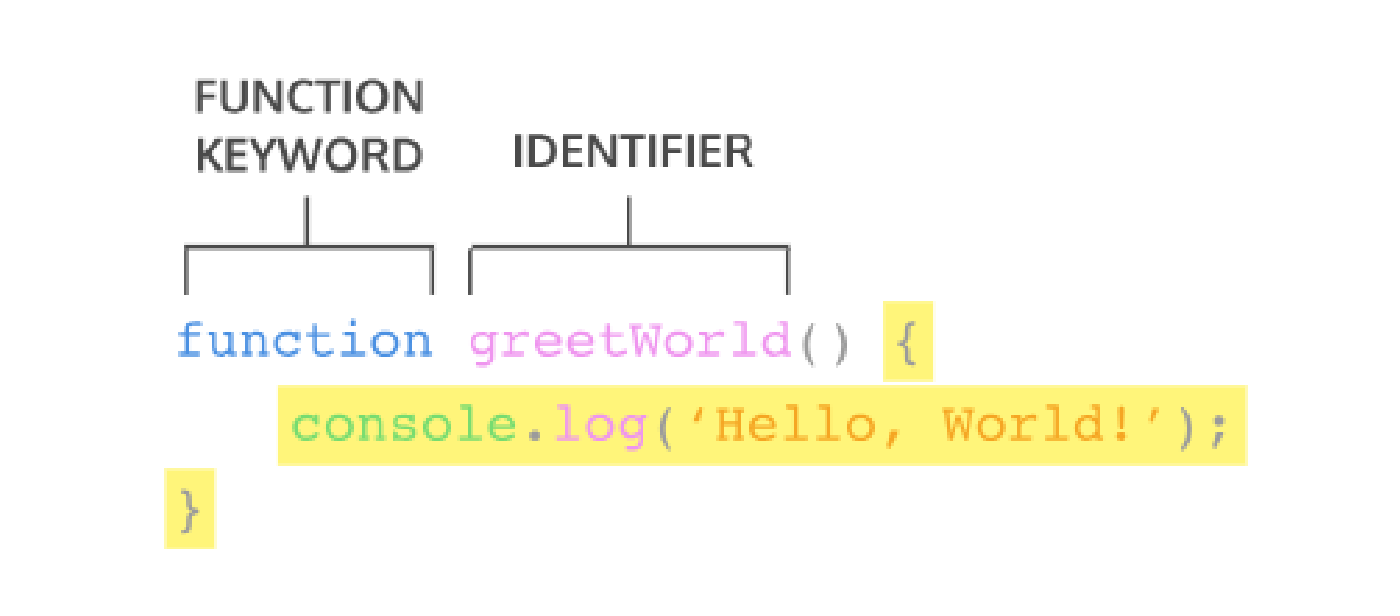
Javascript Functions

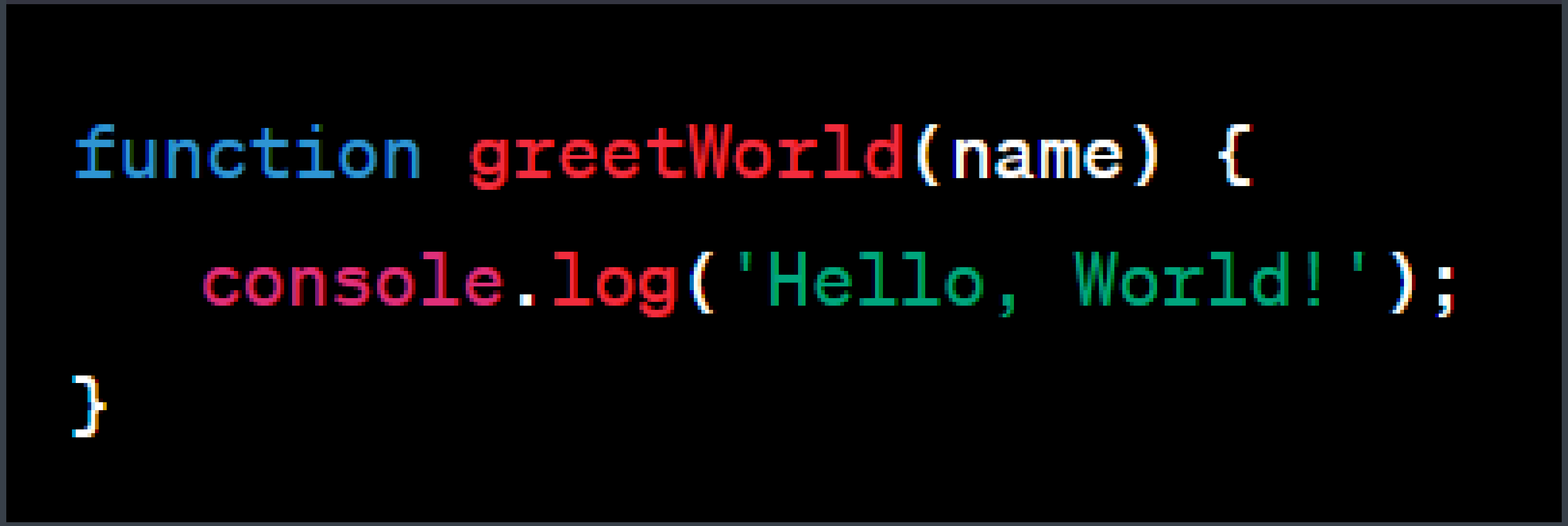
Javascript Functions

- Function Declaration
- Function Expression
- Parameters
- Arguments
