

FULL STACK WEB DEVELOPER

FERNANDO LIRA

FLAG

JAVASCRIPT

APRESENTAÇÃO – FERNANDO LIRA

3



it.fernandolira@gmail.com



<https://www.linkedin.com/in/fernandolira74/>



+351 93 317 99 21



@fernandolira74



Código sem função

```
const d1 = 6;  
const m1 = 12;  
const a1 = 2021;  
  
console.log(`Dia: ${d1}`);  
console.log(`Mês: ${m1}`);  
console.log(`Ano: ${a1}`);  
  
const d2 = 7;  
const m2 = 8;  
const a2 = 2022;  
  
console.log(`Dia: ${d2}`);  
console.log(`Mês: ${m2}`);  
console.log(`Ano: ${a2}`);
```

Código com função

```
const d1 = 6;
const m1 = 12;
const a1 = 2021;

function exibirData1() {
  console.log(`Dia: ${d1}`);
  console.log(`Mês: ${m1}`);
  console.log(`Ano: ${a1}`);
}

const d2 = 7;
const m2 = 8;
const a2 = 2022;

function exibirData2() {
  console.log(`Dia: ${d2}`);
  console.log(`Mês: ${m2}`);
  console.log(`Ano: ${a2}`);
}

exibirData1();
exibirData1();
exibirData1();

exibirData2();
exibirData2();
exibirData2();
```

- **Entrada de dados - Processamento - Saída de dados**
- **Entrada de dados - Processamento**
- **Processamento - Saída de dados**
- **Processamento**

Anatomia da função

```
function somar (num1, num2) {  
    sentença de código;  
    sentença de código;  
    sentença de código;  
    ...  
}
```

Anatomia da função

```
const d1 = 6;  
const m1 = 12;  
const a1 = 2021;  
  
const d2 = 7;  
const m2 = 8;  
const a2 = 2022;  
  
function exibirData(d, m, a) {  
    console.log(`Dia: ${d}`);  
    console.log(`Mês: ${m}`);  
    console.log(`Ano: ${a}`);  
}  
  
exibirData(d1,m1,a1);  
exibirData(d2,m2,a2);  
exibirData(4,6,2022);
```


EXERCÍCIOS

Desafios

Desenvolver duas funções – função somar e função subtrair em que recebem dois parâmetros e indicam o resultado da sua soma e da sua subtração respectivamente

Apresente a tabuada de um número

Apresente metade de uma árvore de natal com ? para N linhas

```
#  
##  
###  
####  
#####  
#####  
#####
```

Extra – o undefined, o null e o NaN

```
var a;  
var b = 7;  
var c = null;  
  
console.log(a);  
console.log(b);  
console.log(c);  
console.log(a + b);  
console.log(b + c);
```

Função e o undefined

```
function semReturn() {  
    console.log("A função foi chamada!");  
}  
  
var a = 5;  
var b = semReturn();  
  
console.log(a);  
console.log(b);
```

Função - Return

```
function retornaSempreUm() {  
    return 1;  
}  
  
var a = retornaSempreUm();  
  
console.log(a);
```

Função - Return

```
function retornaTextoOuNumero(texto) {  
    if (texto)  
        return "Retornou texto";  
    else  
        return 999;  
    //return texto ? "Retornou texto" : 999;  
}  
  
var a = retornaTextoOuNumero();  
  
console.log(a);
```

- **Entrada de dados - Processamento - Saída de dados**
- **Entrada de dados - Processamento**
- **Processamento - Saída de dados**
- **Processamento**

EXERCÍCIOS

Desafios

Implemente um programa que imprima o número de divisores de um inteiro dado.
Exemplo de dados: 6
Resultado: 4 (pois 6 tem 4 divisores, 1, 2, 3 e 6).

Implemente um programa que determine se um inteiro dado é primo.
Exemplo de dados: 23
Resultado: Sim

Um número n é perfeito se a soma dos divisores inteiros de n (excepto o próprio n) é igual ao valor de n . Por exemplo, o número 28 tem os seguintes divisores: 1, 2, 4, 7, 14, cuja soma é exactamente 28. (Os seguintes números são perfeitos: 6, 28, 496, 8128.) Escreva um programa que verifique se um número é perfeito.

Desafios

Indique o número de dias existentes num mês (Fevereiro=28)

```
const mes = 4
```

Verifique se um ano é bissexto

```
const ano = 2018
```

[//https://docs.microsoft.com/pt-pt/office/troubleshoot/excel/determine-a-leap-year](https://docs.microsoft.com/pt-pt/office/troubleshoot/excel/determine-a-leap-year)

Verifique se uma data é válida

```
const dia = 15
```

```
const mes = 12
```

```
const ano = 2021
```

Armazenar uma função numa variável

```
function soma(a,b) {  
    return a+b;  
}  
  
console.log(soma(4,5));  
  
//Equivalente com uma função anónima e  
armazenada numa variável  
  
var resultadoSoma = function (a,b) {  
    return a+b;  
}  
  
console.log(resultadoSoma(4,5));
```

Armazenar uma função numa variável

```
function soma(a,b) {  
    return a+b;  
}  
  
var resultadoSoma = soma;  
  
console.log(soma(4,5));  
console.log(resultadoSoma(9,7));
```

EXERCÍCIOS

Desafios

Escreva uma função que receba um número inteiro e devolva o número de dígitos que esse número inteiro tem.

