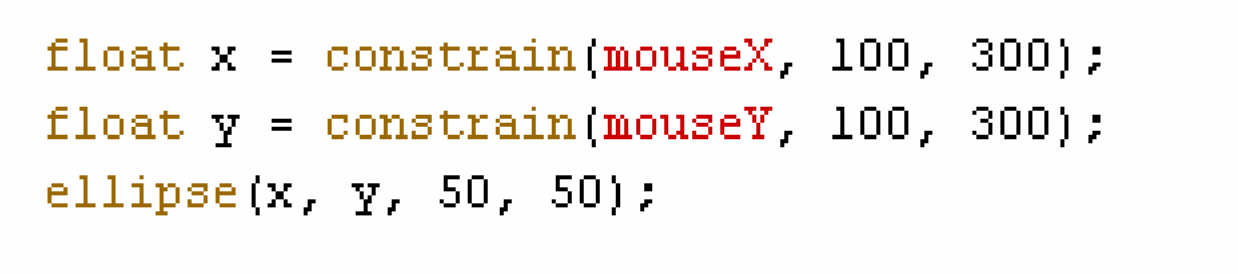
Processing Workshop

Till Nagel, IUAV, 10/2008

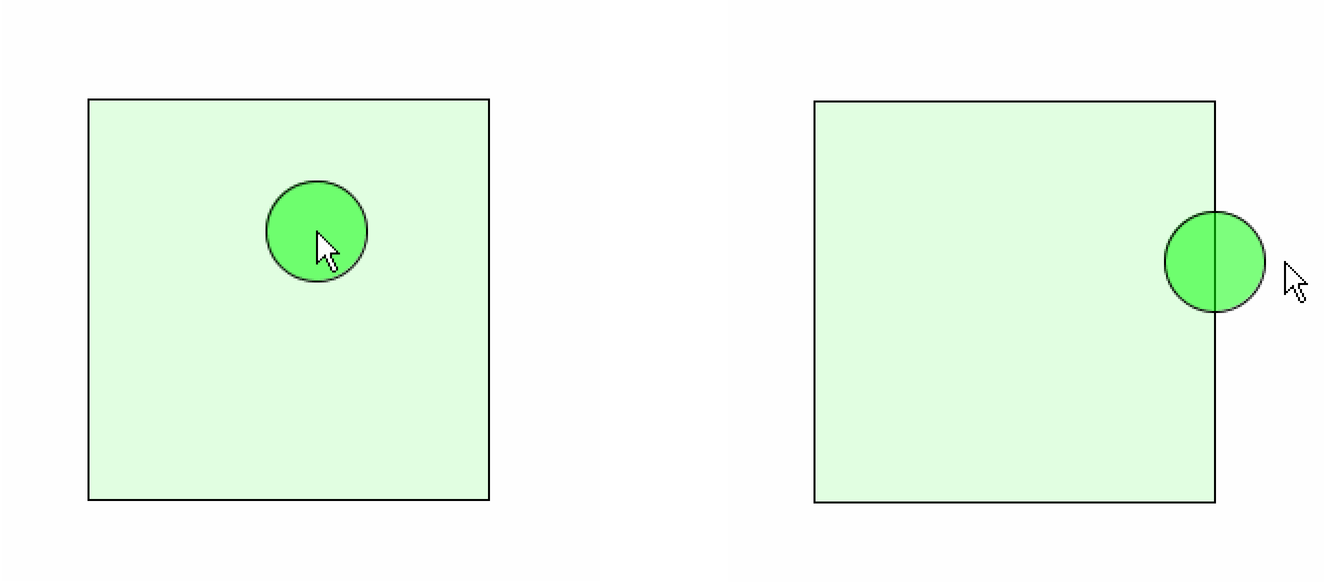
Mouse dragging

Example: SimpleDrag

constrain()

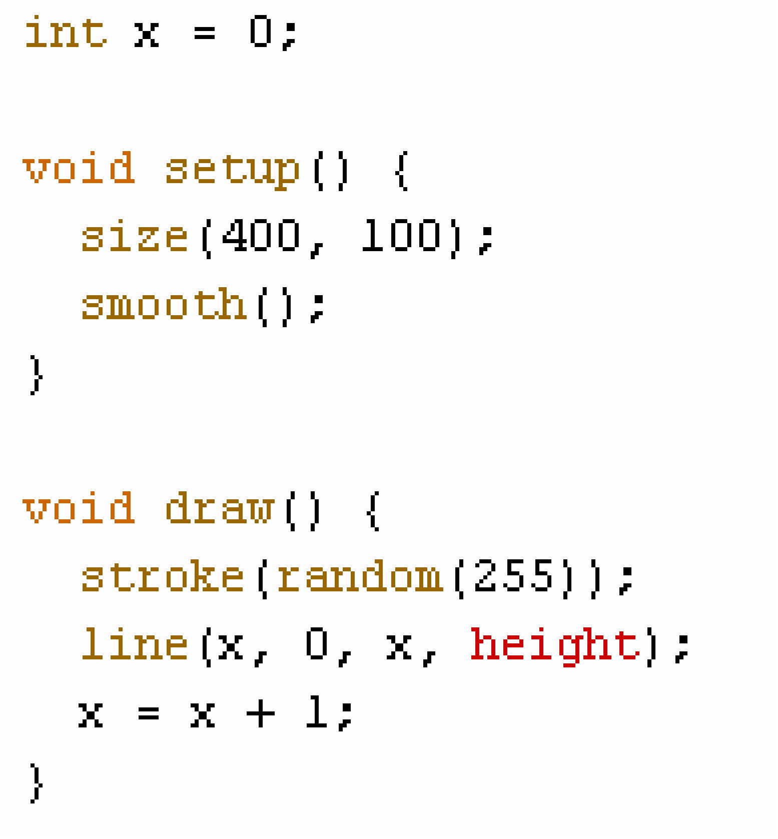


```
float x = constrain(mouseX, 100, 300);
float y = constrain(mouseY, 100, 300);
ellipse(x, y, 50, 50);
```



Example: Constrain

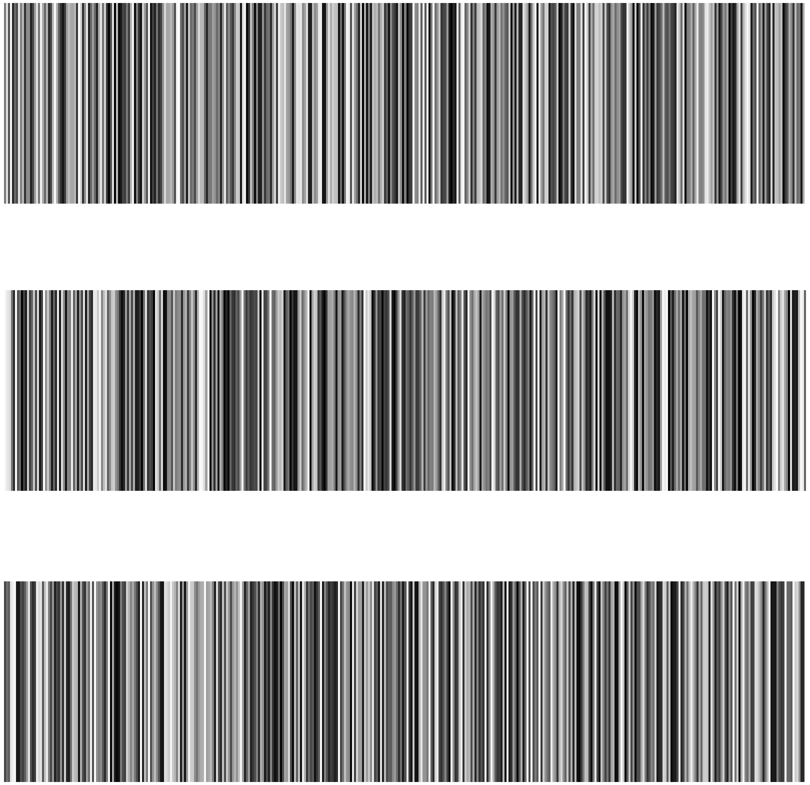
random(value)



