[Scope](https://scotch.io/tutorials/understanding-scope-in-javascript" \l "toc-scope-in-javascript) — Understand the difference between global scope, function scope, and block scope. Understand which variables are available where. Know how the JavaScript engine performs variable lookup.

<https://scotch.io/tutorials/understanding-scope-in-javascript>

Scope is the accessibility of variables, functions, and objects in some particular part of your code during runtime. In other words, scope determines the visibility of variables and other resources in areas of your code. This will help you track changes and keep an account of who did what.

you will realize that scoping parts of your code helps improve efficiency, track bugs and reduce them.

scope provides some level of security to your code.

This will help you track changes and keep an account of who did what.

you will realize that scoping parts of your code helps improve efficiency, track bugs and reduce them.

In the JavaScript language there are two types of scopes:

* Global Scope
* Local Scope

Variables defined inside a function are in the local scope. And they have a different scope for every call of that function. This means that variables having the same name can be used in different functions. This is because those variables are bound to their respective functions, each having different scopes, and are not accessible in other functions.

JavaScript is a single-threaded language so it can only execute a single task at a time.

Variables defined inside a function are in local scope while variables defined outside of a function are in the global scope. Each function when invoked creates a new scope.

the Module Pattern which allows us to scope our functions using both public and private scopes in an object.

Challenge:

Define a function named callFunc that takes one argument, a function f. It should return an array containing the values f(0), f(0), f(1), f(1). You can only call f twice.

Answer: function callFunc(f) {

var f0, f1;

f0 = f(0);

f1 = f(1);

return [f0,f0,f1,f1];

}