Generative Design II

Function

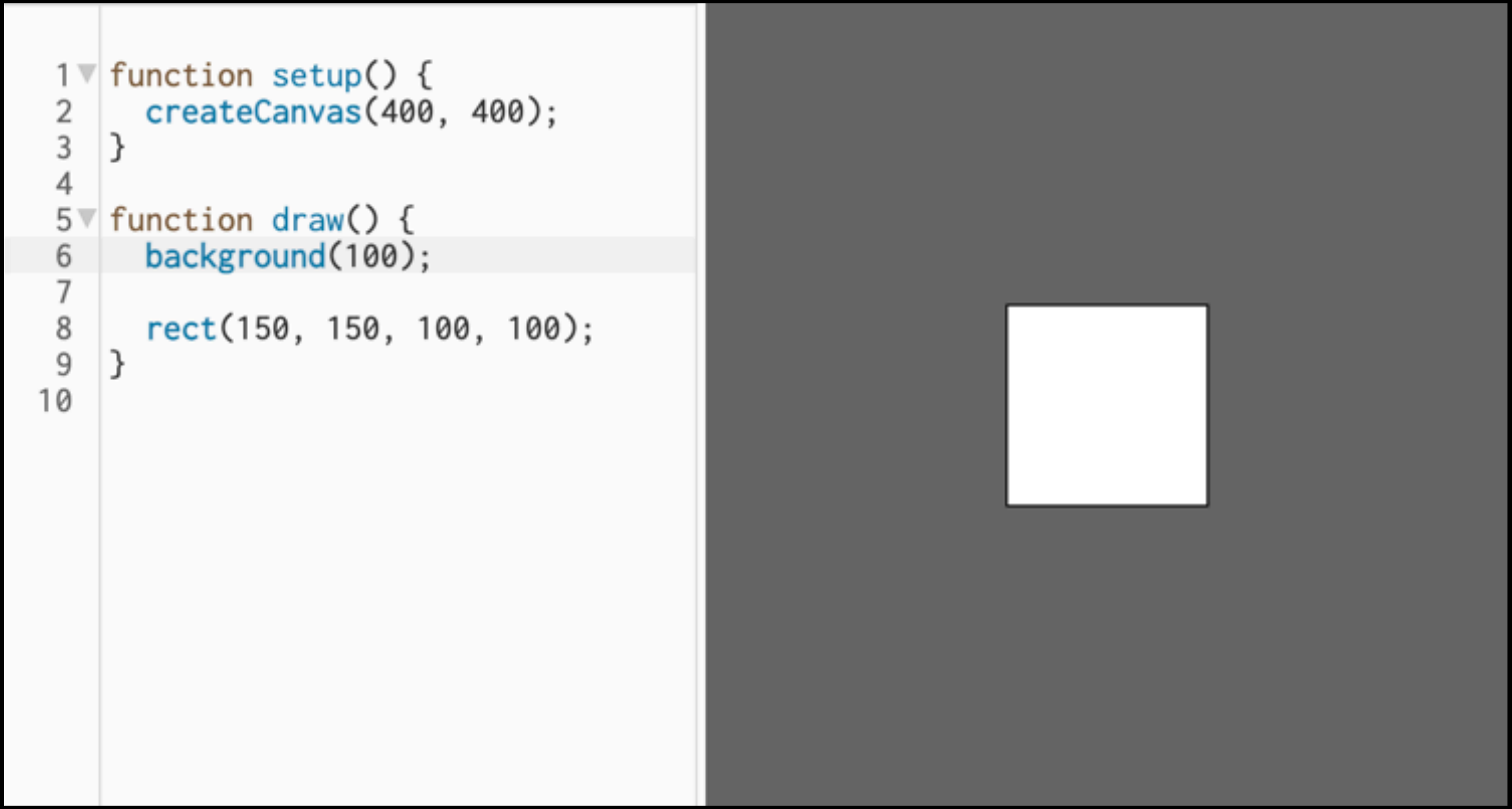
a sequence of program instructions that perform a specific task
특정한 작업을 수행하기 위한 지시들의 모임

```
createCanvas(500, 500);

rect(0, 0, 100, 100);

random(0, 100);
```

Function

```
function myFunction() {
   print("this is something");
   rect(0, 0, 100, 100);
}

myFunction(); // executes the function
```

Function

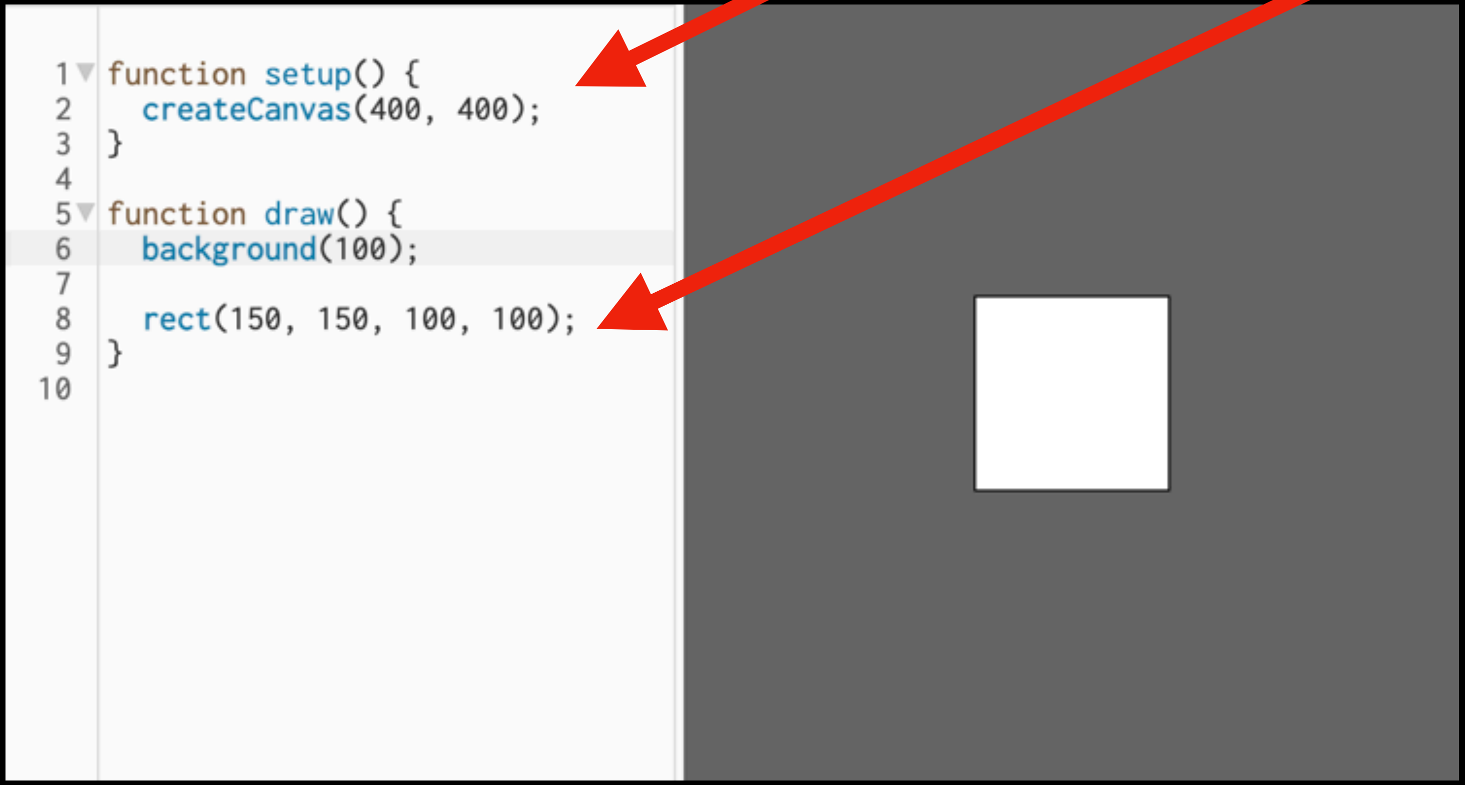


```
function setup() {
  createCanvas(400, 400);
}

function draw() {
  background(100);

  rect(150, 150, 100, 100);
}
```