- Return Statements
- Scope and Closures
- Anonymous Functions

Function Declaration



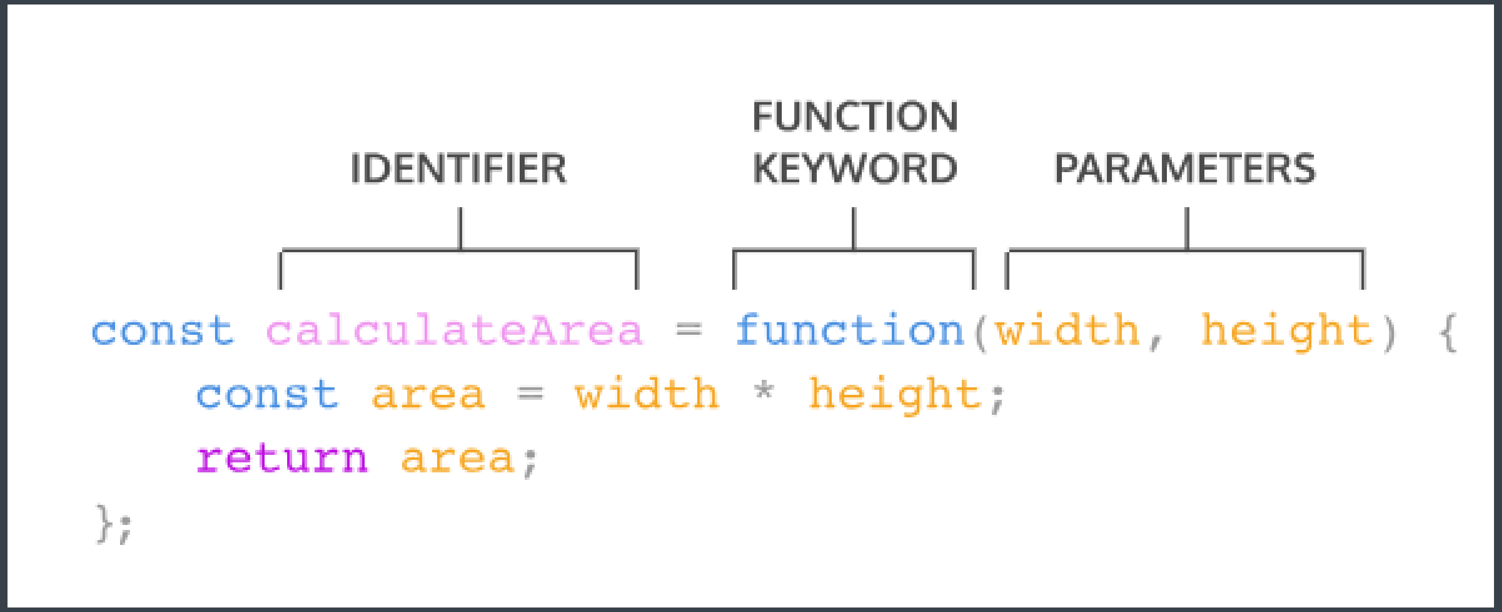
```
function greetWorld() {
    console.log(‘Hello, World!’);
}
```



