MORE ARRAY METHODS

IN THIS LESSON

Learning built-in methods

concat()

sort()

reverse()

slice()

splice()

LEARNING BUILT-IN METHODS

You don't need to memorize them

Learn that they exist, what they do,
understand how to use them

If you need them later, look them up - you will end up memorizing the ones you use often, don't worry about memorizing them all perfectly from the beginning

CONCAT()

Use **concat()** to combine two arrays into one

Does not mutate original arrays

Return value is a new array containing all items from both arrays

Syntax: firstArray**.concat(**secondArray**)**

```
const primaryColorsArr = ['red', 'blue', 'yellow']
const secondaryColorsArr = ['purple', 'green', 'orange'];

const colorsArr = primaryColorsArr.concat(secondaryColorsArr);
```

```
colorsArr ['red', 'blue', 'yellow', 'purple', 'green', 'orange']
```

SORT()

Use **sort()** to alphabetically sort array of strings

Mutates the original array

```
const colorsArr = ['red', 'blue', 'yellow', 'purple', 'green', 'orange']
```

```
colorsArr.sort();
```

```
colorsArr ['blue', 'green', 'orange', 'purple', 'red', 'yellow']
```

REVERSE()

Use **reverse()** to alphabetically sort array of strings

Mutates the original array

```
const colorsArr = ['blue', 'green', 'orange', 'purple', 'red', 'yellow']
```

```
colorsArr.reverse();
```

```
colorsArr    ['yellow', 'red', 'purple', 'orange', 'green', 'blue']
```

SLICE()

Use **slice()** to copy part of an array and place it into a new array

Does not mutate the original array

Return value is a new array with copies of the "sliced" out items

Syntax: array**.slice(**beginIndex, endIndex**)**

```
const testArr = ['a', 'b', 'c', 'd', 'e', 'f', 'g'];
```

```
let slicedArr = testArr.slice(2, 5);
```

slicedArr	['c', 'd', 'e']

```
let slicedArr = testArr.slice(2);
```

slicedArr	['c', 'd', 'e', 'f', 'g']

SPLICE()

Use **splice()** to insert, add to, or remove items from an array at any point, not only the beginning or the end

Mutates the original array

SPLICE() TO INSERT

Syntax: array**.splice**(atIndex, 0, item)

```
const testArr = ['a', 'b', 'c', 'd']
```

```
testArr.splice(2, 0, 'x');
```

testArr ['a','b', **'x'**, 'c', 'd']

Add multiple items:

array**.splice(**atIndex, 0, item1, item2, item3, ...**)**

```
testArr.splice(2, 0, 'x', 'y', 'z');
```

testArr ['a','b', **'x', 'y', 'z'**, 'c', 'd']

SPLICE() TO REMOVE

Syntax: array**.splice(**atIndex, numItemsToRemove)

splice() returns the removed item(s)

```
const testArr = ['a', 'b', 'c', 'd', 'e', 'f']
```

```
const removed = testArr.splice(1, 3);
```

testArr	['a', 'e', 'f']
removed	['b', 'c', 'd']

SPLICE() TO REPLACE

Syntax: array**.splice**(atIndex, numItemsToReplace, item(s))

splice() returns the replaced item(s)

```
const testArr = ['a', 'b', 'c', 'd', 'e', 'f']
```

```
const replaced = testArr.splice(1, 3, 'uno');
```

testArr	['a', 'uno', 'e', 'f']
replaced	['b', 'c', 'd']