Function

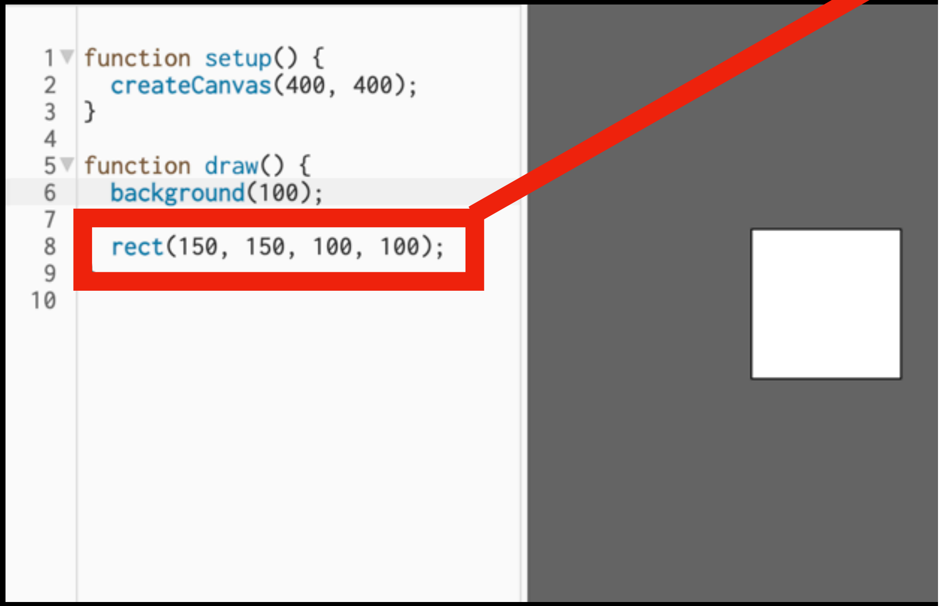
setup();

(1초에 60번씩)
draw();

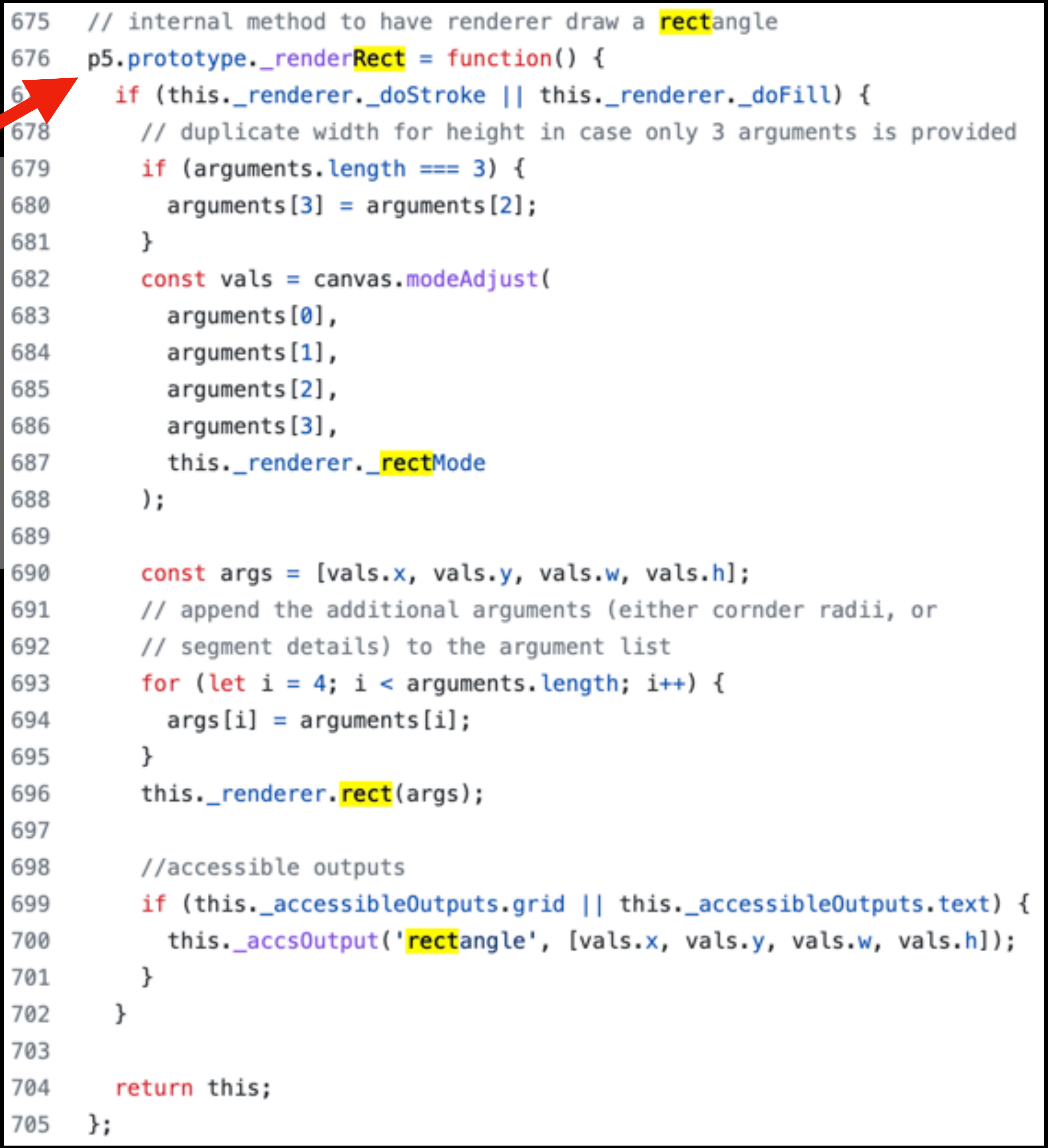


```
function setup() {
  createCanvas(400, 400);
}

function draw() {
  background(100);

  rect(150, 150, 100, 100);
}
```

Function



```
function setup() {
  createCanvas(400, 400);
}

function draw() {
  background(100);

  rect(150, 150, 100, 100);