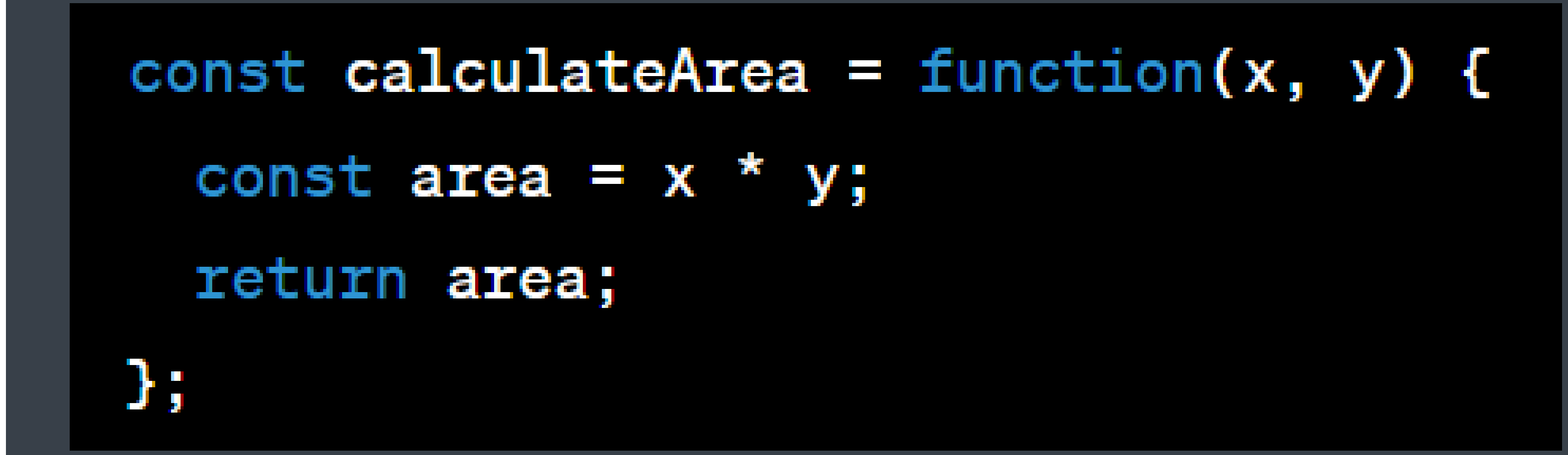
```
function greetWorld(name) {
   console.log('Hello, World!');
}
```

- JavaScript functions are defined with the function keyword.
- You can use a function declaration or a function expression.