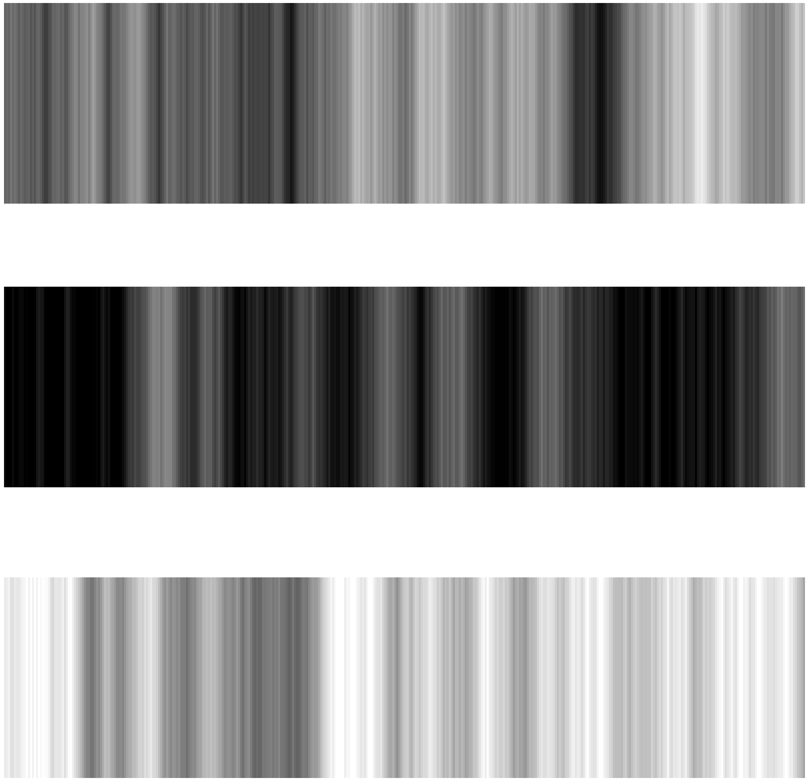
```
int x = 0;

void setup() {
  size(400, 100);
  smooth();
}

void draw() {
  stroke(random(255));
  line(x, 0, x, height);
  x = x + 1;
}
```



random(value)



Example: SoftRandomLines

Iteration

Simplify and compact repetitive code.

Decrease the length of code

Enhance maintainability and managability

Reduce errors

Iteration

```
line(10, 10, 50, 10);
line(10, 20, 50, 20);
line(10, 30, 50, 30);
line(10, 40, 50, 40);
line(10, 50, 50, 50);
line(10, 60, 50, 60);
line(10, 70, 50, 70);
line(10, 80, 50, 80);
line(10, 90, 50, 90);
```

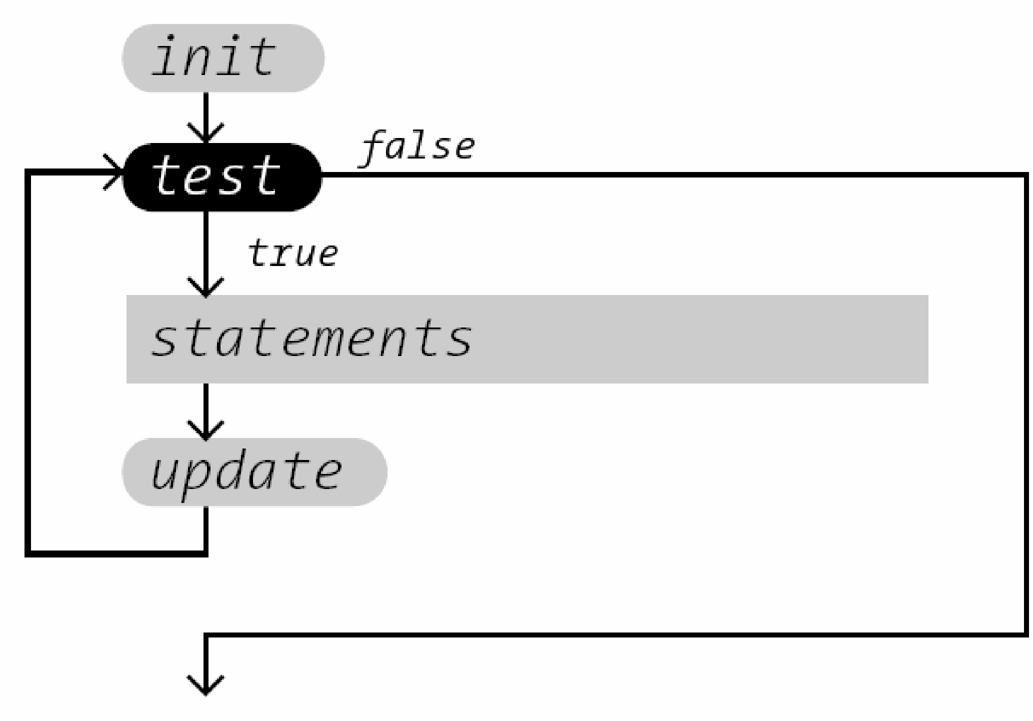
for loop

```
line(10, 10, 50, 10);
line(10, 20, 50, 20);
line(10, 30, 50, 30);
line(10, 40, 50, 40);
line(10, 50, 50, 50);
line(10, 60, 50, 60);
line(10, 70, 50, 70);
line(10, 80, 50, 80);
line(10, 90, 50, 90);
```

```
for (int i = 10; i < 100; i = i + 10) {
  line(10, i, 50, i);
}
```