```

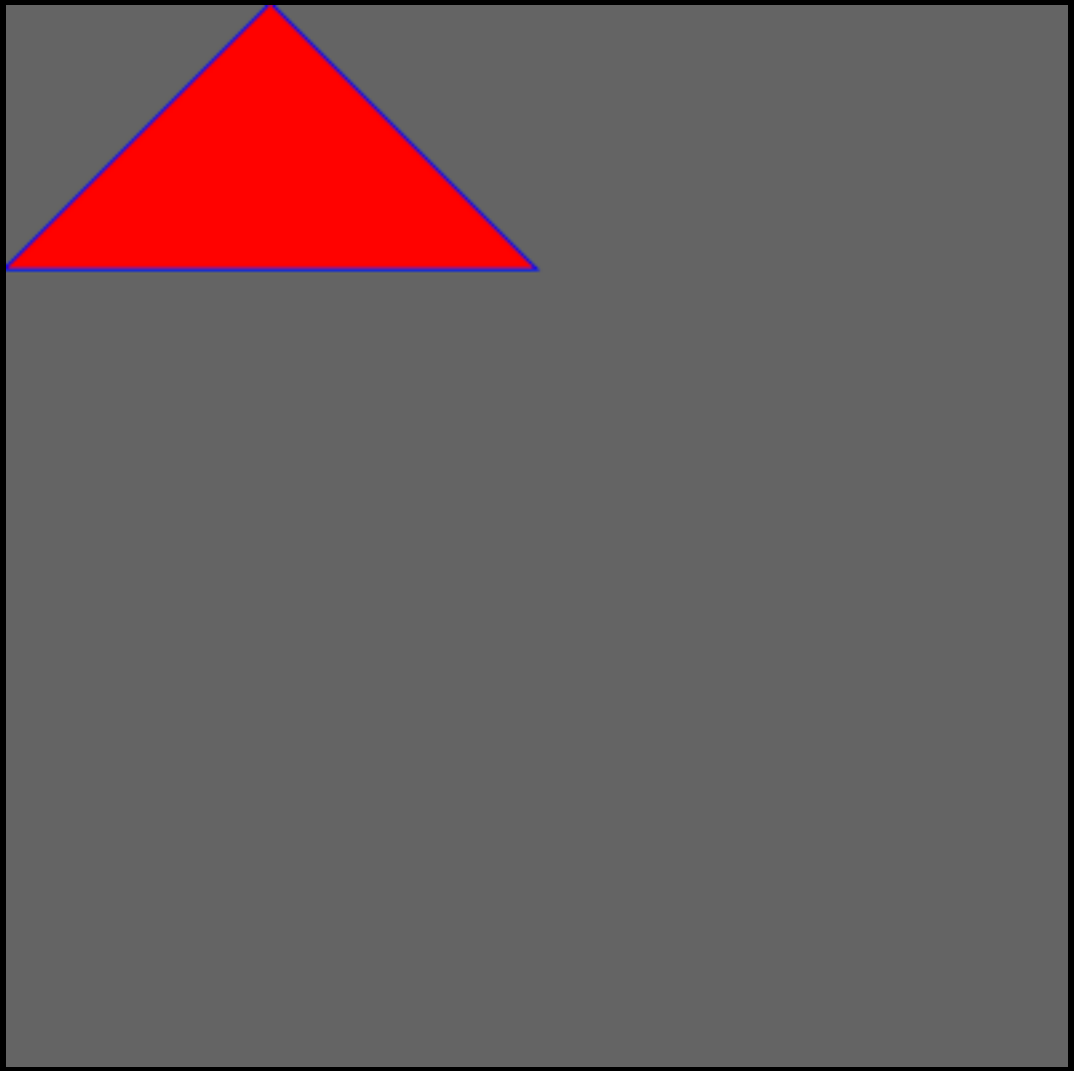


```
// internal method to have renderer draw a rectangle
p5.prototype._renderRect = function() {
  if (this._renderer._doStroke || this._renderer._doFill) {
    // duplicate width for height in case only 3 arguments is provided
    if (arguments.length === 3) {
      arguments[3] = arguments[2];
    }
    const vals = canvas.modeAdjust(
      arguments[0],
      arguments[1],
      arguments[2],
      arguments[3],
      this._renderer._rectMode
    );

    const args = [vals.x, vals.y, vals.w, vals.h];
    // append the additional arguments (either cornder radii, or
    // segment details) to the argument list
    for (let i = 4; i < arguments.length; i++) {
      args[i] = arguments[i];
    }
    this._renderer.rect(args);

    //accessible outputs
    if (this._accessibleOutputs.grid || this._accessibleOutputs.text) {
      this._accsOutput('rectangle', [vals.x, vals.y, vals.w, vals.h]);
    }
  }

  return this;
};
```

https://github.com/processing/p5.js/blob/main/src/core/shape/2d_primitives.js

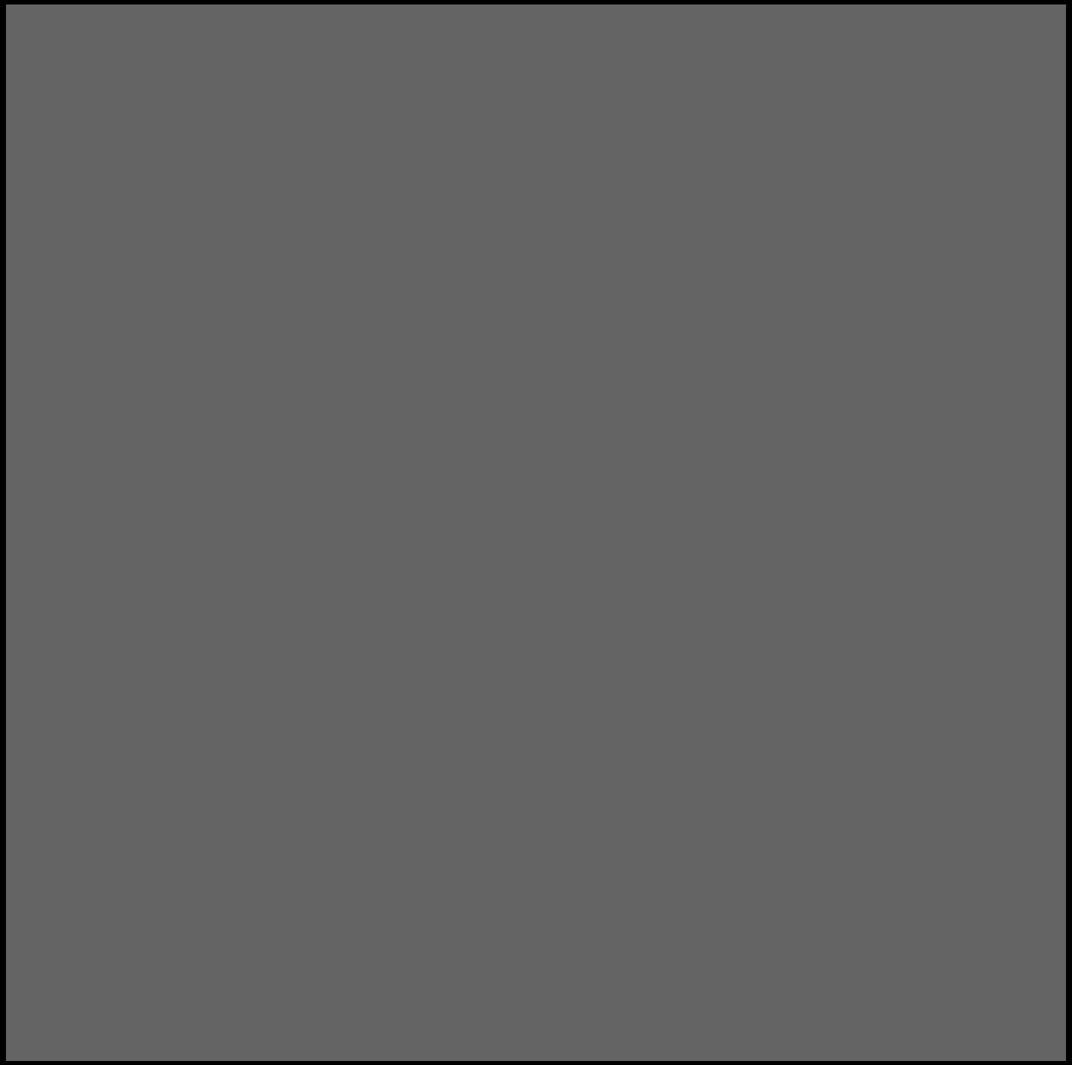
Function

```
function draw() {
  background(100);

  fill(255, 0, 0);
  stroke(0, 0, 255);
  beginShape();
  vertex(100, 0);
  vertex(0, 100);
  vertex(200, 100);
  endShape(CLOSE);
}
```