Function Expression



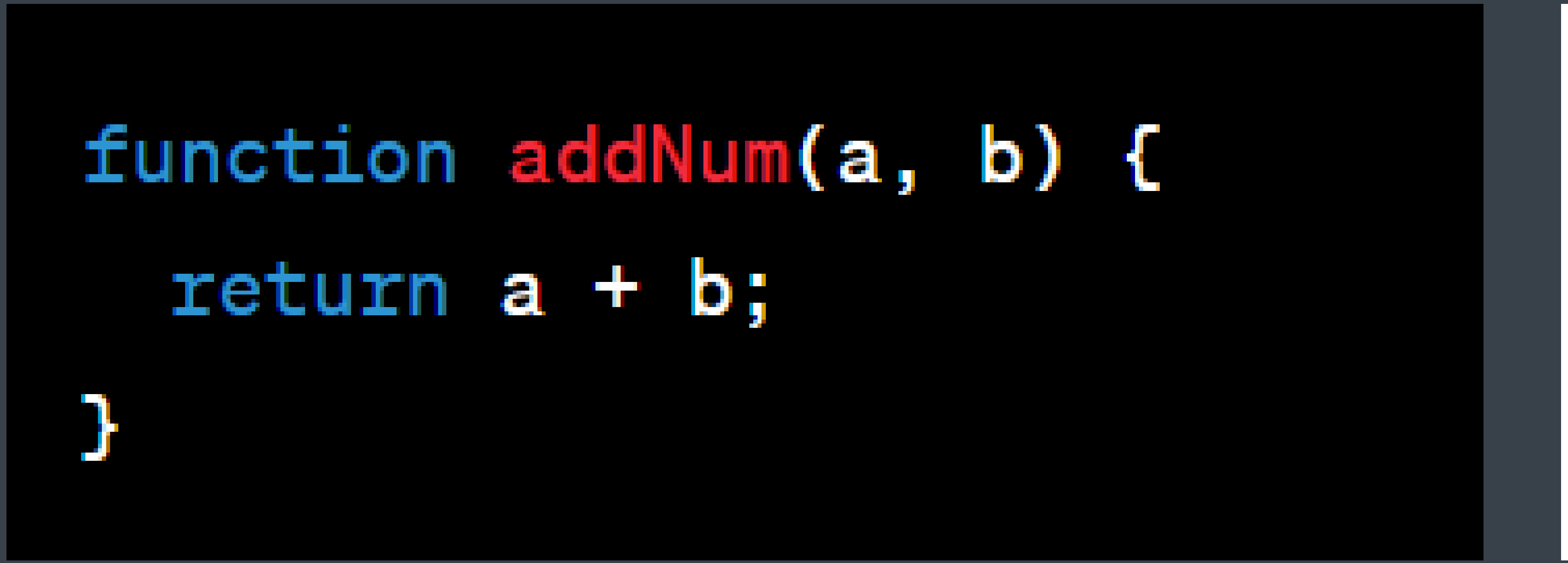
```
const calculateArea = function(width, height) {
    const area = width * height;
    return area;
};
```