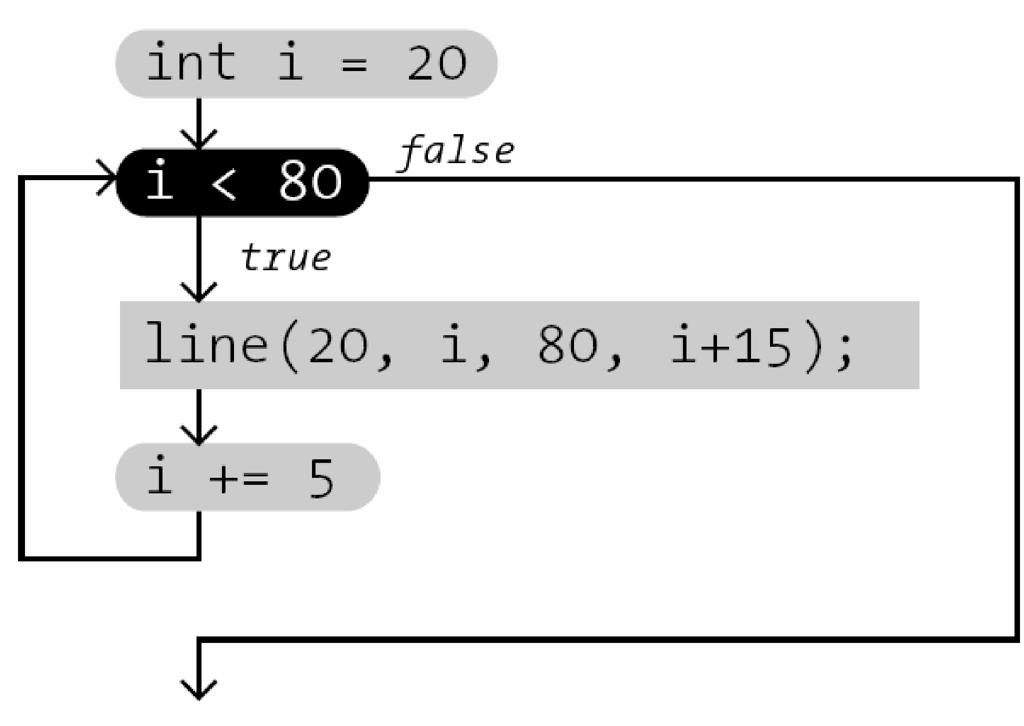
for loop

```
for (init; test; update) {
   statements
}
```



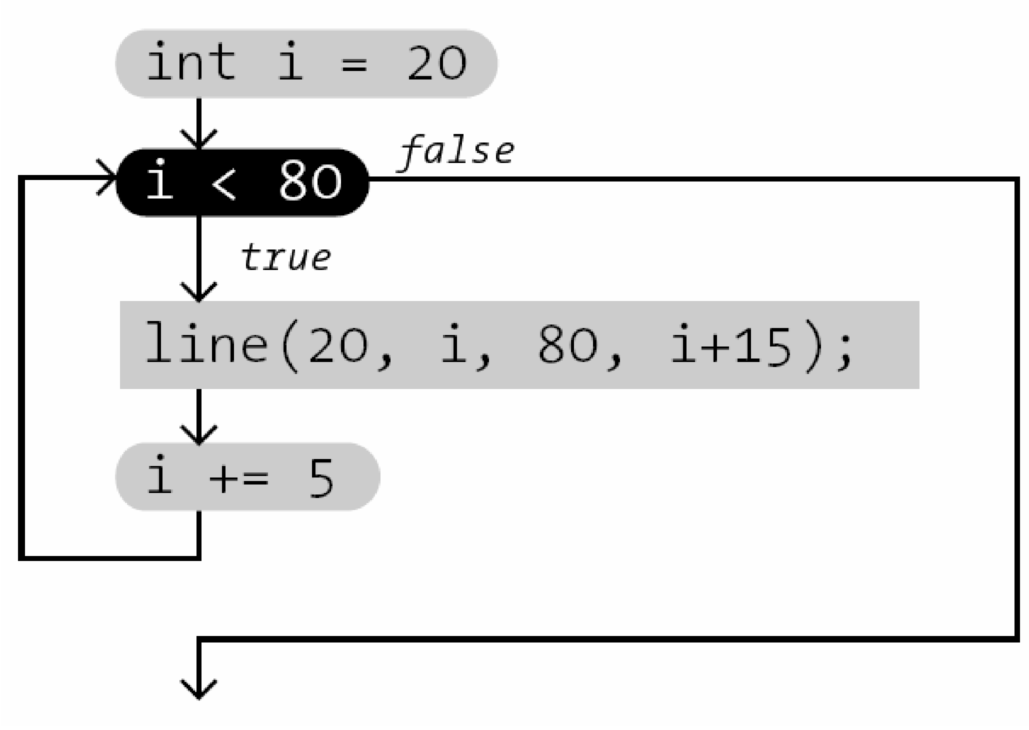
for loop

```
for (int i = 20; i < 80; i += 5) {
   line(20, i, 80, i + 15);
}
```

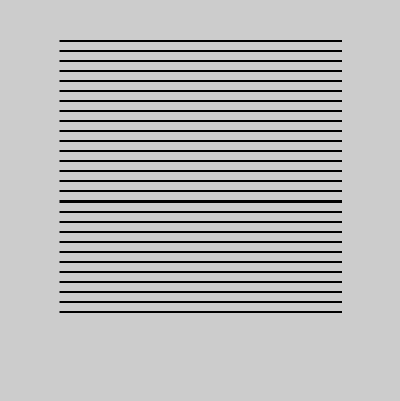


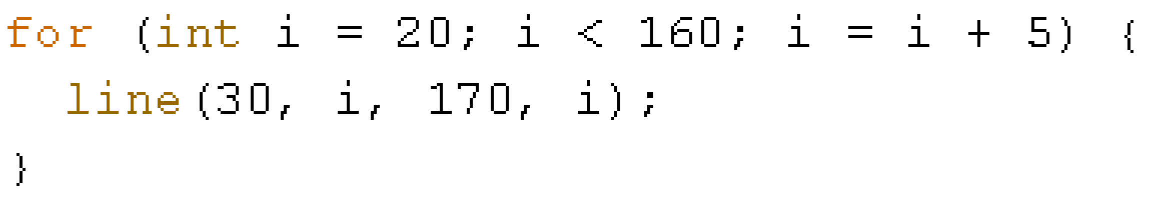
for loop

```
for (int i = 20; i < 80; i += 5) {
  line(20, i, 80, i + 15);
}
```

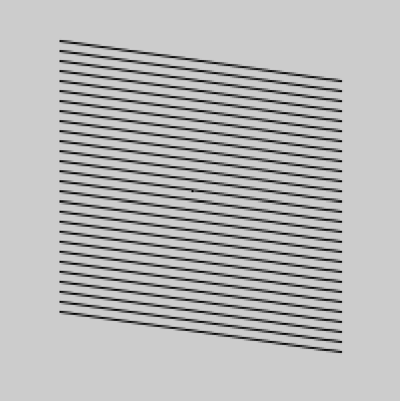


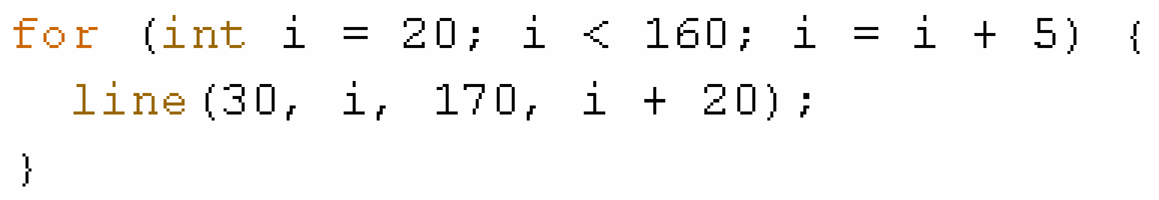
for loop





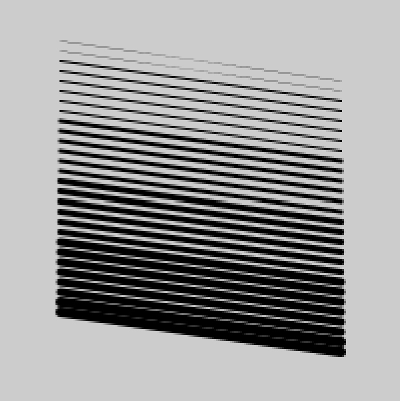
```
for (int i = 20; i < 160; i = i + 5) {
  line(30, i, 170, i);
}
```