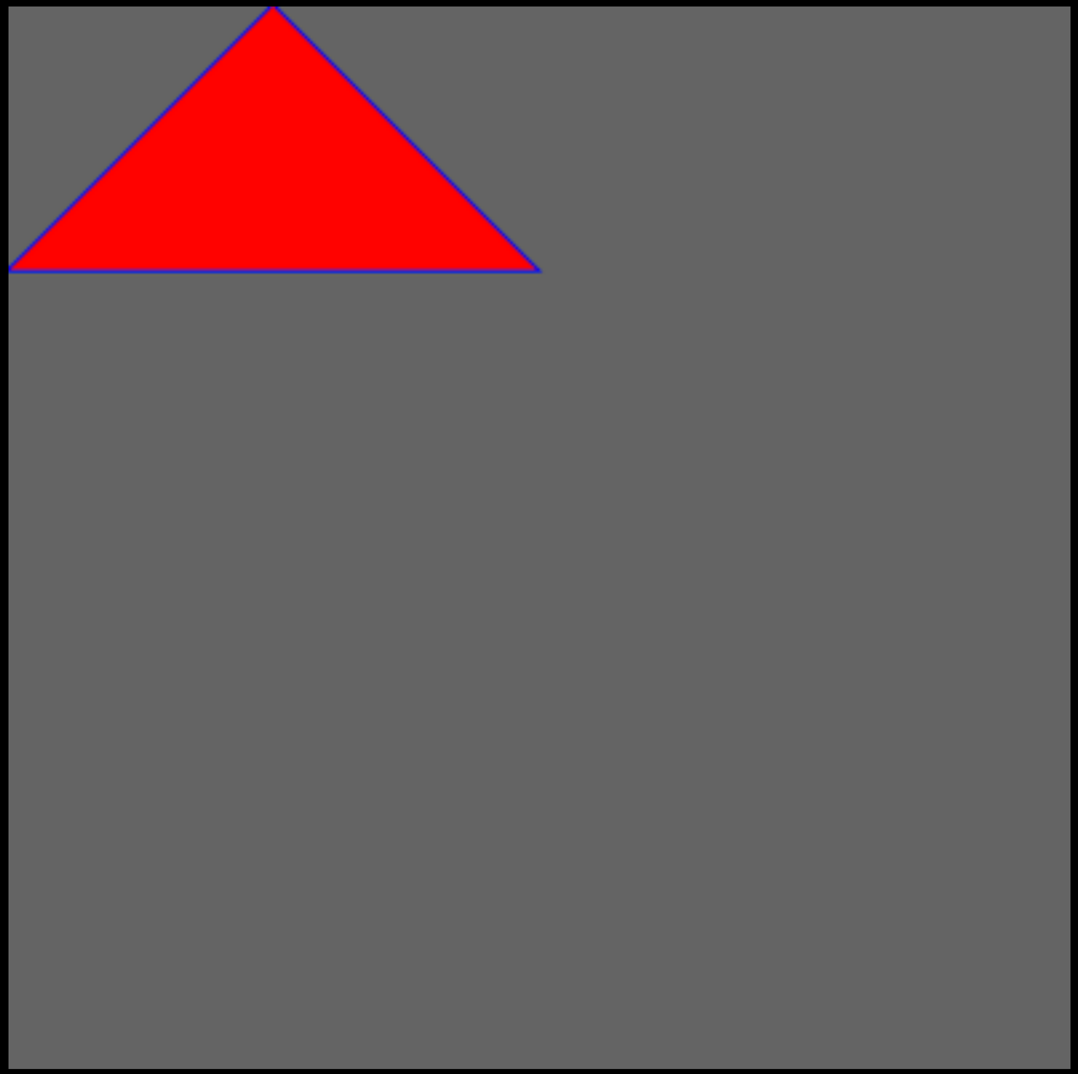
Function

```
function draw() {
  background(100);
}

function drawTri() {
  fill(255, 0, 0);
  stroke(0, 0, 255);
  beginShape();
  vertex(100, 0);
  vertex(0, 100);
  vertex(200, 100);
  endShape(CLOSE);
}
```



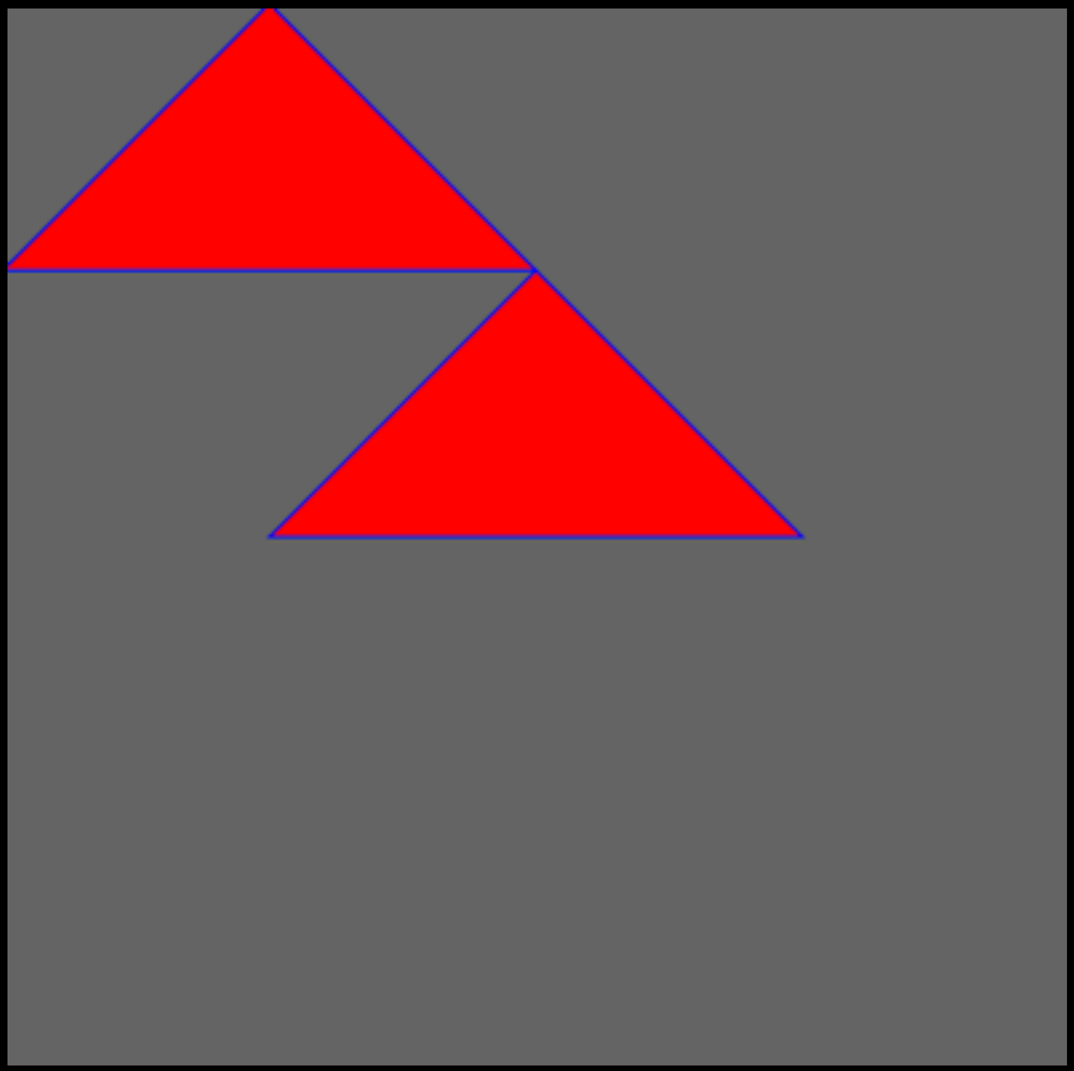
Function

```
function draw() {
  background(100);

  drawTri();
}

function drawTri() {
  fill(255, 0, 0);
  stroke(0, 0, 255);
  beginShape();
  vertex(100, 0);
  vertex(0, 100);
  vertex(200, 100);
  endShape(CLOSE);
}
```



