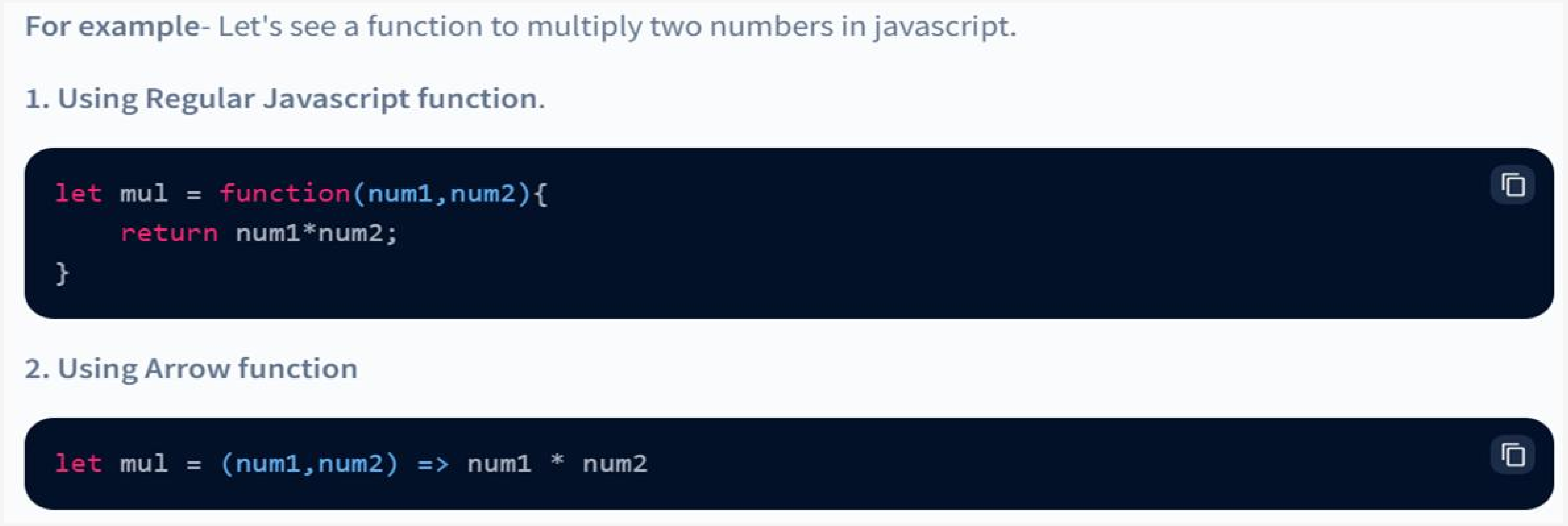
JavaScript – Arrow Function

Vijesh M. Nair
Assistant Professor
Dept. of CSE (AI-ML)

Arrow Function

- Is an alternative to regular function in Javascript.
- ES6 introduced the feature of the arrow function.



For example- Let's see a function to multiply two numbers in javascript.

1. Using Regular Javascript function.

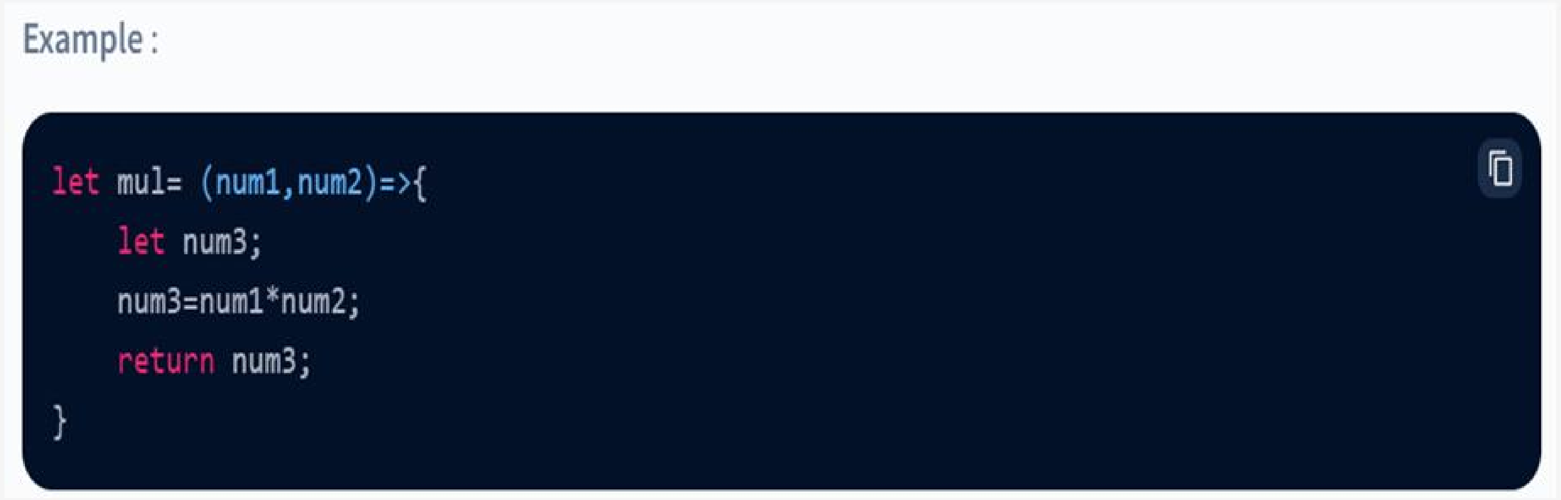
```
let mul = function(num1,num2){
    return num1*num2;
}
```