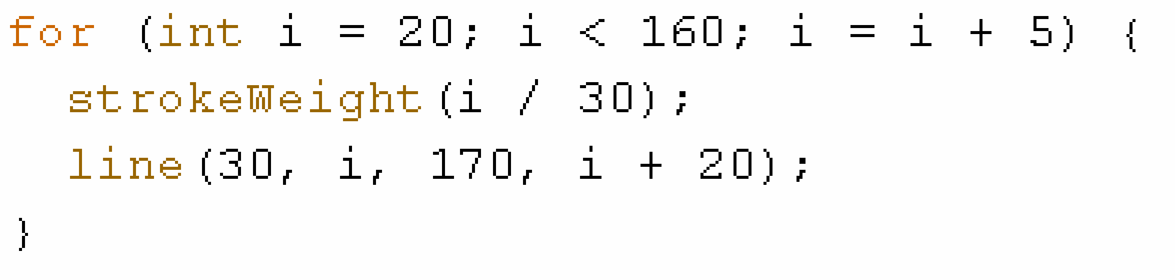




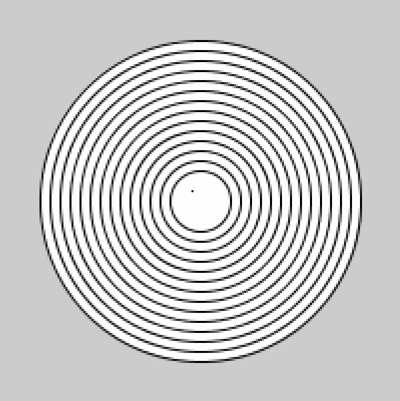
```
for (int i = 20; i < 160; i = i + 5) {
  line(30, i, 170, i + 20);
}
```

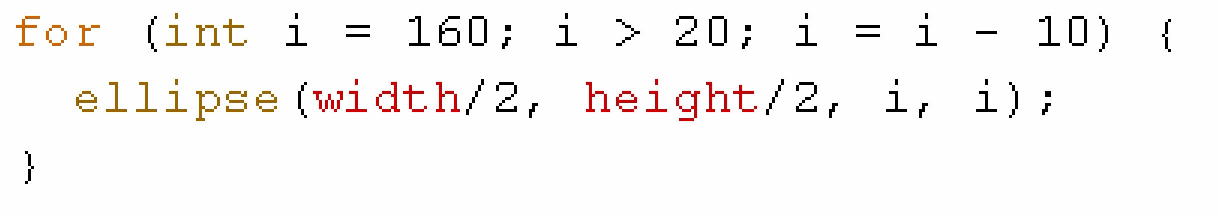
for loop





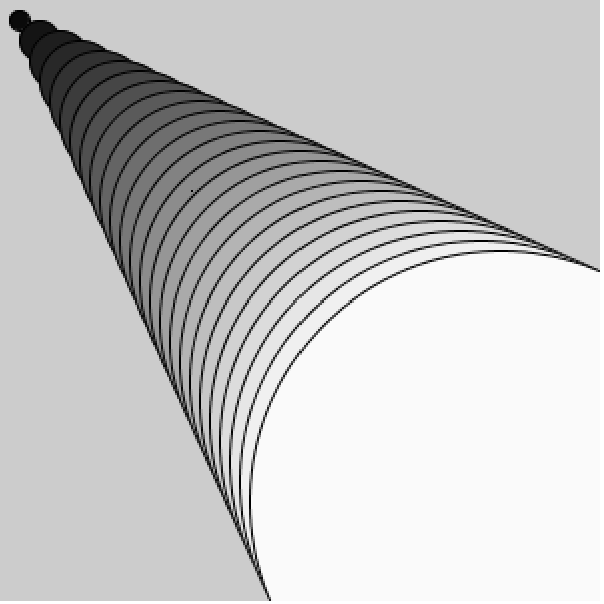
```
for (int i = 20; i < 160; i = i + 5) {
  strokeWeight(i / 30);
  line(30, i, 170, i + 20);
}
```

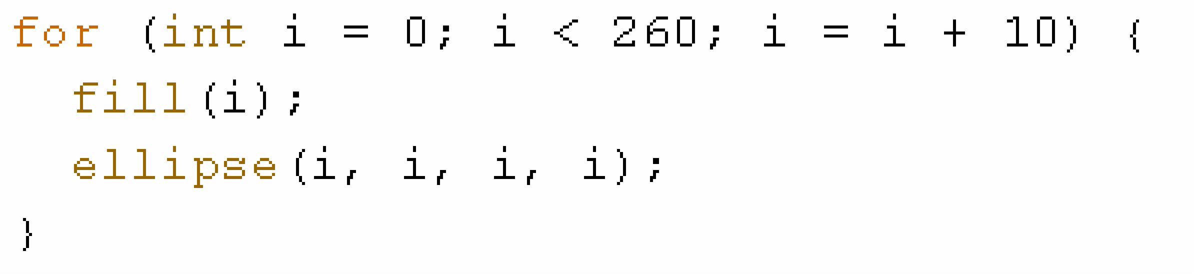




```
for (int i = 160; i > 20; i = i - 10) {
  ellipse(width/2, height/2, i, i);
}
```