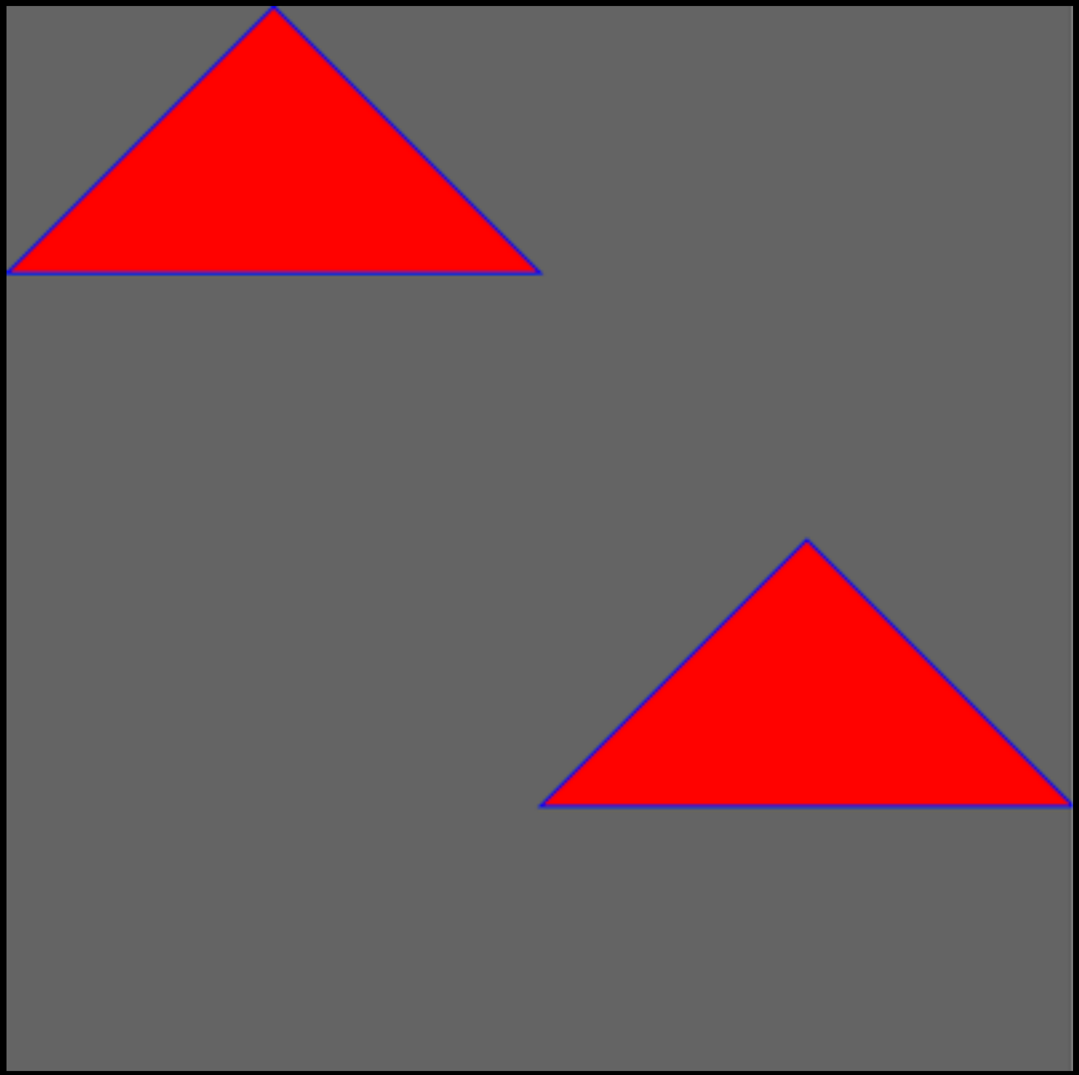
Function

```
function draw() {
  background(100);

  drawTri();

  push();
  translate(100, 100);
  drawTri();
  pop();
}

function drawTri() {
  fill(255, 0, 0);
  stroke(0, 0, 255);
  beginShape();
  vertex(100, 0);
  vertex(0, 100);
  vertex(200, 100);
  endShape(CLOSE);
}
```



Function

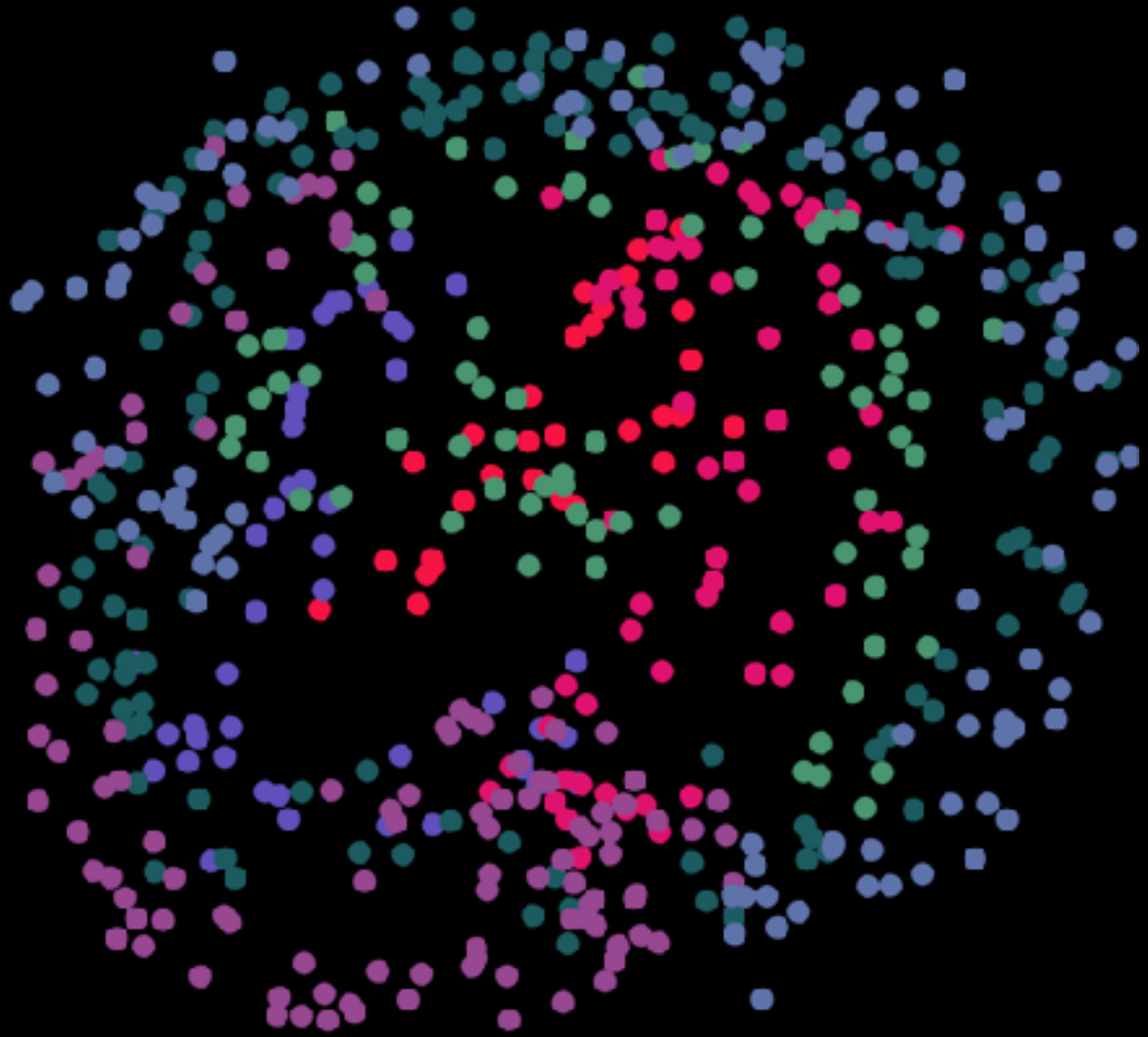
```
function draw() {
  background(100);

  drawTri(0, 0);

  drawTri(200, 200);
}

function drawTri(x, y) {
  push();
  translate(x, y);
  fill(255, 0, 0);
  stroke(0, 0, 255);
  beginShape();
  vertex(100, 0);
  vertex(0, 100);
  vertex(200, 100);
  endShape(CLOSE);
  pop();
}
```



