Recursive Function

## Introduction to the JavaScript recursive functions

A recursive function is a [function](https://www.javascripttutorial.net/javascript-function/) that calls itself until it doesn’t. And this technique is called recursion.

Suppose that you have a function called recurse(). The recurse() is a recursive function if it calls itself inside its body, like this:

function recurse() {

*// ...*

recurse();

*// ...*

}

Code language: JavaScript (javascript)

A recursive function always has a condition to stop calling itself. Otherwise, it will call itself indefinitely. So a recursive function typically looks like the following:

function recurse() {

if(condition) {

*// stop calling itself*

*//...*

} else {

recurse();

}

}Code language: JavaScript (javascript)

Generally, you use recursive functions to break down a big problem into smaller ones. Typically, you will find the recursive functions in data structures like binary trees and graphs and algorithms such as binary search and quicksort.

### **1) A simple JavaScript recursive function example**

Suppose that you need to develop a function that counts down from a specified number to 1. For example, to count down from 3 to 1:

The count down will stop when the next number is zero. Therefore, you add an [if condition](https://www.javascripttutorial.net/javascript-if/) as follows:

function countDown(fromNumber) {

console.log(fromNumber);

let nextNumber = fromNumber - 1;

if (nextNumber > 0) {

countDown(nextNumber);

}

}

countDown(3);Code language: JavaScript (javascript)

Output:

3

### 2**2) Calculate the sum of n natural numbers example**

Suppose you need to calculate the sum of natural numbers from 1 to n using the recursion technique. To do that, you need to define the sum() recursively as follows:

sum(n) = n + sum(n-1)

sum(n-1) = n - 1 + sum(n-2)

...

sum(1) = 1

The following illustrates the sum() recursive function:

function sum(n) {

if (n <= 1) {

return n;

}

return n + sum(n - 1);

}Code language: JavaScript (javascript)

## Summary

* A recursive function is a function that calls itself until it doesn’t
* A recursive function always has a condition that stops the function from calling itself.

# **An Introduction to JavaScript Arrow Functions**

ES6 arrow functions provide you with an alternative way to write a shorter syntax compared to the function expression.

The following example defines a function expression that returns the sum of two numbers:

let add = function (x, y) {

return x + y;

};

console.log(add(10, 20)); // 30

The following example is equivalent to the above add() function expression but use an arrow function instead:

let add = (x, y) => x + y;

console.log(add(10, 20)); // 30;

let add = (x, y) => x + y;

console.log(add(10, 20)); // 30;