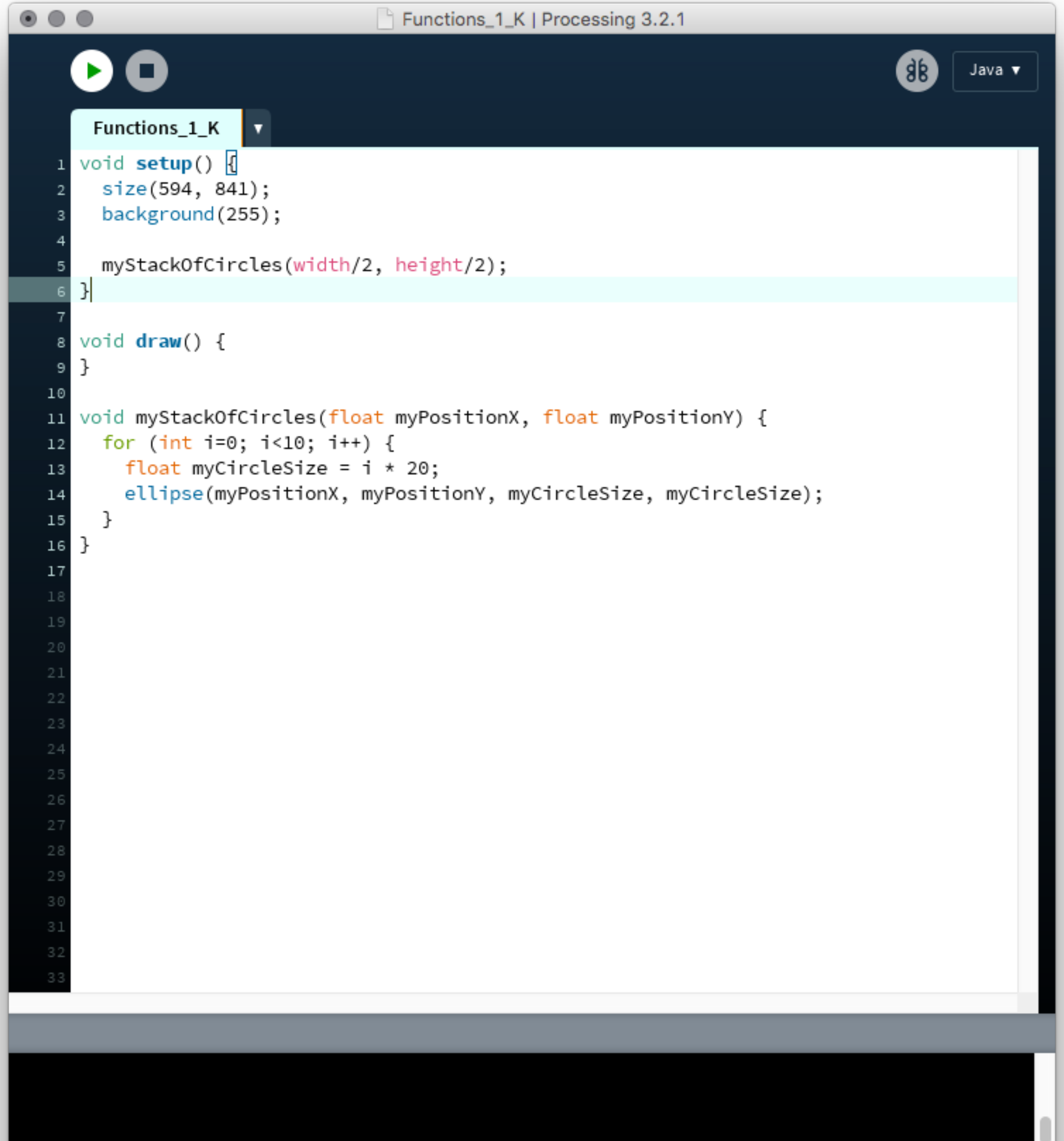
```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles(0, 0);
}

void draw() {
}

void keyPressed() {
  myStackOfRectangles(random(width), random(height));
}

void mousePressed() {
  myStackOfRectangles(mouseX, mouseY);
}

void myStackOfRectangles(float myPositionX, float myPositionY) {

  //float myPositionX = random(width);
  //float myPositionY = random(height);

  for (int i=0; i<10; i++) {
    float myRectX = (i * 10) + myPositionX;
    float myRectY = (i * 10) + myPositionY;
    rect(myRectX, myRectY, 100, 100);
  }
}
```

When the mouse is pressed we pass the mouse position for these values.



