This is a review of what we covered in this tutorial on arrays.

We often want to store lists of values when we're creating programs, and in JavaScript, we can do that using a type of value called an ***array***.

To create an array, we declare a variable like we always do, but then we surround our list of values with square brackets and separate each value with a comma:

```
var xPositions = [33, 72, 64];
```

We can store any sort of JavaScript value in an array - not just numbers. Here's an example where we store a list of strings:

```
var myFriends = ['Winston', 'OhNoesGuy', 'John', 'Sophia'];
```

We often want to display the length of an array, or do something based on the length of the array. Thankfully, every array has a `length` property that will tell us the current length of the array:

```
text(myFriends.length, 200, 200); // Displays "4"
```

When we want to access a particular value in an array, we access it by referencing its "index" in the array, which represents its position. The first index in an array is "0", so if we want to access the first element in an array, we specify the name of the array variable, then square brackets, and 0:

```
text(myFriends[0], 200, 0); // Displays "Winston"
```

The second element is at index "1", the third is at index "2", and the fourth is at index "3":<

```
text(myFriends[1], 200, 100); // Displays "OhNoesGuy"
text(myFriends[2], 200, 200); // Displays "John"
text(myFriends[3], 200, 300); // Displays "Sophia"
```

The zero-based indexing is one of the most confusing aspects of arrays for new programmers, so keep that in mind if you're just getting started with arrays. You'll get used to it eventually!

We often want to take some action for every element in an array, like how we used the text() command to display the names above. Instead of writing that code over and over, it's better to use a for loop to iterate through each of the elements in the array, and do something to each element inside the loop. We have to start from index 0, go until we reach the end of the array, and add 1 to the index each time. Here's how we'd do that:

```
for (var i = 0; i < myFriends.length; i++) {
  text(myFriends[i], 200, i*100);
}
```

Notice how we put i inside the square brackets, because it represents the current index each time the loop is run.

Arrays can be changed in many ways. To start off with, we can change a value in them:

```
myFriends[1] = "TheErrorBuddy";
```

We can also add entirely new values to them, using the push() method, passing in the new value:

```
myFriends.push("Hopper");
```

After running that line of code, our array length property will change to reflect the new length, and the final index in the array will be 4 instead of 3.

If you want a full list of what you can do with arrays in JavaScript, check out this reference. But don't worry, everything in this review will get you very far!