2. Using Arrow function

```
let mul = (num1,num2) => num1 * num2
```

Arrow Function

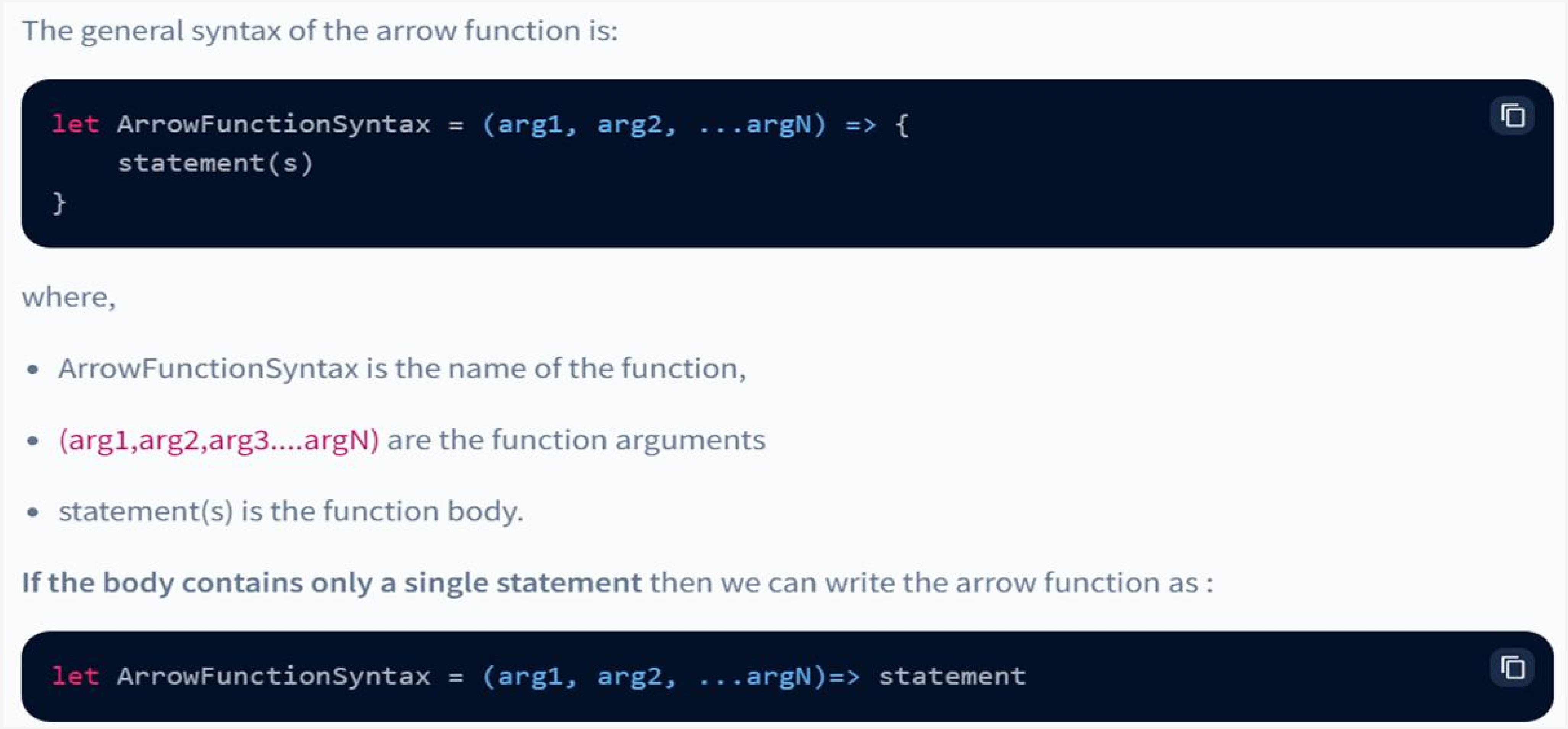
- If there is more than one statement after the arrow, then we need to use curly braces and the "return" keyword.