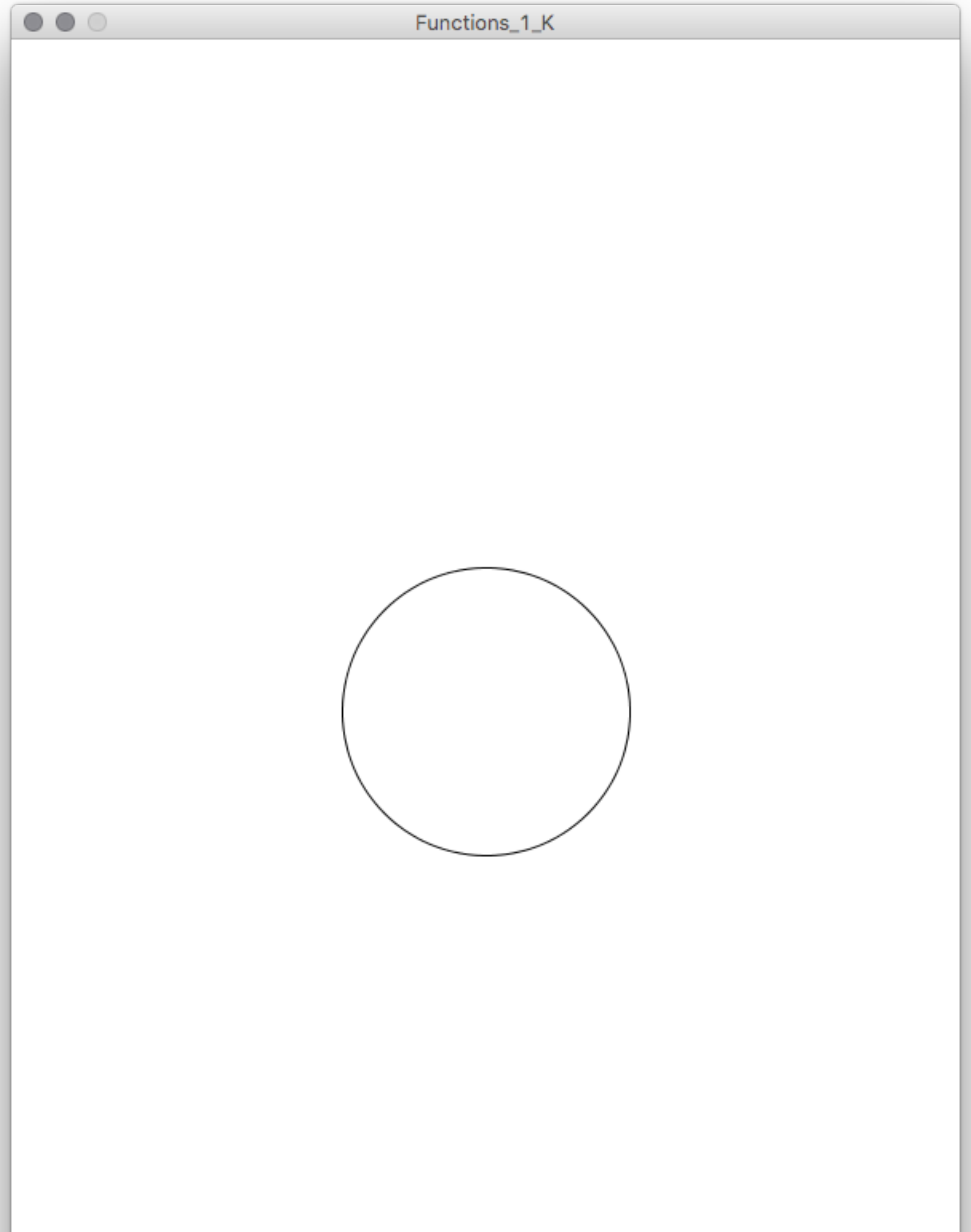
```
void setup() {
  size(594, 841);
  background(255);

  myStackOfRectangles(0, 0);
}

void draw() {
}

void keyPressed() {
  myStackOfRectangles(random(width), random(height));
}

void mousePressed() {
  myStackOfRectangles(mouseX, mouseY);
}

void myStackOfRectangles(float myPositionX, float myPositionY) {

  //float myPositionX = random(width);
  //float myPositionY = random(height);

