03 Arrays

```
[value1,
value2,
value3]
```

An array is a list of **data**.

Each piece of data in an array is stored inside [square brackets], and is identified by an **index number** representing its position in the array.

You can access the values in an array by their position in the list.

You can also add items to the array, or remove items, or modify items that are already in there.

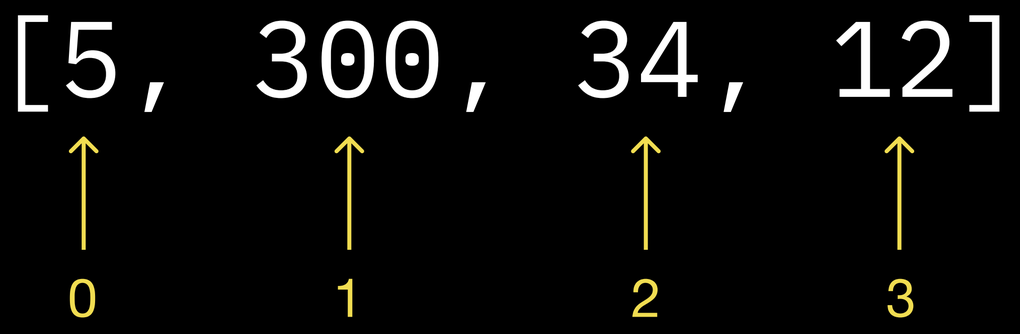
If a single variable is like a box for storing something, then an array is like a shelf to keep boxes on.

var = 📦

[📦 , 📦 , 📦 , 📦]

[5, 300, 34, 12]

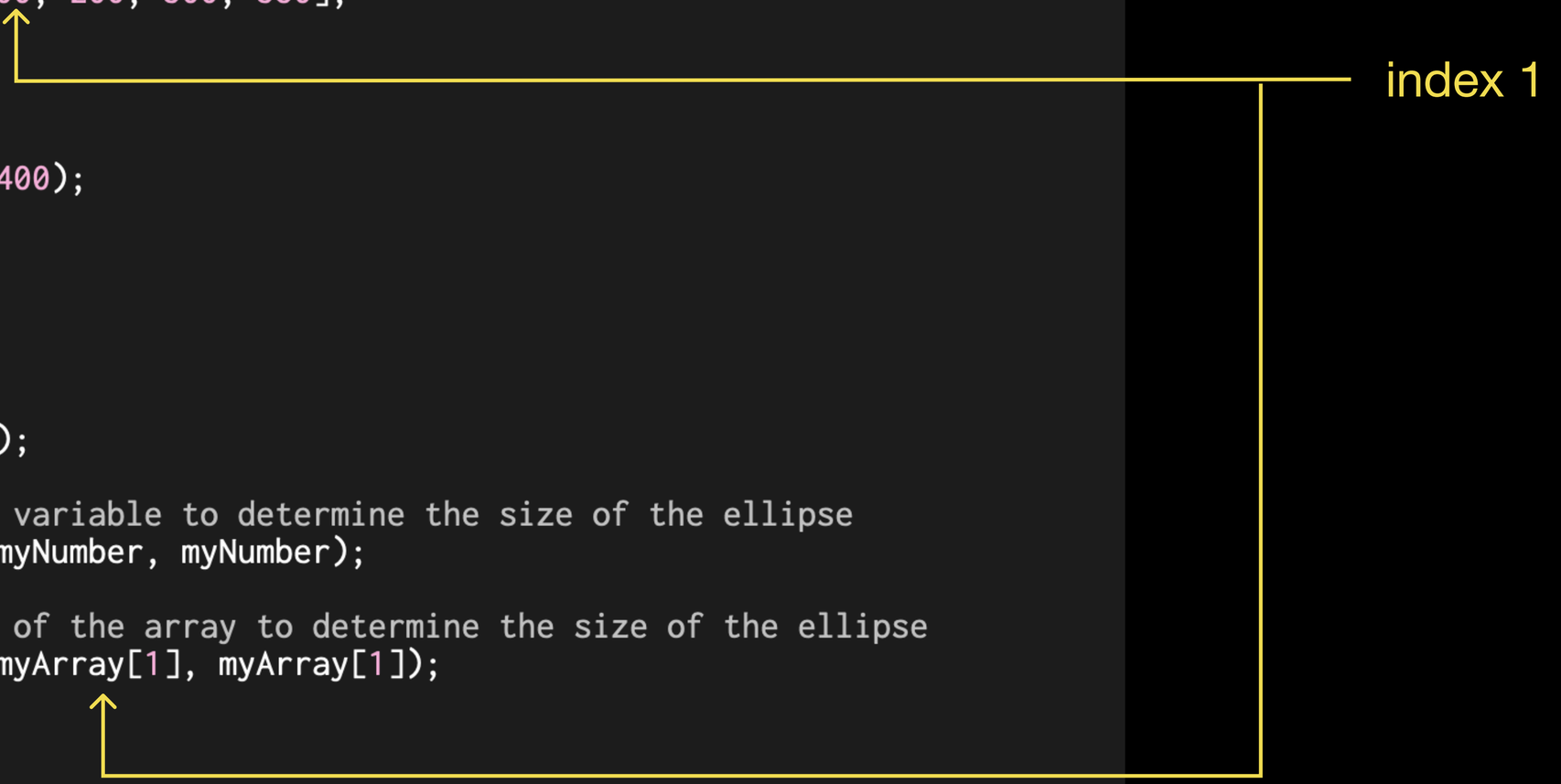
[“box_A”, “box_B”, “box_C”]



