Functional programming

Some basics

What is functional programming

Functional programming is a software development **paradigm** that shapes the way we write code. It gives us guidelines that try to stop us from falling into patterns that commonly lead to bugs.

What are programming paradigms?

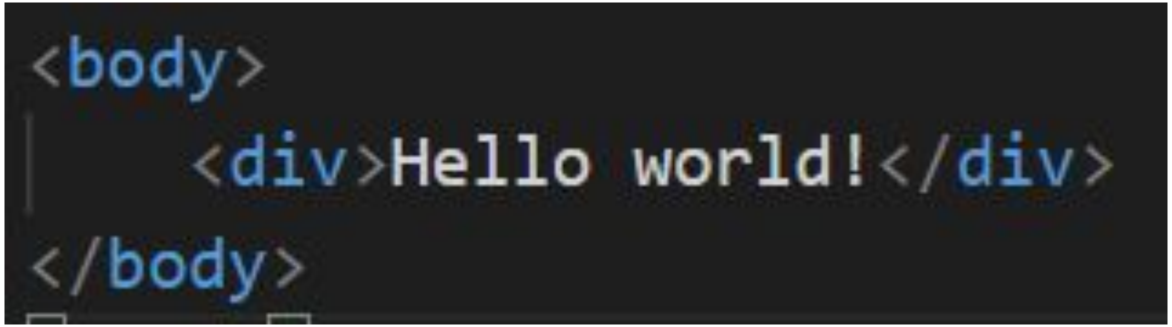
You’ve heard already about different programming paradigms. You’ve heard terms like **imperative programming** or **declarative programming.**

What are programming paradigms?

And you’ve already experienced them. HTML offers an example of a language that follows the **declarative programming** paradigm. In it, you state what you want to do without stating how.

What are programming paradigms?

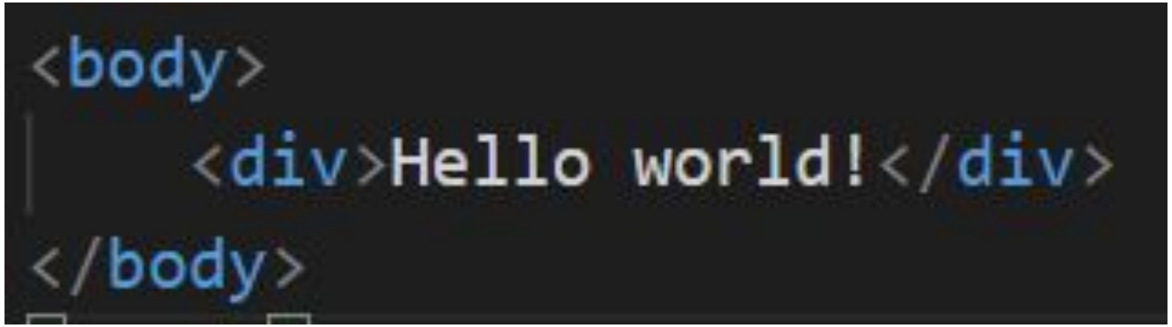
This is an example of **declarative programming**.



```
<body>
    <div>Hello world!</div>
</body>
```

What are programming paradigms?

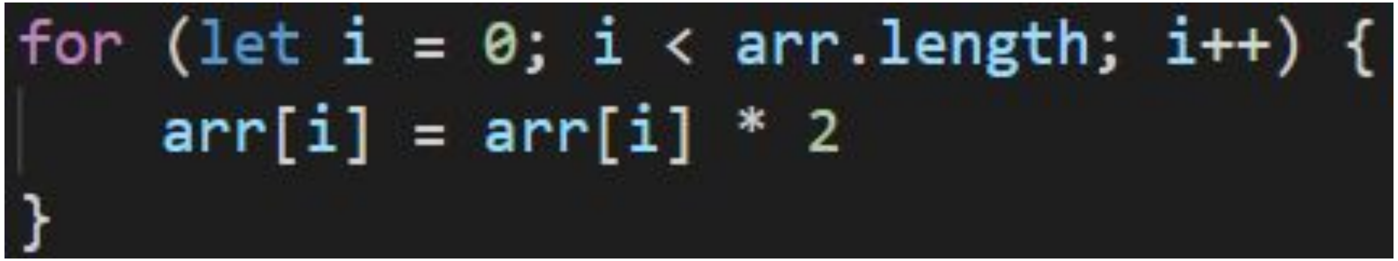
We tell the computer to render the div to the webpage, without telling it how to do it.



```
<body>
    <div>Hello world!</div>
</body>
```

What are programming paradigms?