Class & Objects

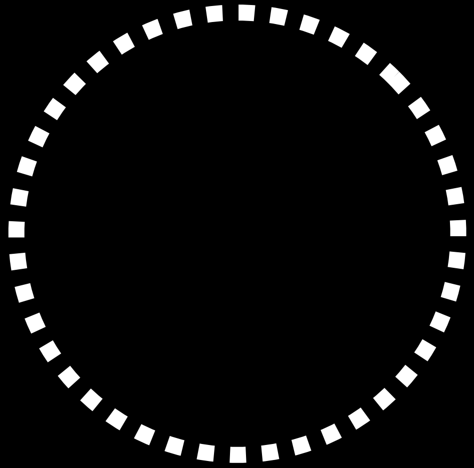
배열 array

https://editor.p5js.org/youngsangcho/sketches/yblJqaaM_



```
let xList = [];
let yList = [];
let colorList = [];

...

function mouseDragged() {
  xList.push(mouseX);
  yList.push(mouseY);
  colorList.push(randomColor);
}
```

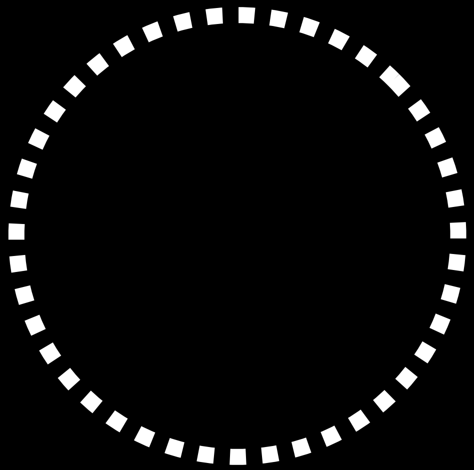
배열 array



x : Number
y : Number
c : Color

```
let xList = [];
let yList = [];
let colorList = [];

...

function mouseDragged() {
  xList.push(mouseX);
  yList.push(mouseY);
  colorList.push(randomColor);
}
```

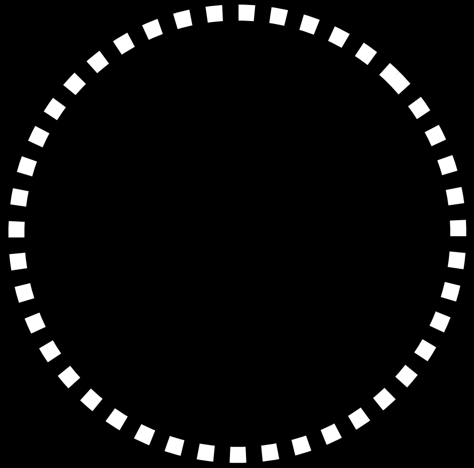
Class



x : Number
y : Number
c : Color

```
class Bubble {

}
```

Class

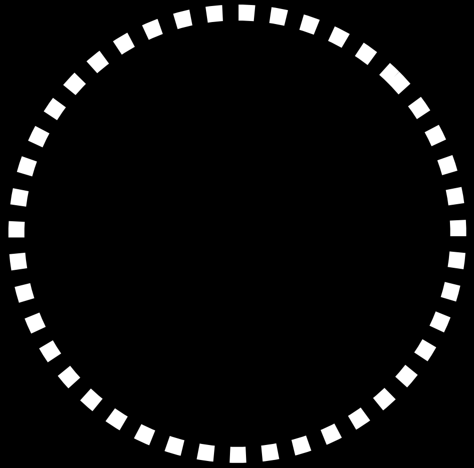


x : Number
y : Number
c : Color

```
class Bubble {

}
```

붕어빵 기계 틀

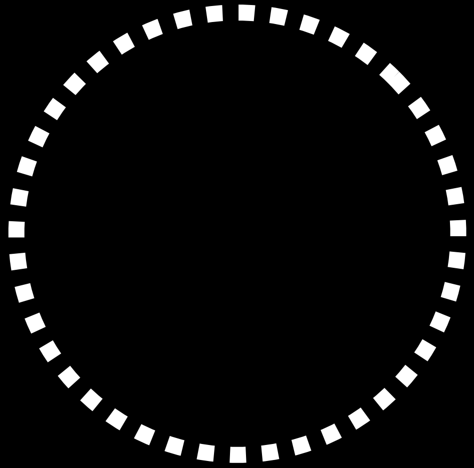
Class



x : Number
y : Number
c : Color

```
class Bubble {
  constructor() {

  }
}
```

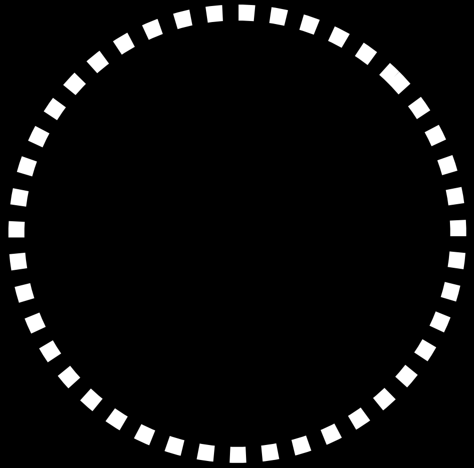
Class



x : Number
y : Number
c : Color

```
class Bubble {
  constructor() {
    this.x = random(width);
    this.y = random(height);
    this.c = color(random(255));
  }
}
```

Class



x : Number
y : Number
c : Color

```
class Bubble {
  constructor() {
    this.x = random(width);
    this.y = random(height);
    this.c = color(random(255));
  }
}
```

기계 안
실제 붕어빵 주형

Class

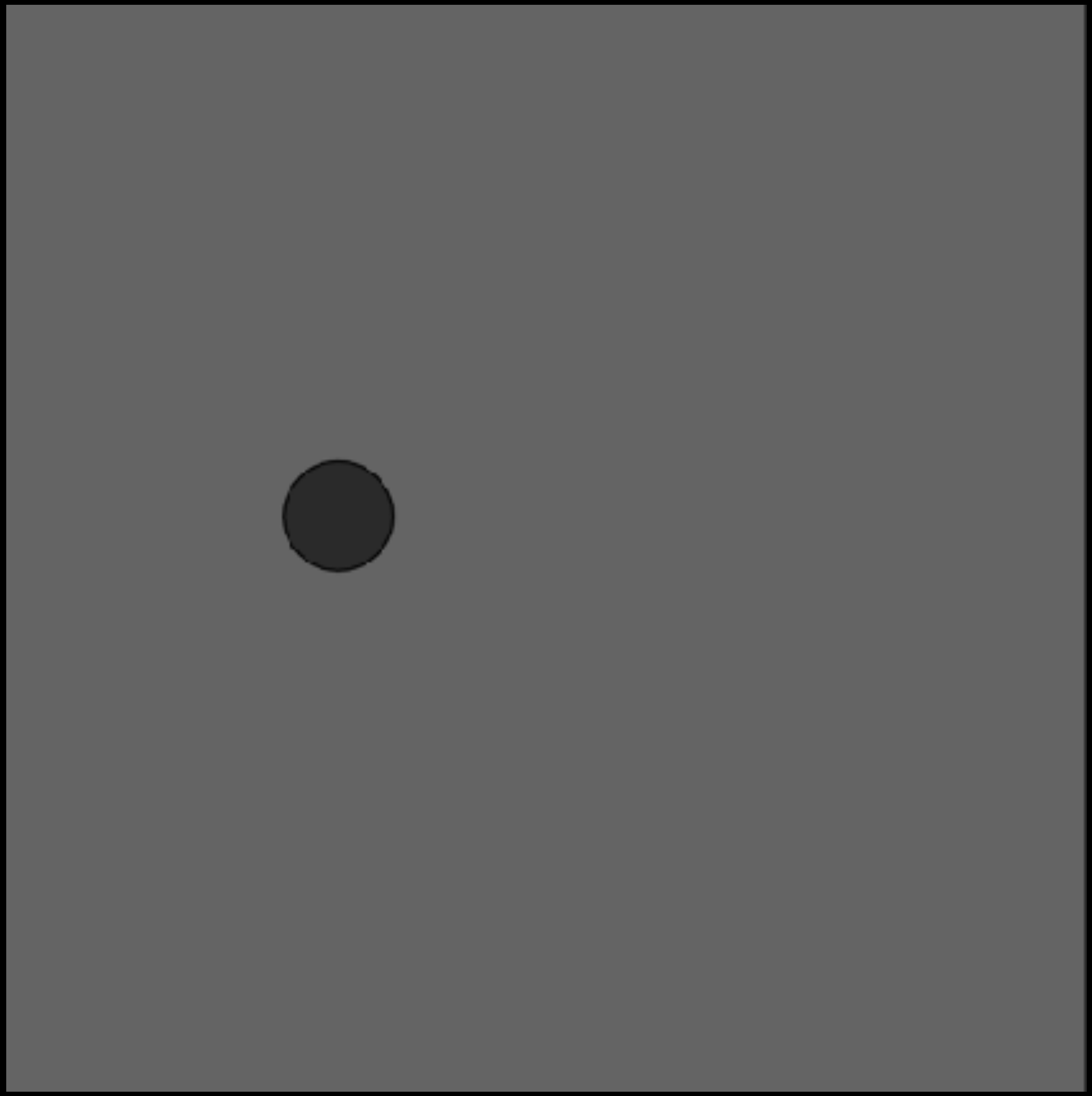
```
function setup() {
  createCanvas(500, 500);
}

function draw() {
}

class Bubble {
  constructor() {
    this.x = random(width);
    this.y = random(height);
    this.c = color(random(255));
  }
}
```