  for (int i=0; i<random(100); i++) {
    float myRectX = (i * 10) + myPositionX;
    float myRectY = (i * 10) + myPositionY;
    rect(myRectX, myRectY, 100, 100);
  }
}
```

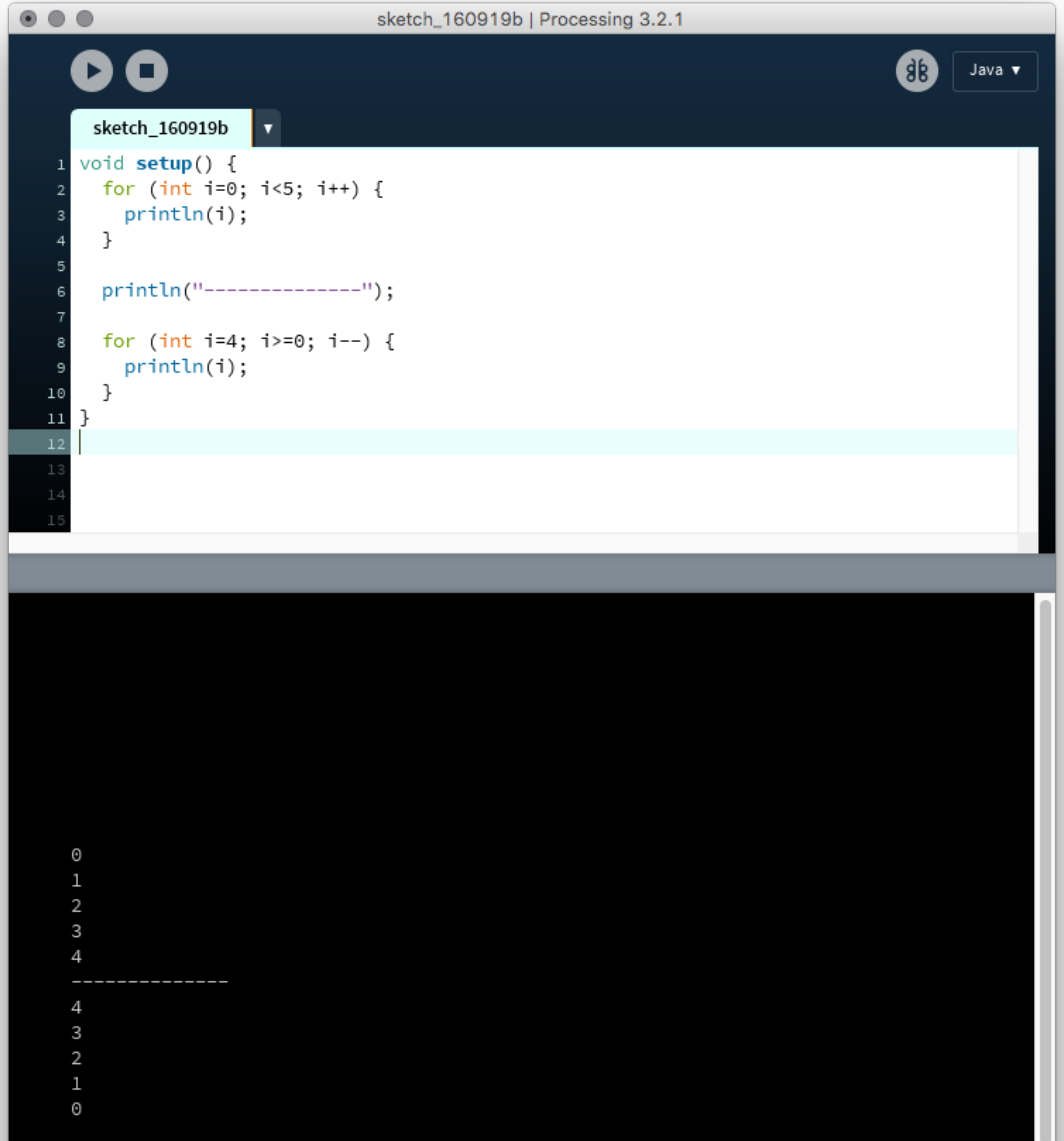
Randomize the number of rectangles drawn.



```
void setup() {
  size(594, 841);
  background(255);

  myStackOfCircles(width/2, height/2);
}

void draw() {
}