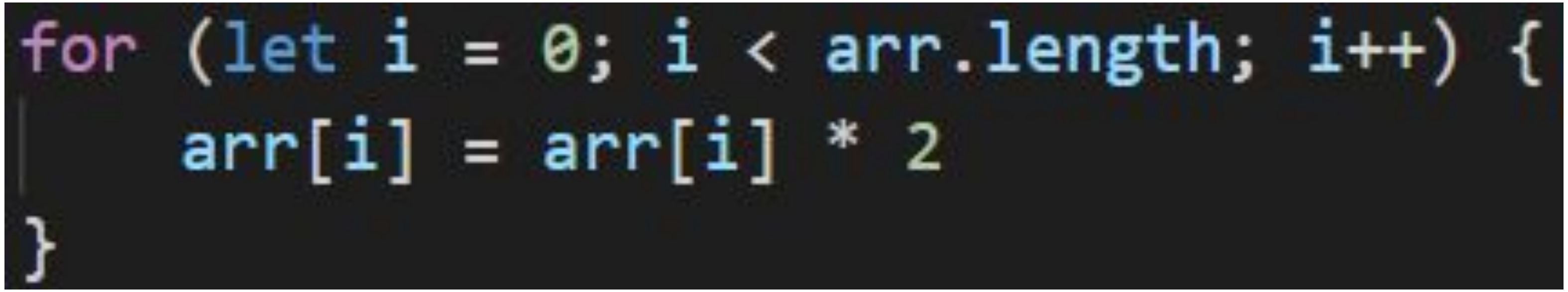
By contrast, this is an example of **imperative programming**.



```
for (let i = 0; i < arr.length; i++) {
    arr[i] = arr[i] * 2
}
```

What are programming paradigms?

Here, we're telling the computer each step of what to do.



```
for (let i = 0; i < arr.length; i++) {
    arr[i] = arr[i] * 2
}
```

Functional programming

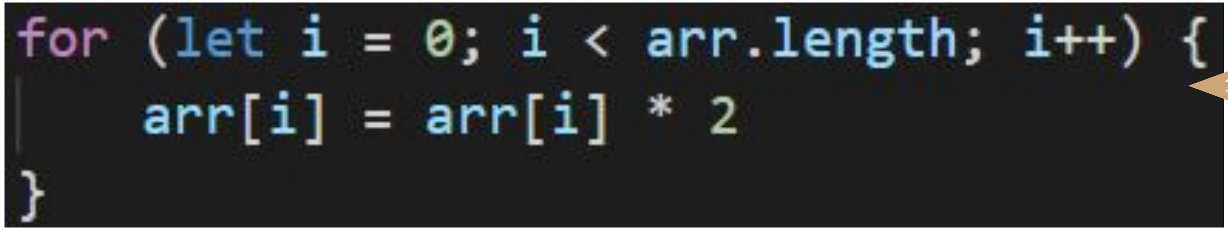
Javascript is a multi-paradigm language. You can use declarative or imperative approaches. **Functional programming** is a subset of declarative programming.

Functional programming

Because it's a sub-paradigm of declarative, it affects the way you write functional code. It means less code, because Javascript has a lot of the in-built functions you commonly need.

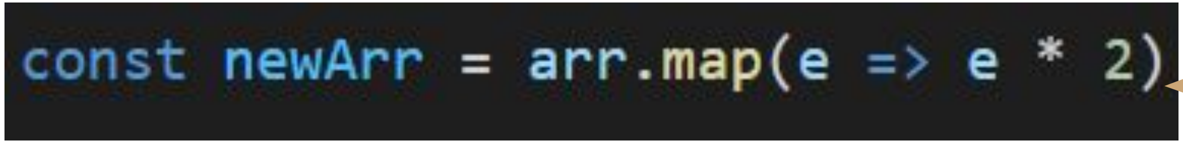
Functional programming

Here's a practical example:



```
for (let i = 0; i < arr.length; i++) {
    arr[i] = arr[i] * 2
}
```

imperative



```
const newArr = arr.map(e => e * 2)
```

declarative

A method tells the computer what to do.