Object

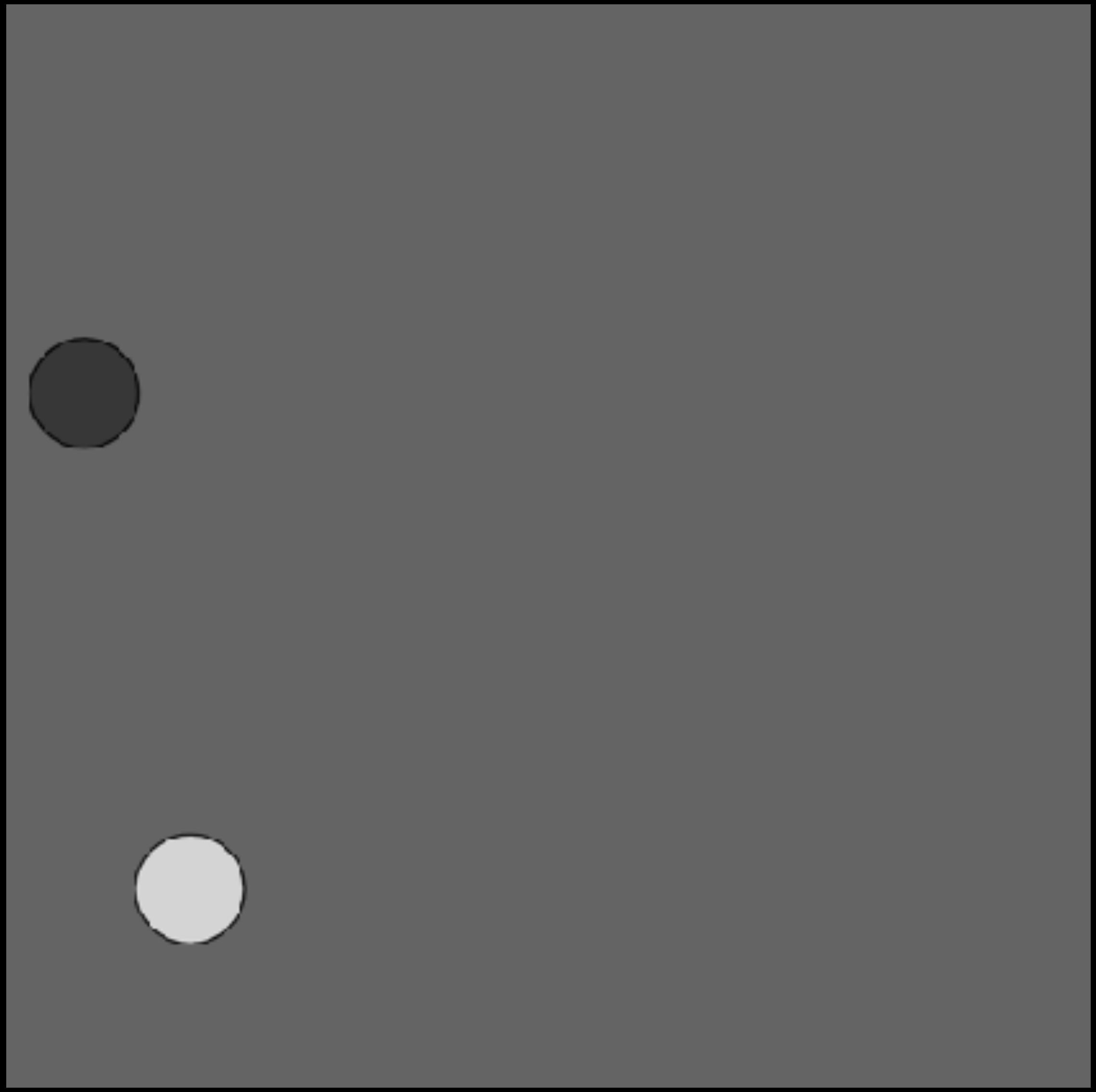
Class

```
let myBubble;

function setup() {
  createCanvas(500, 500);
  myBubble = new Bubble();
}

function draw() {
}

class Bubble {
  constructor() {
    this.x = random(width);
    this.y = random(height);
    this.c = color(random(255));
  }
}
```