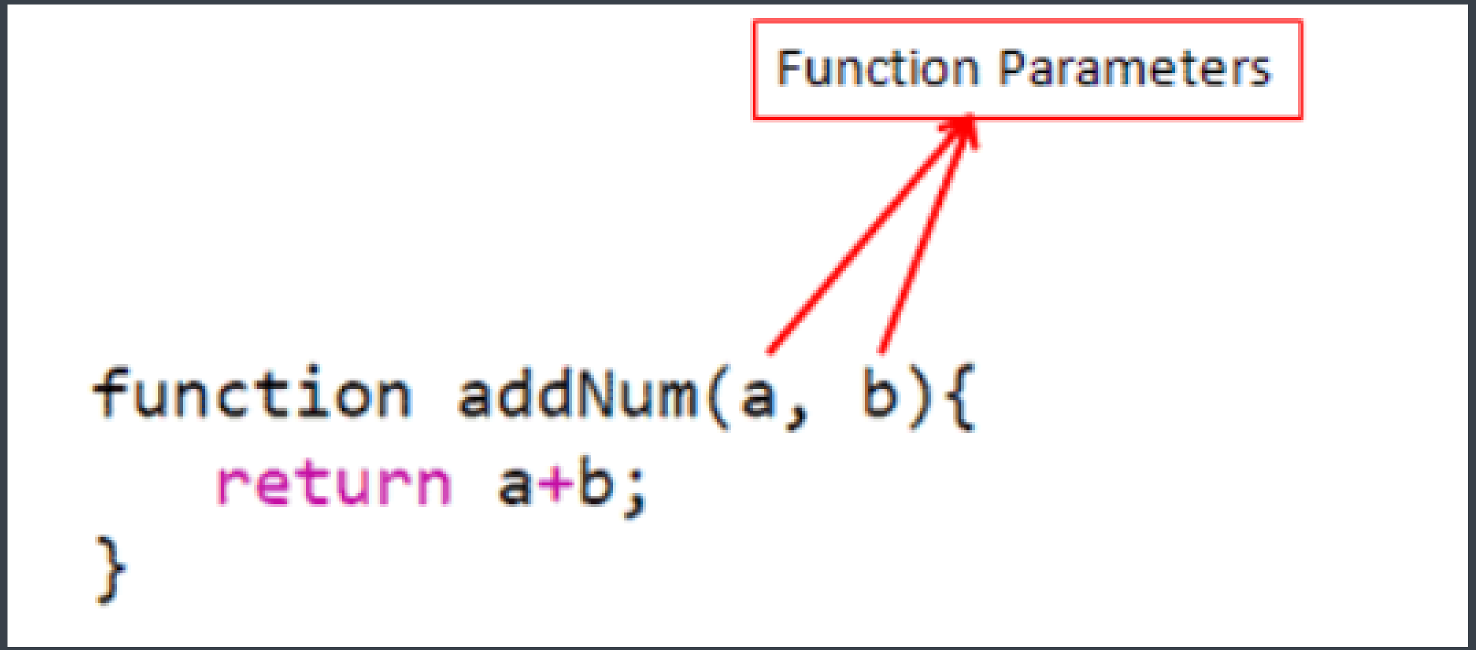
```
const calculateArea = function(x, y) {
  const area = x * y;
  return area;
};
```

- A JavaScript function can also be defined using an expression.
- A function expression can be stored in a variable