Functional programming

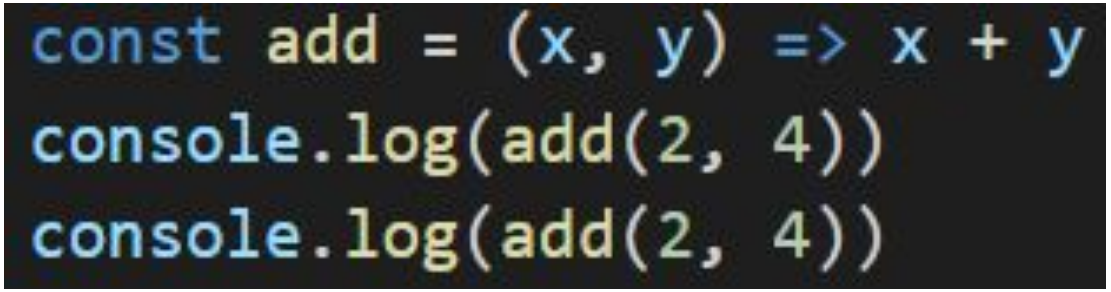
Both these pieces of code will double the numbers in our array. The map method, however, does something else for us too. It returns a new array. That’s also an important part of **functional programming.**

Functional programming

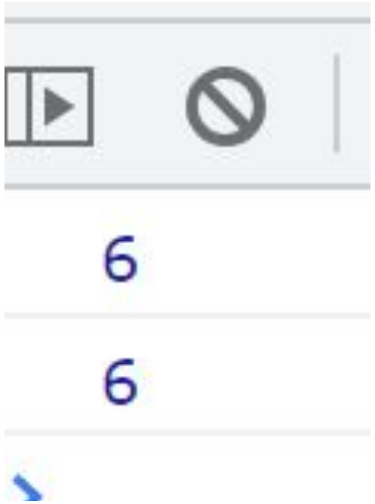
Functional programming **definition**: In functional programming, solutions are simple, isolated functions, without any side effects outside of the function scope.

What is a pure function?

In a pure (or isolated) function, the same input always returns the same output, and it has no side effects.



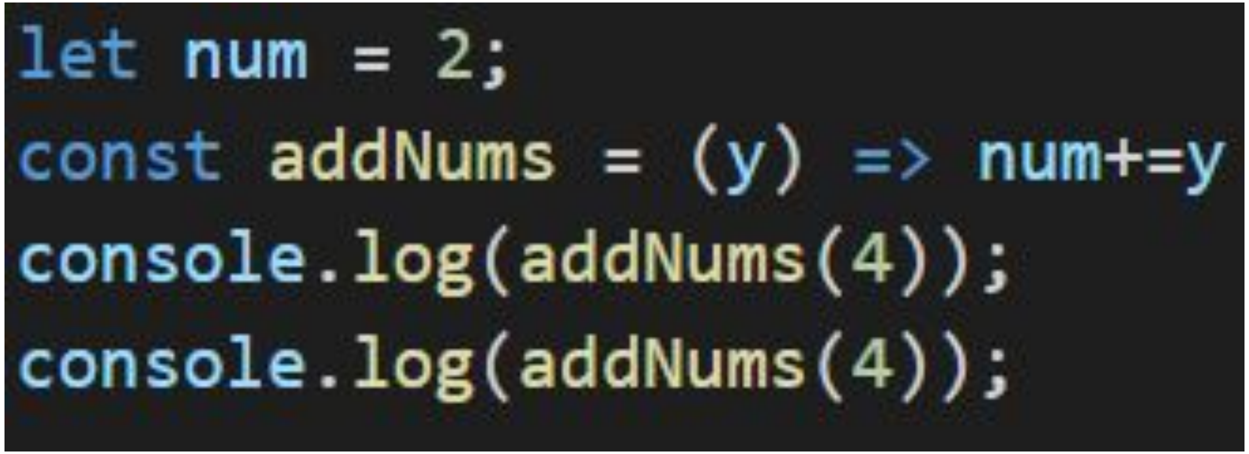
```
const add = (x, y) => x + y
console.log(add(2, 4))
console.log(add(2, 4))
```



```
6
6
```

What is a pure function?

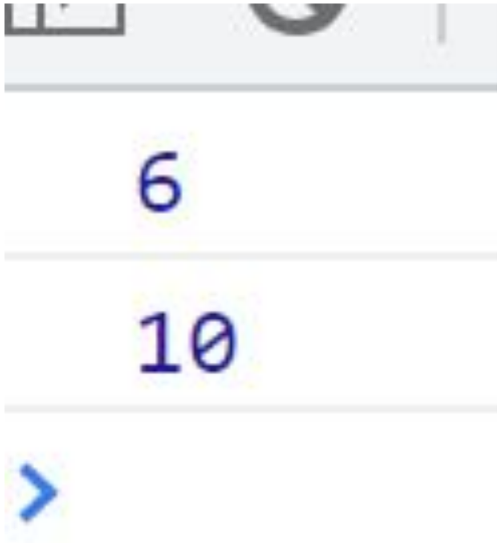
What will this code give us in our console?



```
let num = 2;
const addNums = (y) => num+=y
console.log(addNums(4));
console.log(addNums(4));
```

What is a pure function?