Example :

```
let mul= (num1,num2)=>{
    let num3;
    num3=num1*num2;
    return num3;
}
```

Arrow Function Syntax



The general syntax of the arrow function is:

```
let ArrowFunctionSyntax = (arg1, arg2, ...argN) => {
    statement(s)
}
```

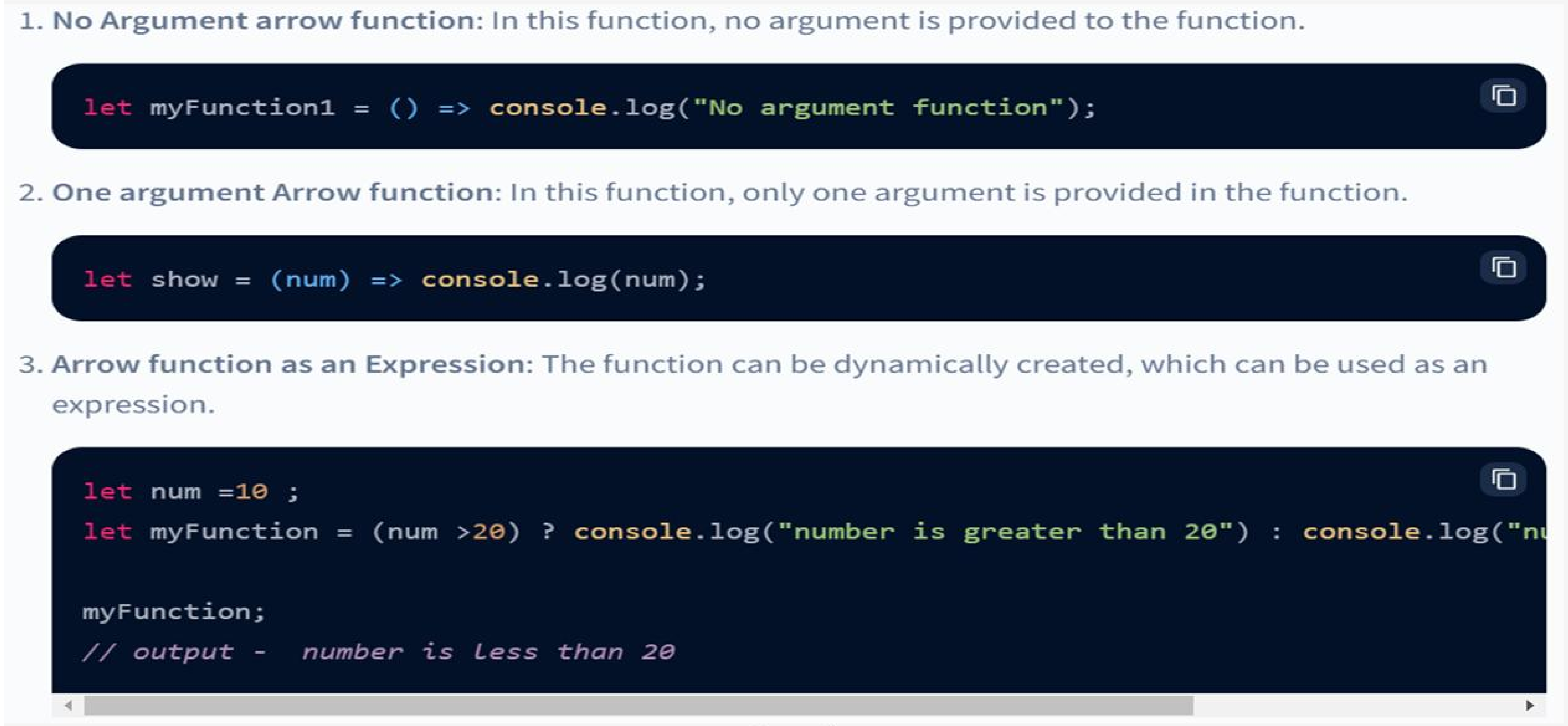
where,

- ArrowFunctionSyntax is the name of the function,
- (arg1,arg2,arg3....argN) are the function arguments
- statement(s) is the function body.