Construa uma função que recebe um valor inteiro e verifica se o valor é par ou ímpar. A função deve retornar um valor booleano.

Construa uma função que recebe um valor mínimo e um valor máxima e retorne um número gerado aleatoriamente entre esses valores.

Passar uma função como argumento

```
function executar(fn) {  
    fn();  
}
```

Passar uma função como argumento

```
function executar(fn) {  
    fn();  
}  
  
executar(3); // erro porque 3();
```


Passar uma função como argumento

```
function executar(fn) {  
    if (typeof fn === "function")  
        console.log(fn());  
}  
  
function bomDia() {  
    return "Bom dia!";  
}  
  
executar(bomDia()); // ou executar(bomDia);
```

Passar uma função como argumento

```
function bomDia() {  
    return "Bom dia!";  
}  
  
const x = bomDia;  
const y = bomDia();  
  
console.log(x);      //[Functio: bomDia]  
console.log(x());    //Bom dia!  
console.log(y);      //Bom dia!  
console.log(y());    //Erro!
```

Parâmetros de funções

```
function soma(a, b, c, d) {  
    return a + b + c + d;  
}  
  
console.log(soma(1,2,3,4)); //10  
console.log(soma(1,2,3)); //NaN
```

Parâmetros de funções

```
function soma(a, b, c, d = 0) { //Valor padrão quando d não é  
    fornecido  
    return a + b + c + d;  
}  
  
console.log(soma(1,2,3,4)); //10  
console.log(soma(1,2,3)); //6
```

Passar uma função como argumento

```
function executar(fn) {  
    if (typeof fn === "function")  
        console.log(fn());  
}  
  
function soma(a, b) {  
    return a + b;  
}  
  
executar(soma); //NaN
```

Passar uma função como argumento

```
function executar(fn, n1, n2) {  
    if (typeof fn === "function")  
        console.log(fn(n1, n2));  
}  
  
function soma(a, b) {  
    return a + b;  
}  
  
executar(soma, 7, 8);
```

Retornar uma função

```
function retornaUmaFuncao () {  
  
    function bomDia() {  
        return "Bom dia!";  
    }  
  
    return bomDia;  
  
}  
  
console.log(retornaUmaFuncao); //[Function: retornaUmaFuncao]  
console.log(retornaUmaFuncao()); //[Function: bomDia]  
console.log(retornaUmaFuncao())(); //Bom dia!
```

Retornar uma função

```
console.log(retornaUmaFuncao()); //Bom dia!  
  
//Equivalente a...  
const umaFuncao = retornaUmaFuncao();  
console.log(umaFuncao());
```


Retornar uma função

```
function retornaUmaFuncaoAnonima() {  
    return function () {  
        return "Boa tarde!";  
    }  
}
```

```
console.log(retornaUmaFuncaoAnonima); //[Function: retornaUmaFuncao]  
console.log(retornaUmaFuncaoAnonima()); //[Function: (anonymous)]  
console.log(retornaUmaFuncaoAnonima()); //Bom tarde!
```

Retornar uma função

```
function retornaUmaFuncaoAnonima() {  
  return function () {  
    return function () {  
      return "Boa tarde!"  
    }  
  }  
}  
  
console.log(retornaUmaFuncaoAnonima); //[Function: retornaUmaFuncao]  
console.log(retornaUmaFuncaoAnonima()); //[Function: (anonymous)]  
console.log(retornaUmaFuncaoAnonima()); //[Function: (anonymous)]  
console.log(retornaUmaFuncaoAnonima()()); //Bom tarde!
```

Retornar uma função - Currying

Em ciência da computação, currying é uma técnica de transformação de uma função que recebe múltiplos parâmetros (mais especificamente, uma n-tupla como parâmetro) de forma que ela pode ser chamada como uma cadeia de funções que recebem somente um parâmetro cada.

Retornar uma função - Currying

```
function somar(a, b) {  
    return a + b;  
}  
  
console.log(somar(4,5));  
  
function somaApenasUm (a) {  
    return function (b) {  
        return a + b;  
    }  
}  
  
console.log(somaApenasUm(4)(5));
```

Retornar uma função - Currying

```
function somaApenasUm (a) {  
    return function (b) {  
        return a + b;  
    }  
}  
  
console.log(somaApenasUm(4)(5));  
  
const somarUm = somaApenasUm(5);  
  
console.log(somarUm); //[Function: anonymous]  
console.log(somarUm()); //Nan porque b não é fornecido  
console.log(somarUm(7)); //12  
console.log(somarUm(8)); //13  
console.log(somarUm(5)); //10
```

EXERCÍCIOS

Desafios

Escreva um programa para extrair os algarismos que compõem um número e os visualize individualmente.

Escreva um programa que calcule a soma dos algarismos que compõem um número. Por exemplo: $7258 = 7+2+5+8 = 22$

Escreva um programa que dado um inteiro positivo N e verifica se ele contém o dígito 3 em qualquer posição.