Array elements can be accessed using their index number.

In programming we start counting from 0, so the first element in an array is index 0, the second one is index 1.. and so on.

```
let myArray = [50, 100, 200, 300, 350];

var myNumber = 40

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

function draw() {
  background(220);

  fill(100, 150, 255);

  //using a standard variable to determine the size of the ellipse
  ellipse(200, 200, myNumber, myNumber);

  //using an element of the array to determine the size of the ellipse
  ellipse(200, 200, myArray[1], myArray[1]);

}
```



```
var words = ["sunny", "cloudy", "rainy", "foggy"]

var index = 0;

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

function draw() {
  background(220);

  fill(255);
  textSize(32);
  text(words[index], 12, 200);

}

function mousePressed() {
  index = index +1;

  if (index == 4) {
    index = 0;
  }

}
```

dynamically accessing array elements

```
var colors = ["#FF5733", "#33FF57", "#3357FF", "#FF33A6", "#6B6769"];
// Colors in HEX format: red, green, blue, pink

var index = 0;

function setup() {
  createCanvas(400, 400);
  rectMode(CENTER);
}

function draw() {
  background(0);

  fill(colors[index]);
  rect(width/2, height/2, 100, 100);
}

function keyPressed() {
  index = index + 1;

  if (index == colors.length) {
    index = 0;
  }
}
```

using array properties to know more about its elements

Create a sketch using arrays to determine:

- The size of a series elements on the canvas
- The position of the elements on the canvas
- The colour of the elements on the canvas

04 Conditionals

if this then that

“Conditions in a programming language are instructions that direct the flow of a program.

This works a bit like rail traffic: a train travels on a rail until it hits a switch, then the train changes its direction of travel.

Condition change the course of a program.”

We use this types of ‘programming’ in our daily life too!

Can you think of any simple real life examples?

if I am hungry 😋,

I go to the kitchen

and make a 🥪.

if I am hungry 😋,
I go to the kitchen
and make a 🥪.
if I am **not** hungry,
I will go for a walk instead 

```
if (hungry == true) {
  goToKitchen()
  makeASandwich()
} else {
  goForAWalk()
}
```

```
if (hungry == true) {
   goToKitchen()
   makeASandwich()
} else {
   goForAWalk()
}
```

Relational operators

- equal to ==
- not equal !=
- greater than >
- less than <
- greater than or equal to >=
- less than or equal to <=
- logical And &&
- logical Or ||

Conditionals

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

  // You can change this to true or false to test both conditions
  let hungry = false;

  if (hungry == true) {
    goToKitchen();
    makeASandwich();
  } else {
    goForAWalk();
  }
}

function goToKitchen() {
  console.log("Going to the kitchen...");
}

function makeASandwich() {
  console.log("Making a sandwich...");
}

function goForAWalk() {
  console.log("Going for a walk...");
}

function draw() {
  background(0);
}
```

Conditionals

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

function draw() {
  background(0);
  stroke(255);
  noFill();

  // Check if mouse is over half of the canvas

  if (mouseX > 200) {

    fill(255, 0, 0);

  }

  rect(150, 150, 100, 100);
  fill(255);

  textSize(18);
  text("mouseX: ", 10, 30);
  text(mouseX, 95, 30);
}
```

05 Loops

make a 
make a 
make a 
make a 
make a 
make a 

Loops in programming are used to repeat a block of code multiple times.

They are essential for performing tasks that need to be done repeatedly, such as iterating through a list of items, drawing patterns, or performing calculations.