void myStackOfCircles(float myPositionX, float myPositionY) {
  for (int i=0; i<10; i++) {
    float myCircleSize = i * 20;
    ellipse(myPositionX, myPositionY, myCircleSize, myCircleSize);
  }
}
```

In this example, 10 circles are drawn but we only see one.

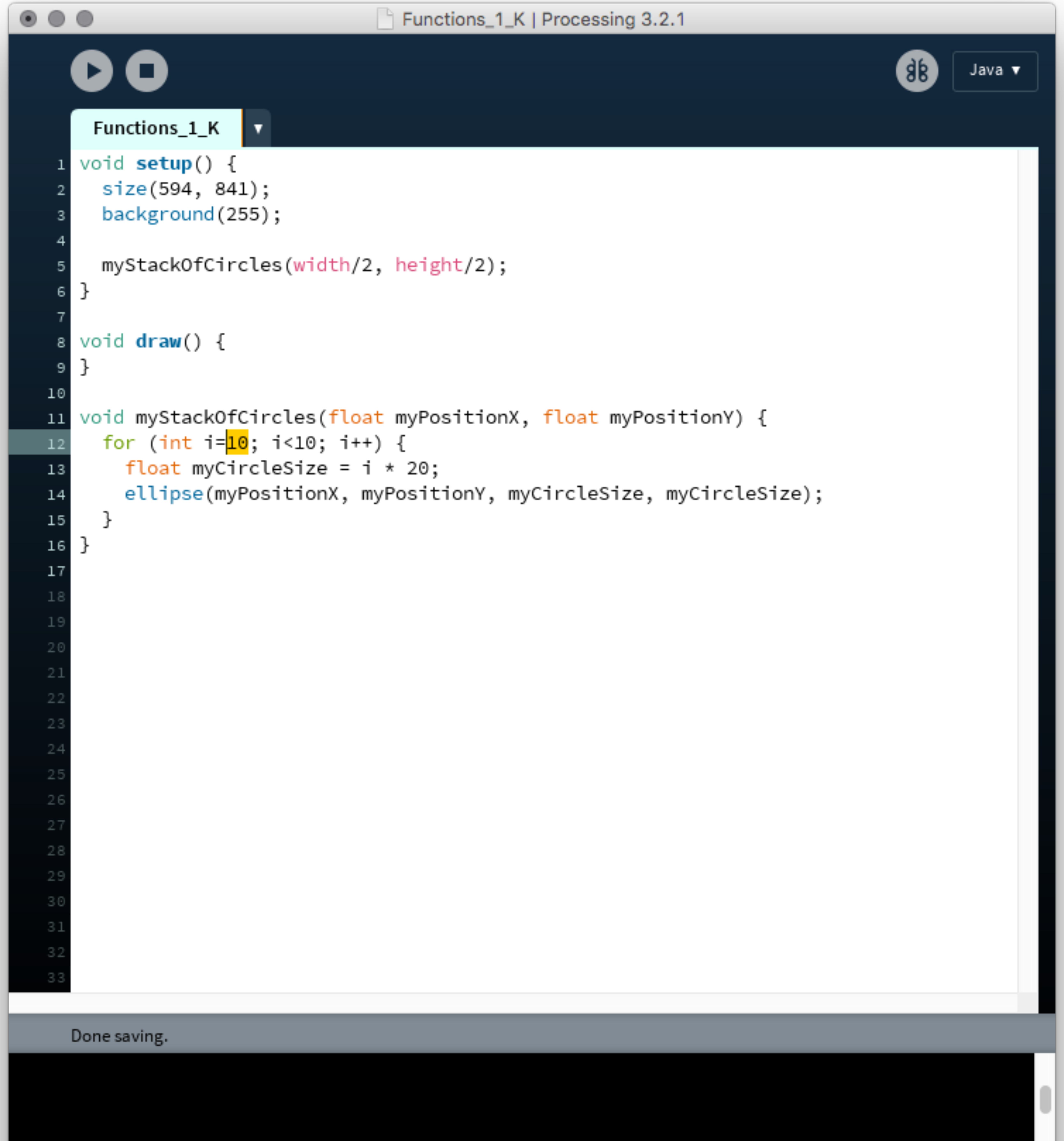




```
void setup() {
  for (int i=0; i<5; i++) {
    println(i);
  }

  println("-------------");

  for (int i=4; i>=0; i--) {
    println(i);