If the body contains only a single statement then we can write the arrow function as :

```
let ArrowFunctionSyntax = (arg1, arg2, ...argN)=> statement
```

Types of Arrow Function



1. **No Argument arrow function**: In this function, no argument is provided to the function.

```
let myFunction1 = () => console.log("No argument function");
```

2. **One argument Arrow function**: In this function, only one argument is provided in the function.

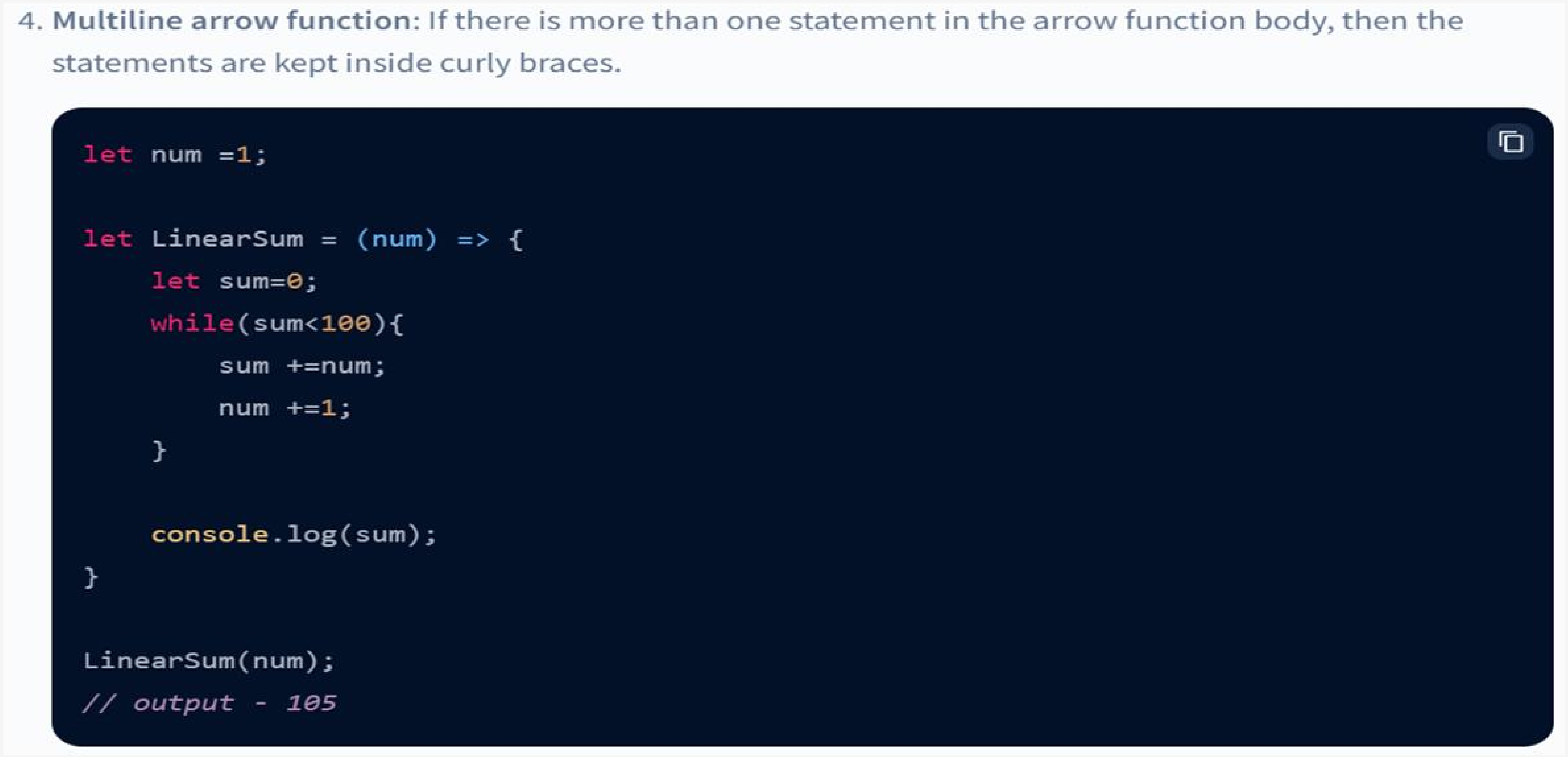
```
let show = (num) => console.log(num);
```