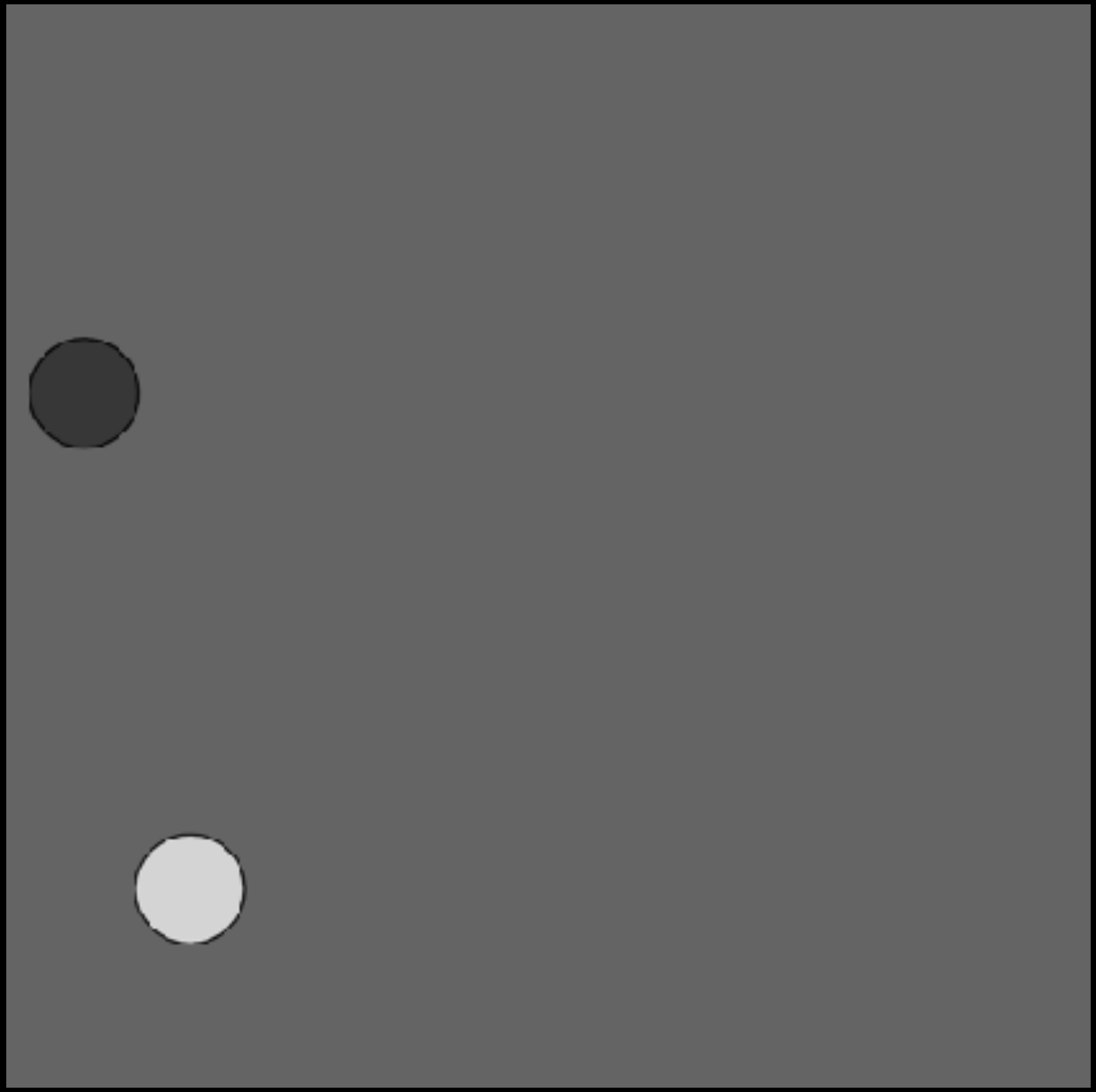
```
let myBubble;

function setup() {
  createCanvas(500, 500);
  myBubble = new Bubble();
}

function draw() {
}

class Bubble {
  constructor() {
    this.x = random(width);
    this.y = random(height);
    this.c = color(random(255));
  }

  render() {
    fill(this.c);
    ellipse(this.x, this.y, 50, 50);
  }
}
```



```
let myBubble;

function setup() {
  createCanvas(500, 500);
  myBubble = new Bubble();
}

function draw() {
  background(100);
  myBubble.render();
}

class Bubble {
  constructor() {
    this.x = random(width);
    this.y = random(height);
    this.c = color(random(255));
  }

  render() {
    fill(this.c);
    ellipse(this.x, this.y, 50, 50);
  }
}
```

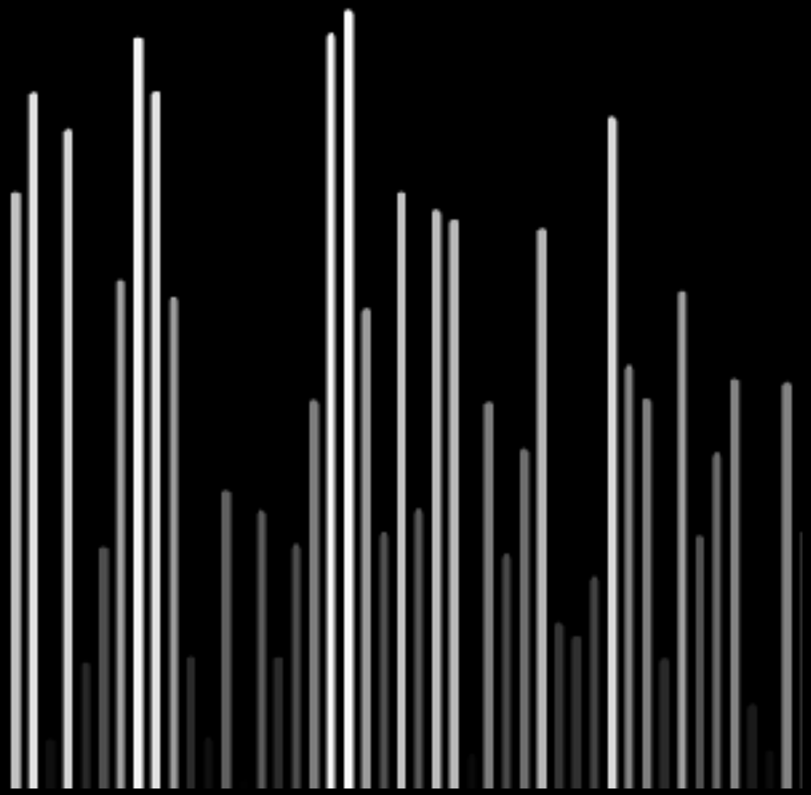


```
let myBubble1;
let myBubble2;

function setup() {
  createCanvas(500, 500);
  myBubble1 = new Bubble();
  myBubble2 = new Bubble();
}

function draw() {
  background(0);
  myBubble1.render();
  myBubble2.render();
}

class Bubble {
  constructor() {
    this.x = random(width);
    this.y = random(height);
    this.c = color(random(255));
  }

  render() {
    fill(this.c);
    ellipse(this.x, this.y, 50, 50);
  }
}
```



```
let myBubble1;
let myBubble2;

function setup() {
  createCanvas(500, 500);
  myBubble1 = new Bubble(100, 100);
  myBubble2 = new Bubble(300, 100);
}

function draw() {
  background(0);
  myBubble1.render();
  myBubble2.render();
}

class Bubble {
  constructor(x, y) {
    this.x = x;
    this.y = y;
    this.c = color(random(255));
  }

  render() {
    fill(this.c);
    ellipse(this.x, this.y, 50, 50);
  }
```

https://editor.p5js.org/youngsangcho/sketches/dR1ia4oWk



```
class Bar {
  constructor () {
    this.value = random(100);
    this.color = color(random(255));
  }
}
```

```
let bar;

function setup() {
  createCanvas(500, 500);
  bar = new Bar();
}
```

```
class Bar {
  constructor () {
    this.value = random(100);
    this.color = color(random(255));
  }
}
```

```
let bars = [];

function setup() {
  createCanvas(500, 500);
  for (let i = 0; i < 100; i++) {
    bars.push(new Bar());
  }
}
```

x 100

```
class Bar {
  constructor () {
    this.value = random(100);
    this.color = color(random(255));
  }
}
```

```
let bars = [];

function setup() {
  createCanvas(500, 500);
  for (let i = 0; i < 100; i++) {
    bars[i] = new Bar();
  }
}

function draw() {
  background(0);
  for (let i = 0; i < 100; i++) {
    bars[i].display();
  }
}

class Bar {
  constructor () {
    this.value = random(100);
    this.color = color(random(255));
  }

  display () {
    //...
  }
}
```

Sample Code

https://editor.p5js.org/youngsangcho/sketches/x5HGkmvh3

https://editor.p5js.org/youngsangcho/sketches/sf6O1aCZD

과제 1: 개별과제

과제 2: Data Visualization Sketch

일러스트레이터 or 포토샵으로 아이디어 스케치하기

비쥬얼은 러프해도 OK
다른 데이터와의 조합, 인터랙션, 정렬 등 다양한 아이디어 더해보기.

과제

매주 월요일 밤 10시

기한 맞춰 제출

평가 항목
+ 과제별 요구사항
+ 아이디어, 디자인
+ 노력, 시간

과제

남의 코드 베끼지 말기. 가능한 직접 쓰기.

다른 사람/인터넷의 코드를 참조하는 경우,
+ 블로그와 코드 안에 출처 밝히고,
+ 이해해서 내 것으로 만든 경우에만 인정. (모르면 질문)

과제 검사 시, 질문할 수도.