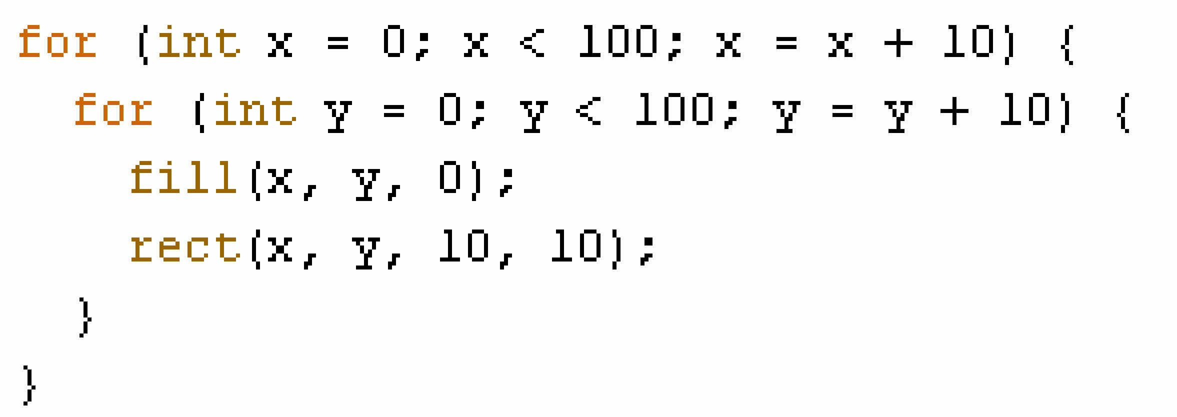
for loop



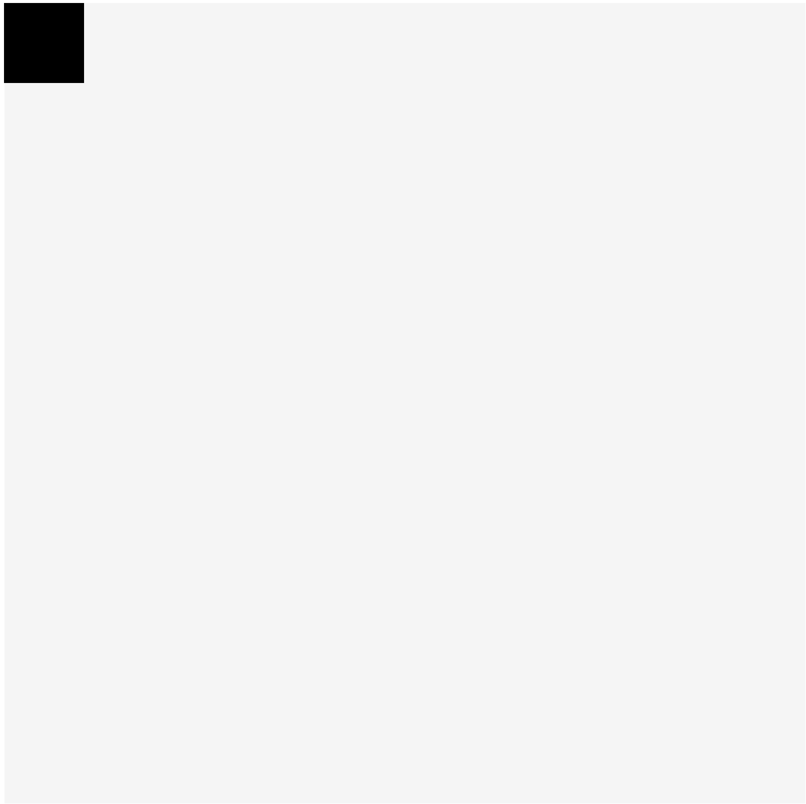


```
for (int i = 0; i < 260; i = i + 10) {
  fill(i);
  ellipse(i, i, i, i);
}
```

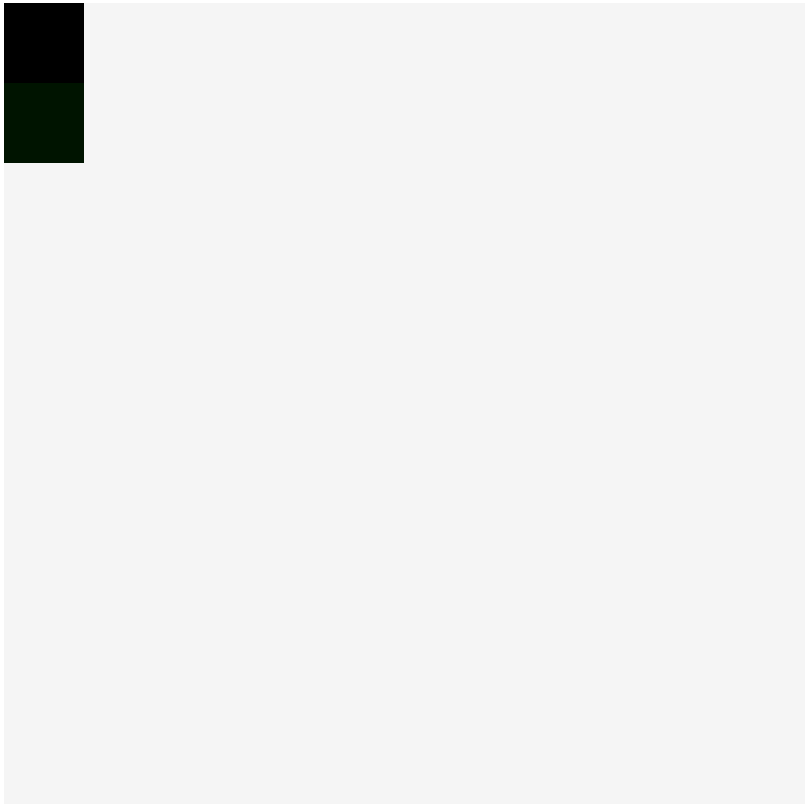
Nested loops

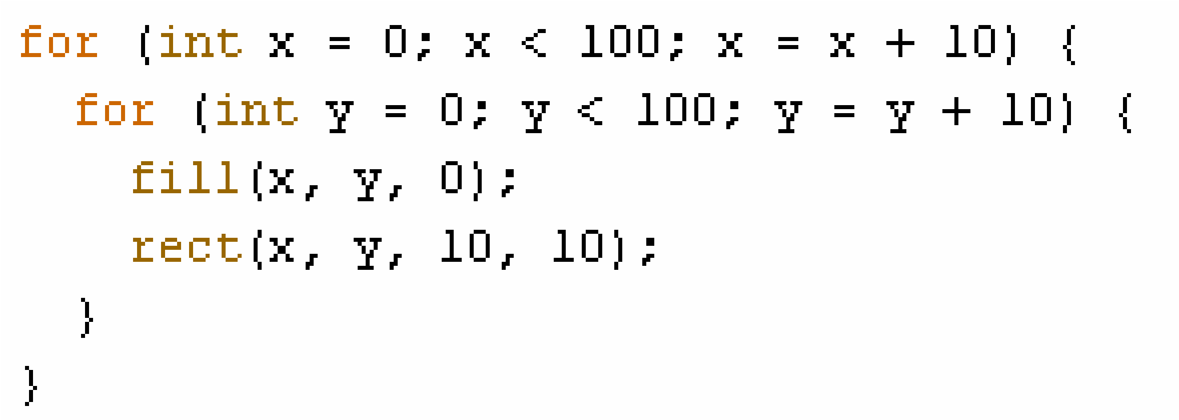


```
for (int x = 0; x < 100; x = x + 10) {
  for (int y = 0; y < 100; y = y + 10) {
    fill(x, y, 0);
    rect(x, y, 10, 10);
  }
}
```