Parameters



```
function addNum(a, b) {
  return a + b;
}
```



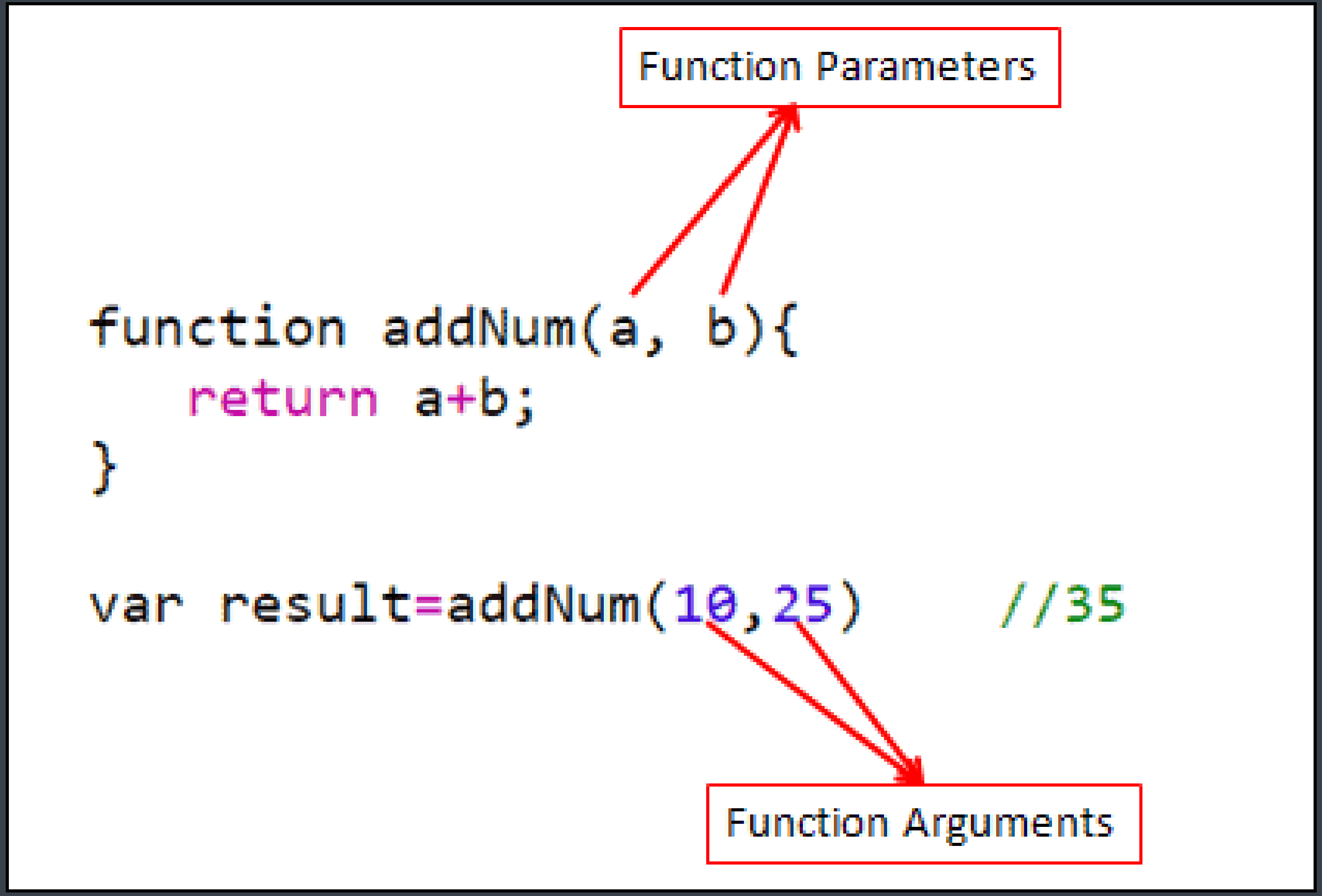
Function Parameters

```
function addNum(a, b){
    return a+b;
}
```

- Function parameters are the names listed in the function definition.

