Recursion in JavaScript

Introduction

4 min

Recursion is a powerful tool for solving problems that require the execution of a similar action multiple times until a certain condition is met. For many problems, a recursive solution will result in fewer lines of code and will be easier to comprehend than a solution that uses a for or while loop.

You may find that recursion is a difficult concept to wrap your head around at first. That's fine! This lesson is meant as an introduction. As you see more examples, you will start to feel comfortable with the concept.

In this lesson, you will learn about recursion while implementing a function that returns the factorial of a number. Factorial is the product of an integer and all positive numbers less than it.

Let's consider 4 factorial:

4!=4×3×2×1=244!=4×3×2×1=24

Four factorial is equal to the product of 4 x 3 x 2 x 1, which is 24. The exclamation mark denotes that the number 4 is being factorialized.

1! and 0! are both valid base cases of factorial. The factorial product of both numbers is 1.

Before we dive into recursion, you will consider how factorial is implemented with an *iterative* approach.

Instructions

1. Checkpoint 1 Passed

1.

index.js includes a function called iterativeFactorial(). The function accepts an integer as an argument, and returns the factorial of it.

Take a look at how we implemented the function. Run the code when you're ready to move to the next checkpoint.

2. Checkpoint 2 Passed

2.

Set a constant named fourFactorial equal to the value returned from iterativeFactorial(), with 4 as the argument.

Hint

You can pass 4 into the function as follows:

iterativeFactorial(4);

3. Checkpoint 3 Passed

3.

Log the value saved to fourFactorial to the console.

Hint

Use console.log() as shown below:

```
console.log(fourFactorial);
```

index.js

```
const iterativeFactorial = (n) => {
  let result = 1;
  while(n > 0) {
    result *= n;
    n -= 1;
  }
  return result;
}

// Set fourFactorial
const fourFactorial = iterativeFactorial(4);

console.log(fourFactorial)


module.exports = {
  iterativeFactorial
};
```