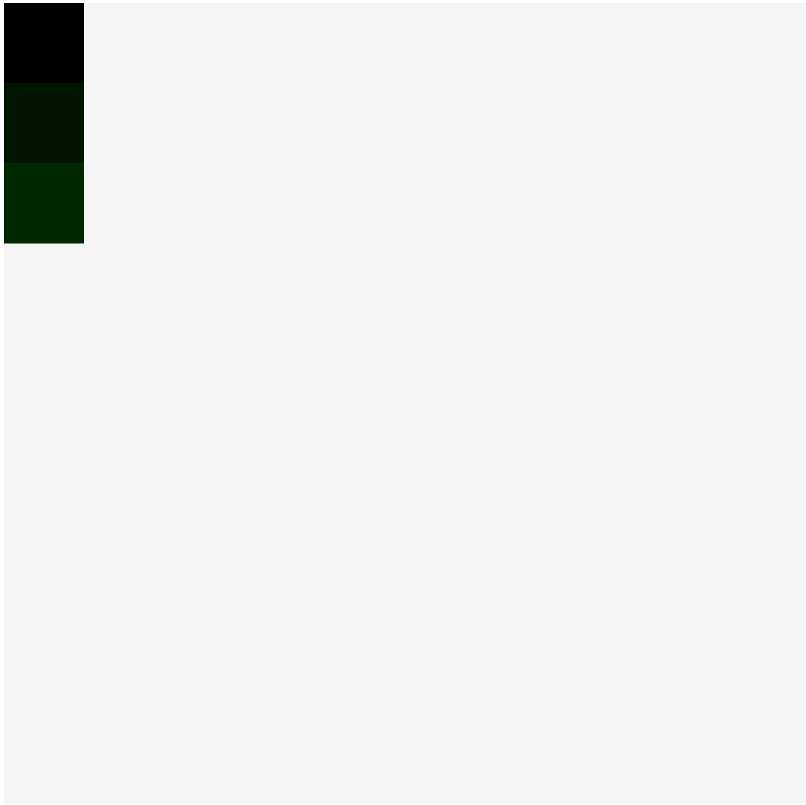


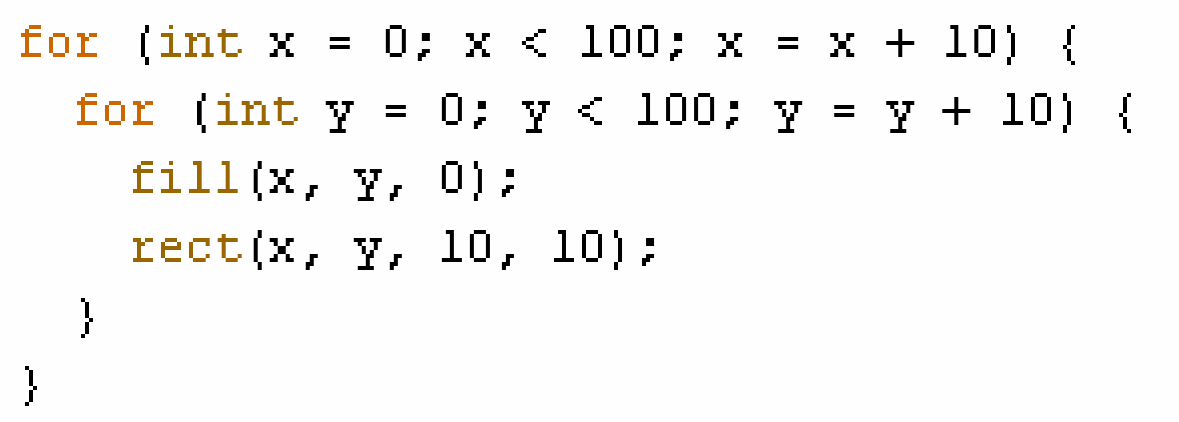
```
for (int x = 0; x < 100; x = x + 10) {
  for (int y = 0; y < 100; y = y + 10) {
    fill(x, y, 0);
    rect(x, y, 10, 10);
  }
}
```



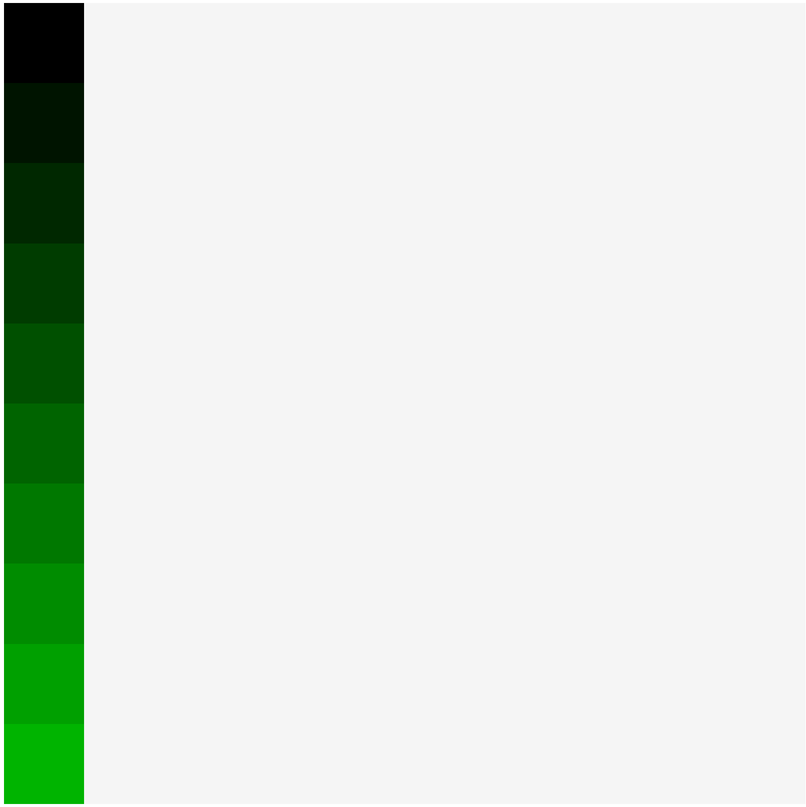


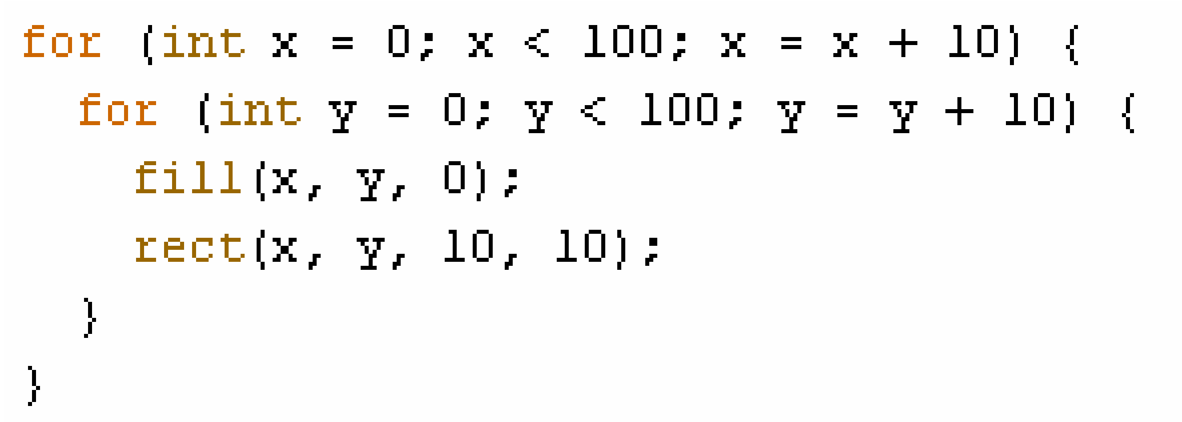
```
for (int x = 0; x < 100; x = x + 10) {
  for (int y = 0; y < 100; y = y + 10) {
    fill(x, y, 0);
    rect(x, y, 10, 10);
  }
}
```