We will see this:

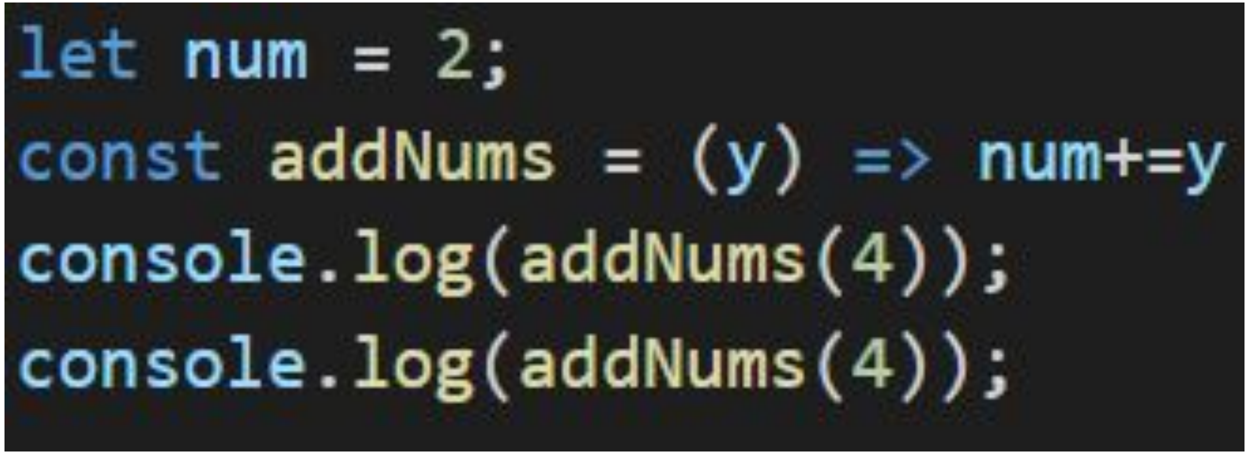


```
6
10
>
```

The first time we call the function we get one thing. We call it again and get something else.

What is a pure function?

This relies on shared state to do its job, incrementing a variable from outside.



```
let num = 2;
const addNums = (y) => num+=y
console.log(addNums(4));
console.log(addNums(4));
```

What is a pure function?

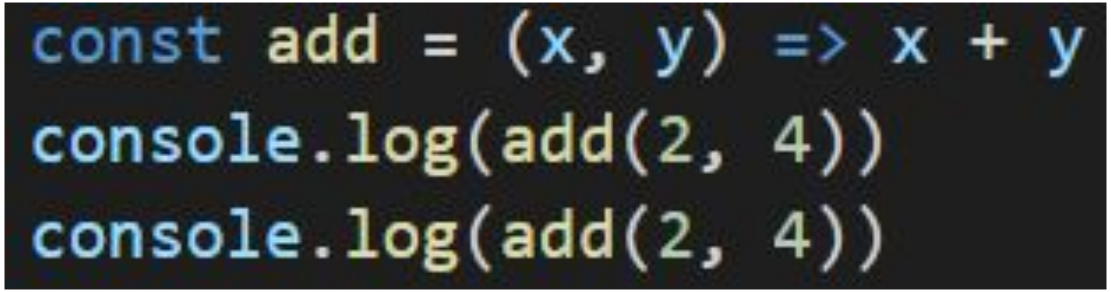
Functions are the building blocks of functional programming, and we want to keep them as pure as possible. That generally means we don’t want to change, or mutate data.

What is an isolated function?

An isolated function does not depend on global variables. Any information the function needs should be passed into it as an argument.

What is a pure function?

Here, for example, there is no dependence on variables outside the function. The information is passed as arguments.



```
const add = (x, y) => x + y
console.log(add(2, 4))
console.log(add(2, 4))
```

Different types of functions

Because functions form our building blocks, it’s important to know about some different kinds of functions we’ll use a lot in functional programming.

What is what are higher order functions?

These include…

- **Higher order functions:** functions that take a function as an argument, or

  return a function as a value.

What is a callback function?

- **Callbacks:** a callback is a function passed into another function as an argument to be executed later.

What is a lambda function

- **Lambda:** This is essentially another way to say an arrow function.

First class functions

- **First-class functions:** Functions that can be assigned to a variable, passed into another function, or returned from another function just like any other normal value. (In JS, this is all functions)