  }
}
```

```
0
1
2
3
4
-------------
4
3
2
1
0
```

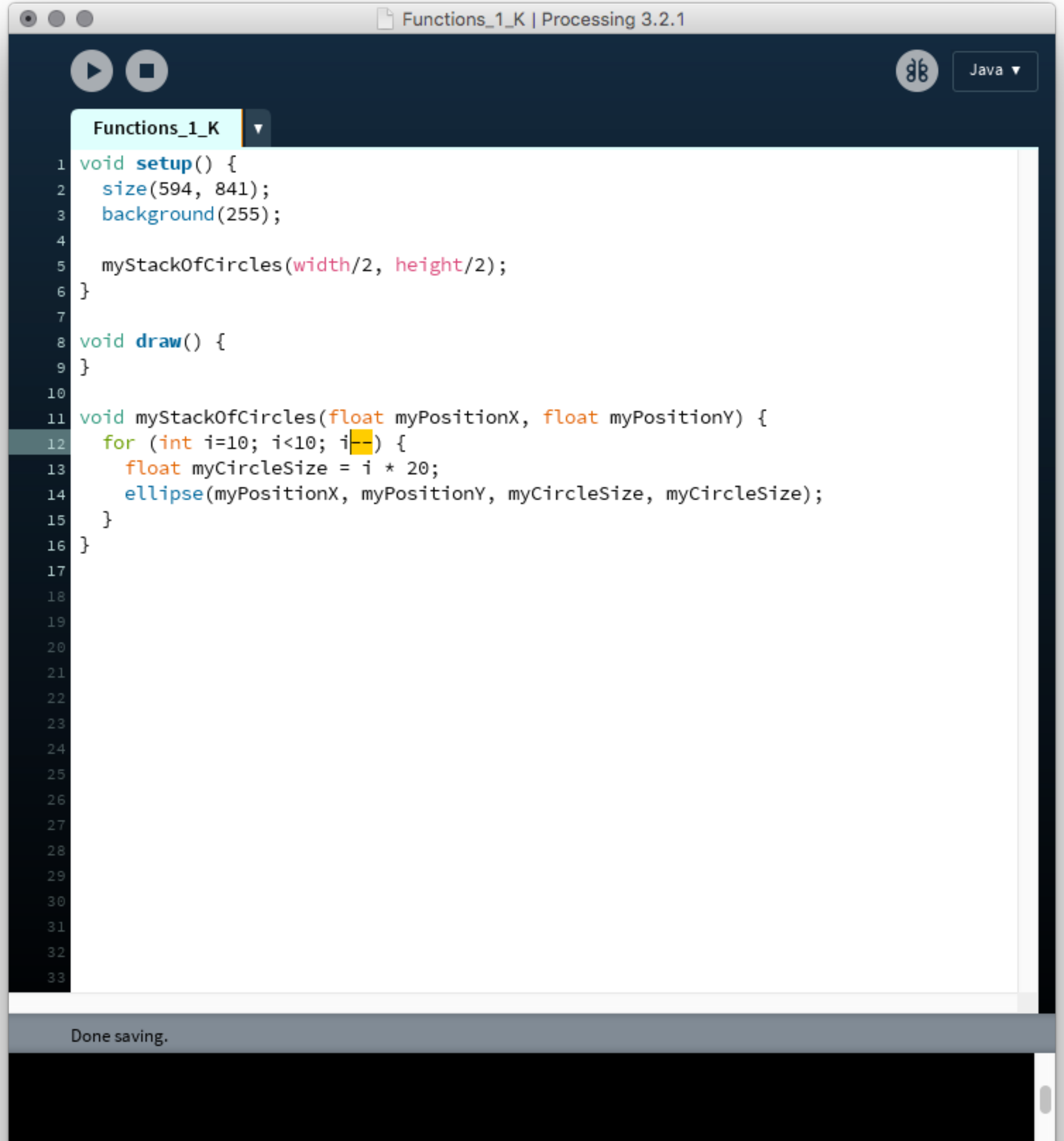
For loops can count up or down.



```
void setup() {
  size(594, 841);
  background(255);

  myStackOfCircles(width/2, height/2);
}

void draw() {
}

void myStackOfCircles(float myPositionX, float myPositionY) {
  for (int i=10; i<10; i++) {
    float myCircleSize = i * 20;
    ellipse(myPositionX, myPositionY, myCircleSize, myCircleSize);