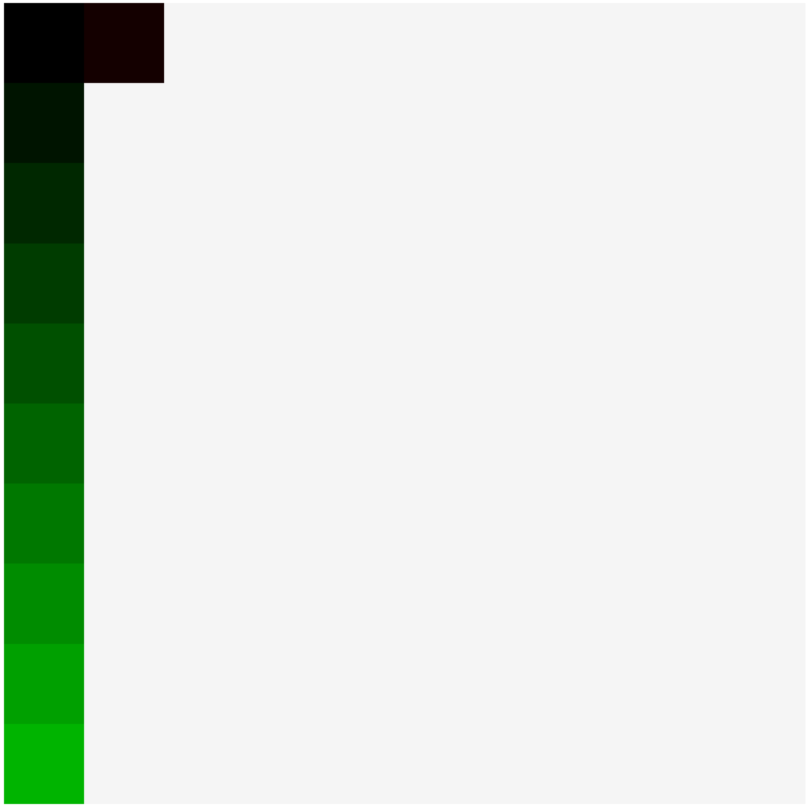


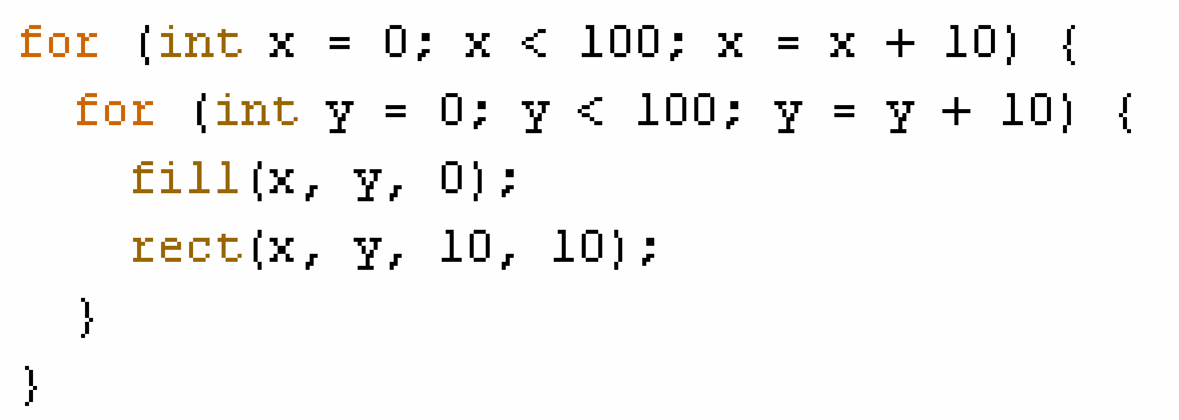
```
for (int x = 0; x < 100; x = x + 10) {
  for (int y = 0; y < 100; y = y + 10) {
    fill(x, y, 0);
    rect(x, y, 10, 10);
  }
}
```



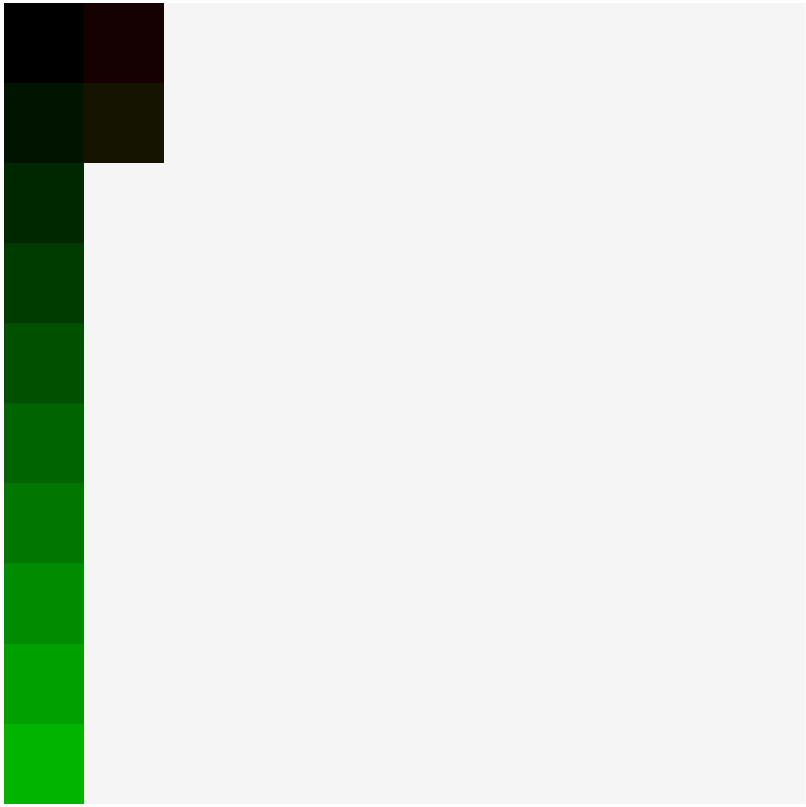


```
for (int x = 0; x < 100; x = x + 10) {
  for (int y = 0; y < 100; y = y + 10) {
    fill(x, y, 0);
    rect(x, y, 10, 10);
  }
}
```

