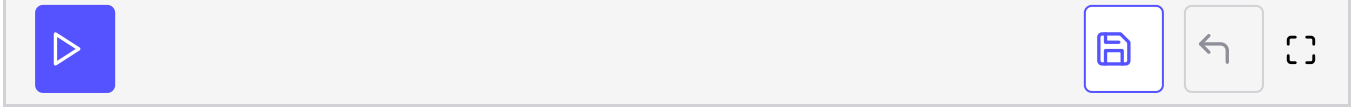
for-loops, continued

Learn how to repeat a set of code as many times as you like, without using functions. For-loops are a fundamental part of every programming language. We'll see how to write them and how to use them with arrays.

Refactoring our Code

For-loops are one of the few places with a single letter name is acceptable. Instead of `index`, it's common practice to use `i` or `j`. Let's refactor our code from the previous lesson.

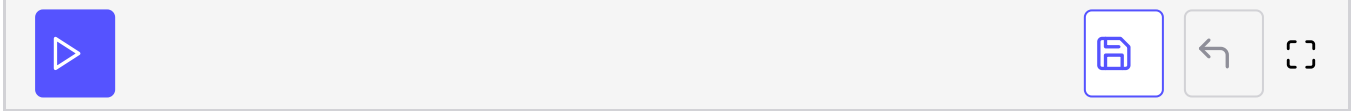
```
for(let i = 0; i < 10; i = i + 1) {
    console.log(i);
}
```



It's also common to see `i = i + 1` written another way: `i++`. This is a shorthand built into the language. It means "add one to this variable's value". So `i = i + 1` is completely equivalent to `i++`.

Similarly, `i--` means `i = i - 1`.

```
for(let i = 0; i < 10; i++) {
    console.log(i);
}
```



And this is the final version of our for-loop.

Looping Through an Array

A for-loop is a great way to work with every value in an array. If we want to work with every item, we want to start the loop at 0 and end it at the last item.

```
let arr = ['abc', 'def', 'ghi'];

for(let i = 0; i < arr.length; i++) {
    console.log(arr[i]);
}
```

Like if-statements, a single-line loop body can be written without brackets. The above and below blocks are equivalent.

```
let arr = ['abc', 'def', 'ghi'];

for(let i = 0; i < arr.length; i++)
    console.log(arr[i]);
```

Infinite Loops

When we write a loop, we should be as sure as we can that the condition in the middle will eventually be met. If not, we're stuck in an infinite loop. This can break a JavaScript execution environment.

To see this, run the block below. It'll never finish. We put `true` in as the condition, so it'll never be false. CAUTION: it may break the page and force you to refresh.

```
for(i = 0; true; i++) {
    console.log('This will print infinitely...');
}
```

That's it for loops! (no pun intended)

Quiz

Feel free to test your understanding.

1 Which one of these will print the numbers from 10 - 100, inclusive? In other words, which loop will print 10, 11, 12, ..., 99, 100? Multiple answers may be correct.

- [ ] A)
```
for(let i = 10; i < 100; i++) { console.log(i); }
```

- [ ] B)
```
for(let i = 10; i < 101; i++) { console.log(i); }
```

- [ ] C)
```
for(let i = 10; i <= 99; i++) { console.log(i); }
```

- [ ] D)
```
for(let i = 10; i <= 100; i++) { console.log(i); }
```

2 Given an array `arr`, which of these is a correct way to log all items in `arr`? Multiple answers may be correct.

- [ ] A)
```
for(let i = 0; i < arr.length; i++) { console.log(arr[i
]); }
```

- [ ] B)
```
for(let i = 0; i <= arr.length - 1; i++) { console.log(a
rr[i]); }
```

- [ ] C)
```
for(let i = 1; i <= arr.length; i++) { console.log(arr[
i + 1]); }
```

D)

```
for(let i = 1; i <= arr.length; i++) { console.log(arr[
i - 1]); }
```

3

Given an array `arr`, which of these is a correct way to log all items in `arr` BACKWARDS? Multiple answers may be correct.

A)

```
for(let i = 0; i < arr.length; i--) { console.log(arr[i
]); }
```

B)

```
for(let i = arr.length; i > 0; i--) { console.log(arr[i
]); }
```

C)

```
for(let i = arr.length - 1; i > 0; i--) { console.log(ar
r[i]); }
```

D)

```
for(let i = arr.length - 1; i >= 0; i--) { console.log(a
rr[i]); }
```

CHECK ANSWERS