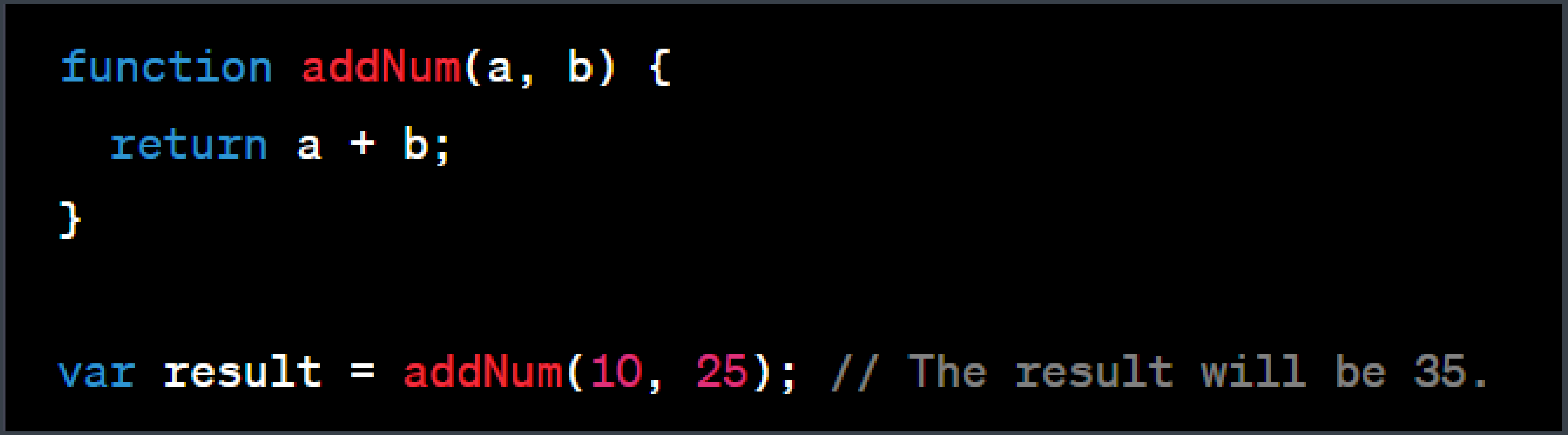
Arguments

- **Function arguments are the real values passed to (and received by) the function.**



```
function addNum(a, b){
   return a+b;
}

var result=addNum(10,25)    //35
```

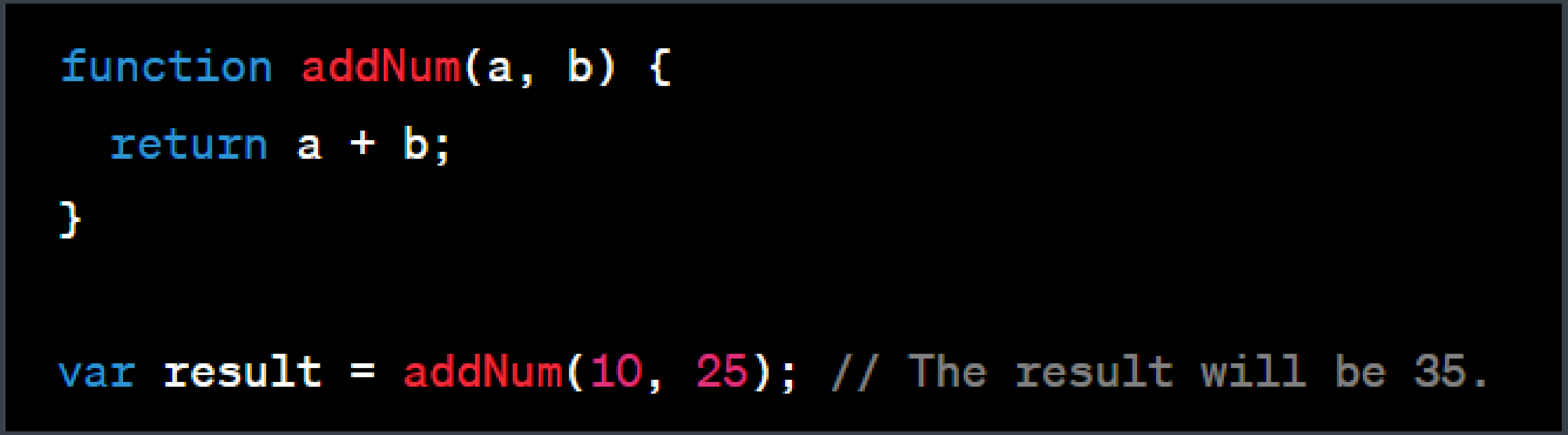


```
function addNum(a, b) {
  return a + b;
}

var result = addNum(10, 25); // The result will be 35.
```