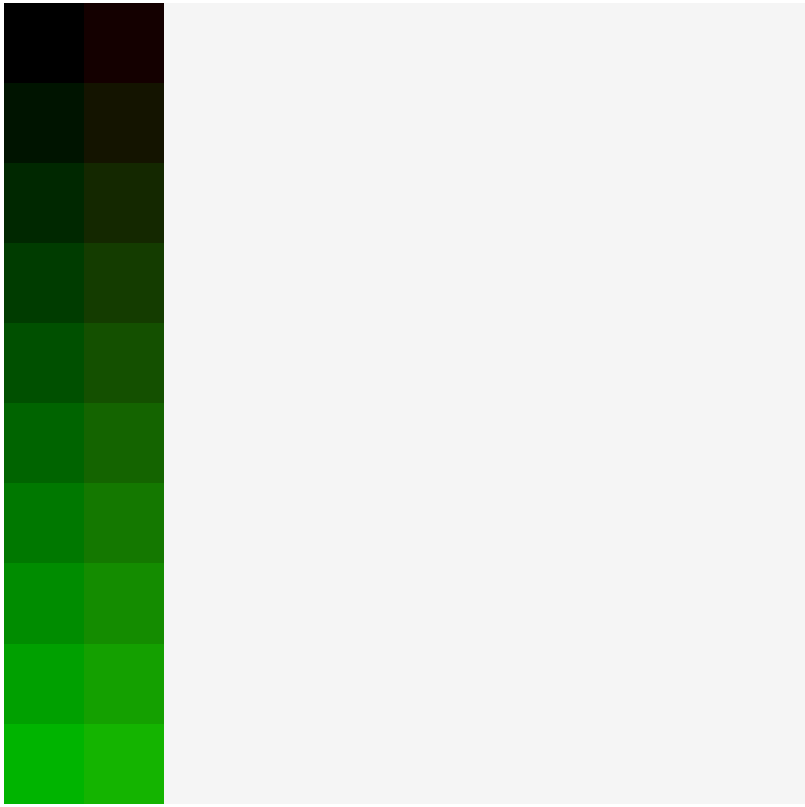


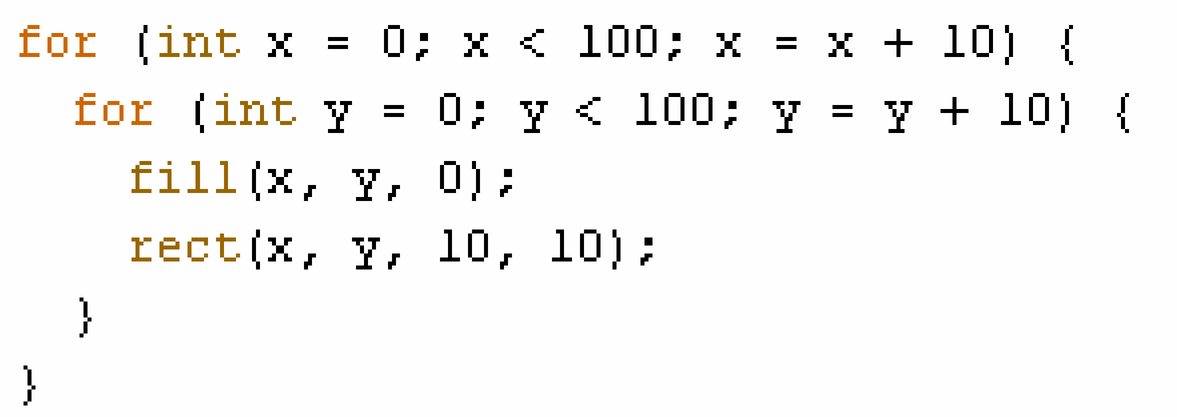


```
for (int x = 0; x < 100; x = x + 10) {
  for (int y = 0; y < 100; y = y + 10) {
    fill(x, y, 0);
    rect(x, y, 10, 10);
  }
}
```

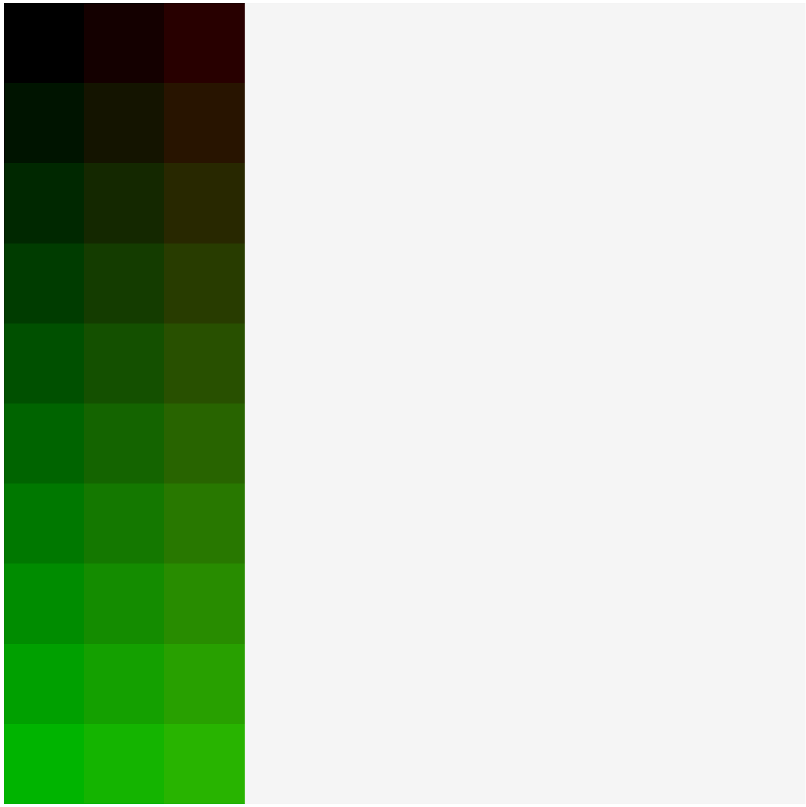


```
for (int x = 0; x < 100; x = x + 10) {
  for (int y = 0; y < 100; y = y + 10) {
    fill(x, y, 0);
    rect(x, y, 10, 10);
  }
}
```

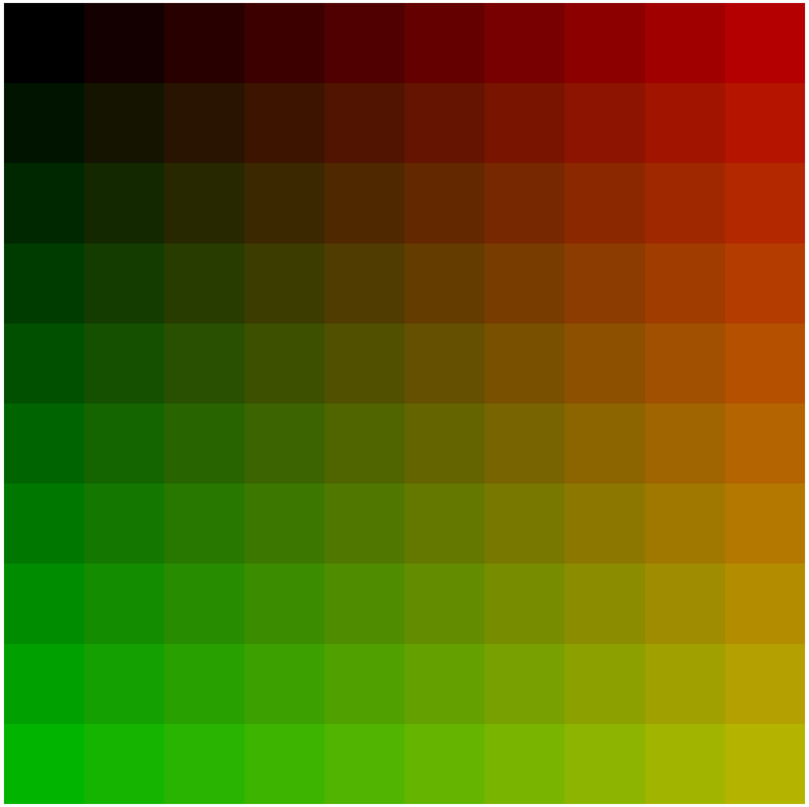




```
for (int x = 0; x < 100; x = x + 10) {
  for (int y = 0; y < 100; y = y + 10) {
    fill(x, y, 0);
    rect(x, y, 10, 10);
  }
}
```



```
for (int x = 0; x < 100; x = x + 10) {
  for (int y = 0; y < 100; y = y + 10) {
    fill(x, y, 0);
    rect(x, y, 10, 10);
  }
}
```



Exercises

E13: Create an interactive element, and use three different images or icons for the states normal, mouse-over and mouse-out.

Variation: Create an outer glow which responds to the proximity of the mouse pointer.
Lookup `dist()` in the Processing reference and use it.

Exercises

E14: Create various different 2D patterns. Choose three you like most and make screenshots.

Variation: Integrate randomness.

Variation: Make them interactive.