BACK-END

FUNÇÕES JAVASCRIPT

AULA 3

Função = Bloco de código definido que pode ser chamado em qualquer lugar sem a necessidade de reescrevê-lo.

**Sintaxe básica**

Para definir a função uma função no JavaScript, é utilizada a palavra **function:**

Function nomeDaFuncao(parametro1, parametro2) {  
 return

.....

}

**Parâmetros x Argumentos**

Parâmetros: Variáveis listadas na definição da função;

Argumentos: Valores passados para a função quando chamada

Ex:

function soma(a,b(**PARAMETROS**)) {

Return a + b

}

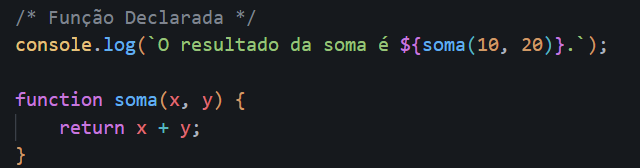
let resultado = soma(5,3(**ARGUMENTOS**))

console.log(resultado) //exibe 8 (5+3)

**Funções Declaradas**

Funções definidas utilizando apenas **function.** Essas funções são **hoisted**, elas podem ser chamadas antes mesmo do momento de sua criação.

Ex:

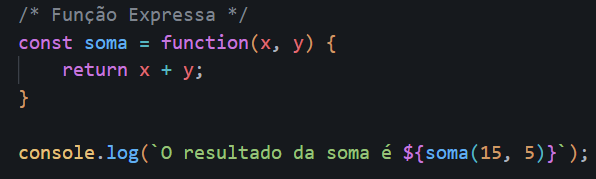


// Exibe 30

**Funções expressas/anônimas**

Funções atribuídas a variáveis. **Não** **são** **hoisted**, então só podem ser chamadas depois de serem criadas. Além disso, essa função não é nomeada, seu nome é dado pela variável

Ex/:

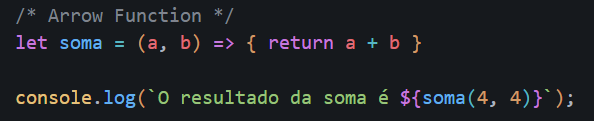


// Exibe 20

**Arrow Functions (Mais utilizada)**

Funções expressas, porém, com sintaxe mais curta.

Ex:

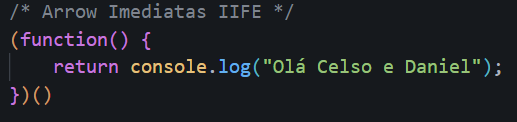


// Exibe 8

**Arrow Imediatas IIFE**

Funções que executam imediatamente após serem definidas.

Ex:

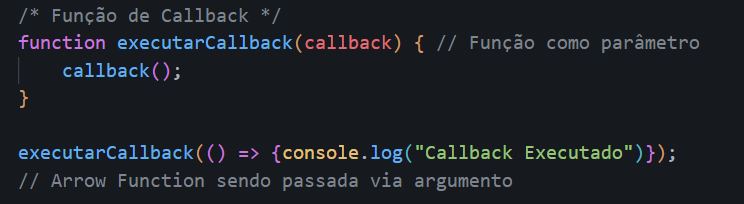


// Exibe Olá Celso e Daniel

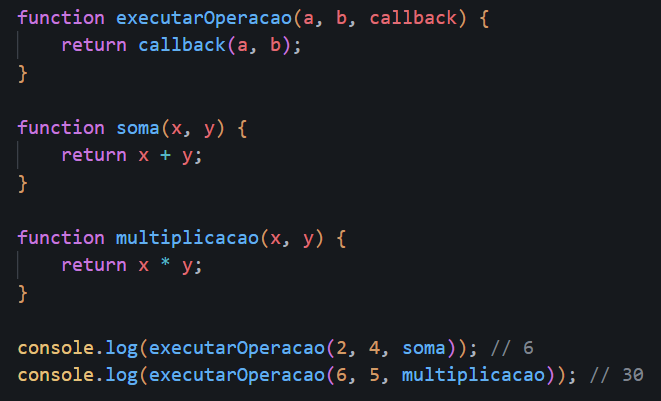
**Função de Callback**

Uma função passada como **argumento** para outra função, é invocada dentro da outra função.

Ex:



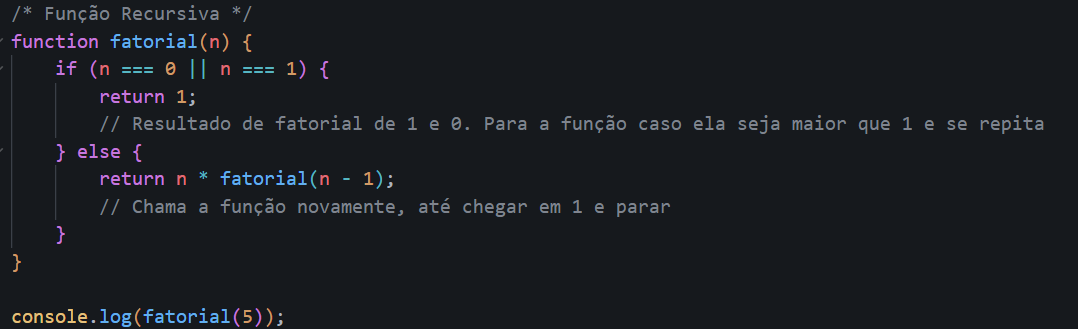
Ex2:



**Funções Recursivas**

Função que chama a si mesma.

Ex:

// Exibe 120

**Funções Assíncronas**

Função que realiza operações que demoram um certo tempo para serem concluídas (como consultas a banco de dados, leitura de arquivos, aquisições de rede, etc.) sem bloquear o resto dos códigos (enquanto ela é executada, outras coisas também são executadas). Para utilizar essa função, utilizamos a palavra async e await.

Ex:

Exibe: 