3. **Arrow function as an Expression**: The function can be dynamically created, which can be used as an expression.

```
let num =10 ;
let myFunction = (num >20) ? console.log("number is greater than 20") : console.log("nu

myFunction;
// output -  number is less than 20
```

Types of Arrow Function



4. **Multiline arrow function**: If there is more than one statement in the arrow function body, then the statements are kept inside curly braces.

```
let num =1;

let LinearSum = (num) => {
    let sum=0;
    while(sum<100){
        sum +=num;
        num +=1;
    }

    console.log(sum);
}

LinearSum(num);
// output - 105
```

Setting CSS Styles with JavaScript

Need to change CSS with Javascript

1. To load stylesheets dynamically.

2. After an initial page render, sometimes, there will be a need to change a particular style for the element in a page dynamically. In cases like this, writing CSS in Javascript helps.

3. If on any webpage some progress is happening and we want to change the look and feel after progress is finished. Here as well, changing styles in Javascript helps.

Js CSS with Element Style Property

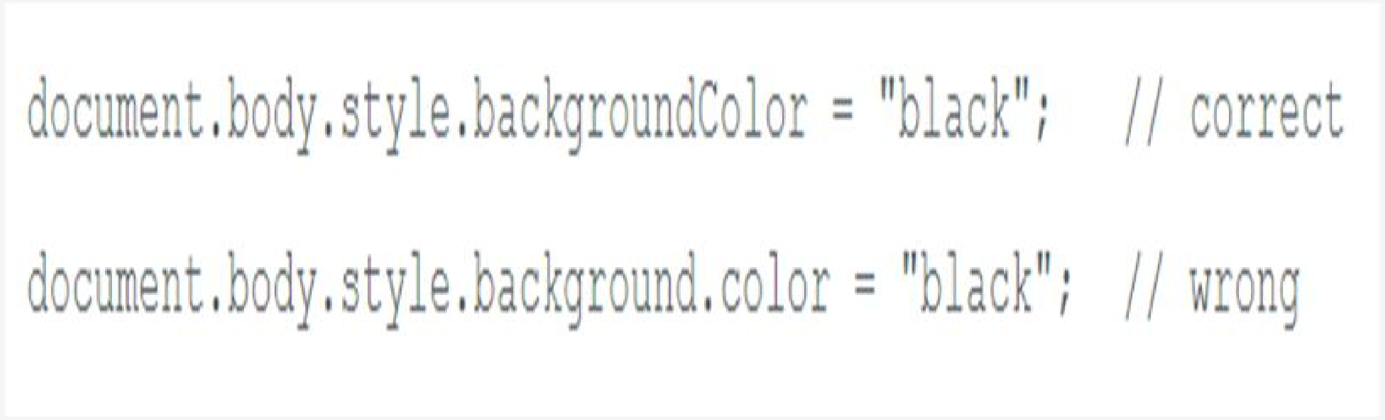
- Every HTML element in the Javascript DOM contains a Javascript object property called **style**.
- The **style object** contains many properties that correspond to CSS properties, so we can set properties on the style object to change element CSS styles directly.

```
color: black;                          // CSS

document.body.style.color = "black" // Javascript CSS
```

Js CSS with Element Style Property

- All CSS properties are accessible directly through the **style object**; there is no need to wade through sub-objects to find the CSS property.



```
document.body.style.backgroundColor = "black";    // correct

document.body.style.background.color = "black";  // wrong
```

Javascript CSS Versus External CSS

- Javascript CSS looks slightly different from external CSS because Javascript has different linguistic conventions than pure CSS.
- Many properties in external CSS files include hyphens in their names, and Javascript does not allow hyphens in its property names.

```
// The "margin" property in external CSS and Javascript CSS.
margin: 0px;
document.body.style.margin = "0px";


// The "background color" property in external CSS and Javascript CSS.
background-color: white;
document.body.style.backgroundColor = "white";
```

What should I use, Javascript CSS / CSS..?

- When choosing whether to use Javascript CSS, maintenance is a major concern.
- It can be more difficult to maintain Javascript CSS than to maintain external CSS files because you need to know Javascript and CSS simultaneously.
- Since Javascript is not always up-to-date with the newest CSS features, it is best to use external CSS when relying on the newest CSS.
- Well-established CSS properties are reliable in Javascript, but anything bleeding-edge won't be available in the Javascript standard immediately.

Thank You!