Return Statement

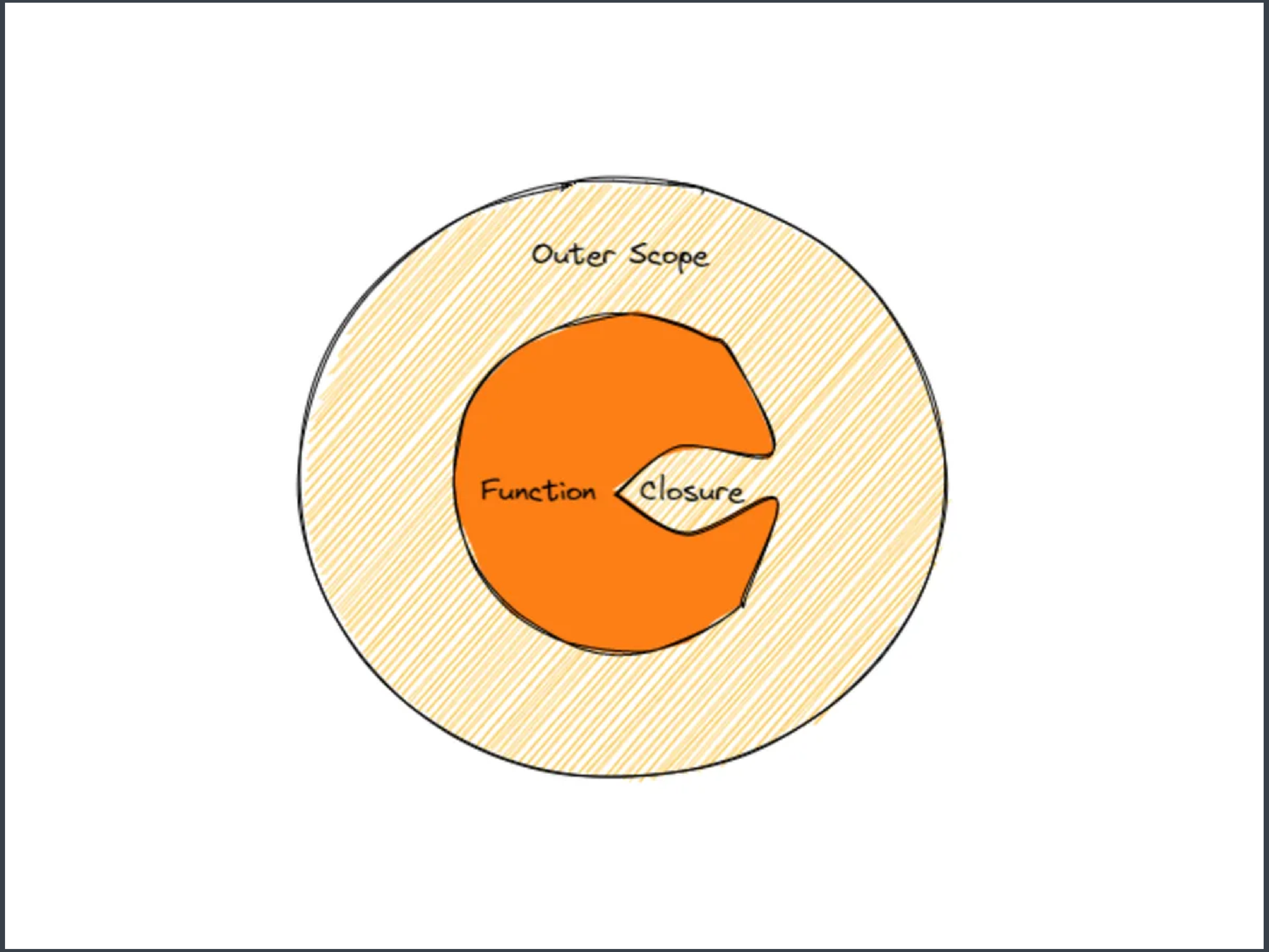
- The return statement in the add function returns the sum of x and y, which is stored in the result variable.es passed to (and received by) the function.



```
function addNum(a, b) {
  return a + b;
}

var result = addNum(10, 25); // The result will be 35.
```

Scope and Closure



Scope and Closure

- A scope in JavaScript defines what variables you have access to. There are two kinds of scope – global scope and local scope.
- A closure is usually returned so you can use the outer function’s variables at a later time.

Anonymous Function

- It is a function that does not have any name associated with it.
- An anonymous function is not accessible after its initial creation, it can only be accessed by a variable it is stored in as a function as a value.
- An anonymous function can also have multiple arguments, but only one expression.