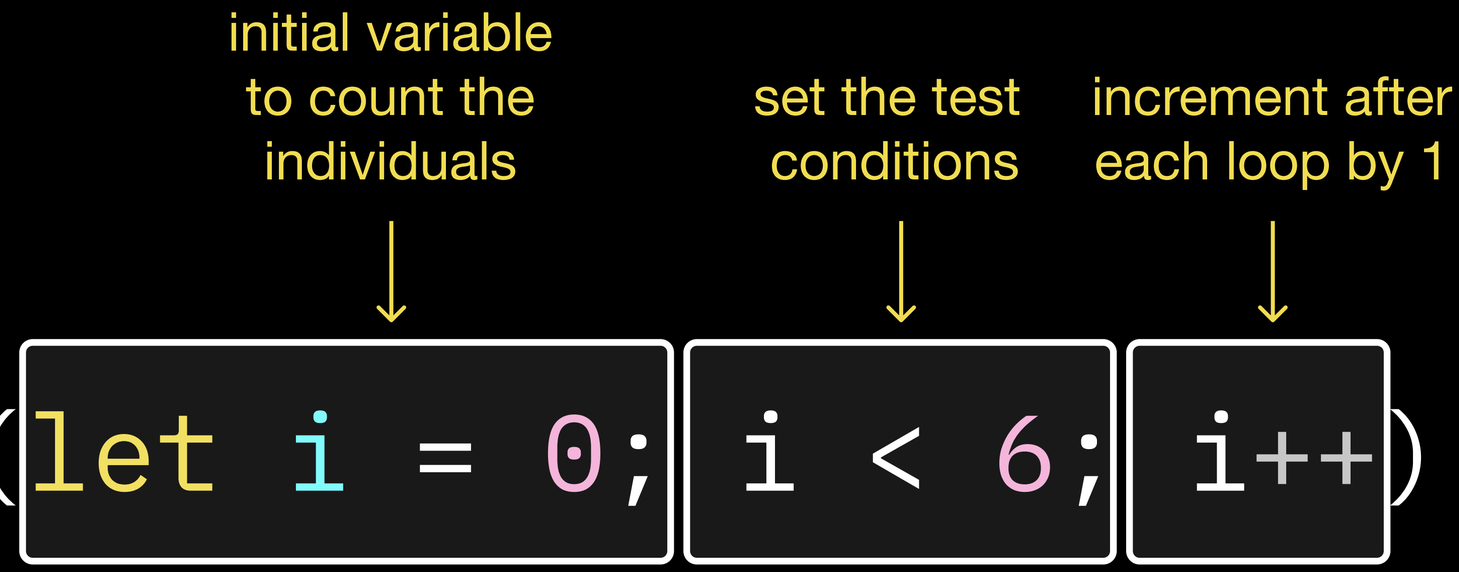
So far we have learnt that the draw( ) function iterates as a loop as well, but what happens if you want to have other loops inside that? We need to use a **For Loop or a While Loop.**

there are 6 individuals,

make a  for each of them

```
console.log(“make a sandwich”)
console.log(“make a sandwich”)
console.log(“make a sandwich”)
console.log(“make a sandwich”)
console.log(“make a sandwich”)
console.log(“make a sandwich”)
```

```
for (let i = 0; i < 6; i++) {
  console.log(“make a sandwich”);
}
```




```
for (let i = 0; i < 6; i++) {
  console.log("make a sandwich");
}
```

Loops

```
function setup() {
  createCanvas(300, 300);
}

function draw() {
  background(0);
  stroke(255);


  //Not using for loops

  // line(30, 0, 30, height);
  // line(60, 0, 60, height);
  // line(90, 0, 90, height);
  // line(120, 0, 120, height);
  // line(150, 0, 150, height);
  // line(180, 0, 180, height);
  // line(210, 0, 210, height);
  // line(240, 0, 240, height);
  // line(270, 0, 270, height);


  //Using for loops

  for (let lineX = 30; lineX <= 270; lineX += 30) {
    line(lineX, 0, lineX, height);
  }

}
```

```
function setup() {
  createCanvas(400, 400);
  background(220);
}

function draw() {

  for ( var x = 0; x <= width; x = x + 40) {

    noStroke();
    fill(0, 0, 255);
    ellipse(x, 200, 10, 10);

  }


}
```

Pair up with one of your peers, and create a sketch that follows these instructions:

- Draw a series of elements across the canvas
- Create an if statement condition to check something about your elements
- Apply a different behaviour if that condition is true.
- Invent and add your own rules.
- Present your work to others.

[For example, if the elements position is past the middle of the canvas, fill it with different colour]

More on: https://editor.p5js.org/troglodisme/sketches