  }
}
```

Done saving.

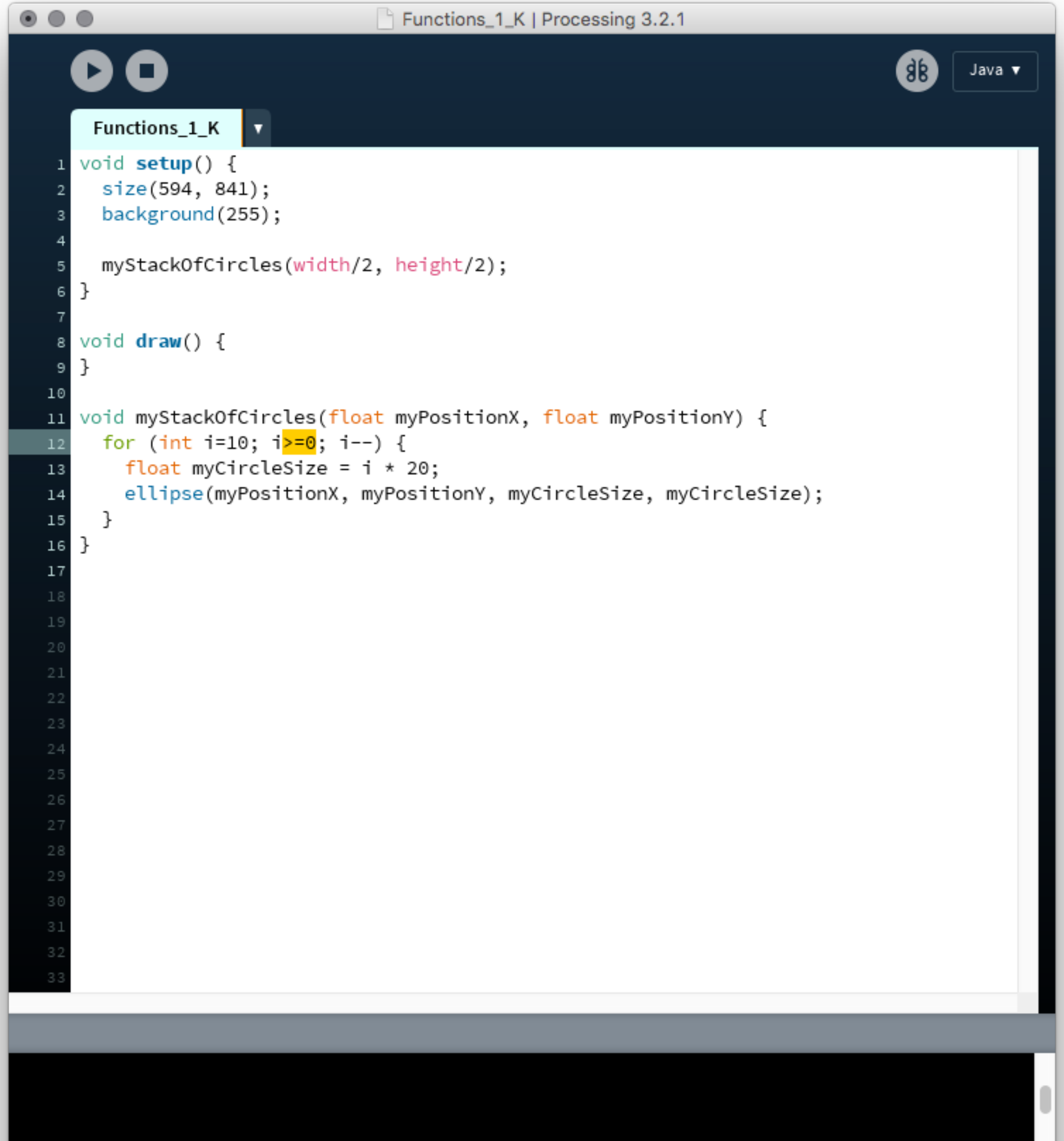
To do this, i should start at the high number.



```
void setup() {
  size(594, 841);
  background(255);

  myStackOfCircles(width/2, height/2);
}

void draw() {
}

void myStackOfCircles(float myPositionX, float myPositionY) {
  for (int i=10; i<10; i--) {
    float myCircleSize = i * 20;
    ellipse(myPositionX, myPositionY, myCircleSize, myCircleSize);
  }
}
```

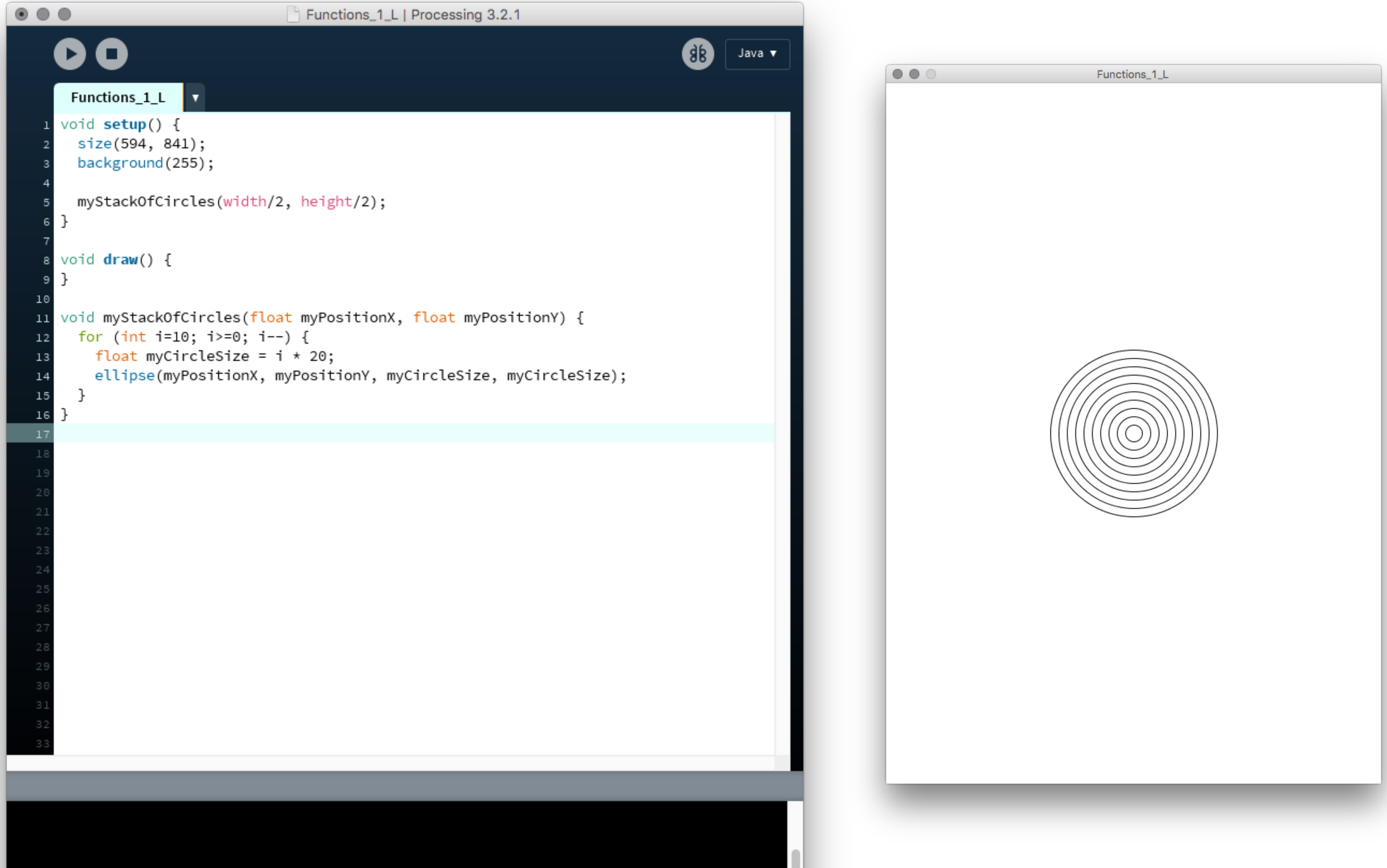
Done saving.

And decrease with each execution of the loop.



```
void setup() {
  size(594, 841);
  background(255);

  myStackOfCircles(width/2, height/2);
}

void draw() {
}

void myStackOfCircles(float myPositionX, float myPositionY) {
  for (int i=10; i>=0; i--) {
    float myCircleSize = i * 20;
    ellipse(myPositionX, myPositionY, myCircleSize, myCircleSize);
  }
}
```

And continue while i is greater than or equal to zero.



```
void setup() {
  size(594, 841);
  background(255);

  myStackOfCircles(width/2, height/2);
}

void draw() {
}

void myStackOfCircles(float myPositionX, float myPositionY) {
  for (int i=10; i>=0; i--) {
    float myCircleSize = i * 20;
    ellipse(myPositionX, myPositionY, myCircleSize, myCircleSize);
  }
}
```