Aula 2 – Function e Arrow Function

Lista de Exercícios

Funções Tradicionais

1. Crie uma função chamada “saudar” que receba um nome e retorne a frase: "Olá, <nome>!".
2. Crie uma função “chamada” dobrar que receba um número e retorne o dobro.
3. Crie uma função “ehPar” que retorne true se um número for par e false caso contrário.
4. Crie uma função “media3” que receba três notas e retorne a média delas.
5. Crie uma função “maiorNumero” que receba três números e retorne o maior deles.

Expressão de Função

6. Crie uma **expressão de função** chamada “tudoEmMaiusculo” que receba uma string e retorne em maiúsculo.
7. Crie uma função de expressão chamada “calcularAreaRetangulo” que receba base e altura e retorne a área.

Escopo e Hoisting

8. Escreva um código que chame a função exemplo() **antes de ela ser declarada** (função tradicional) e explique o resultado.
9. Crie uma função de expressão chamada testeHoisting e tente chamá-la antes de sua declaração. O que acontece?

Parâmetros e Argumentos

10. Crie uma função chamada login que receba um nome de usuário com valor default "Morador".
11. Crie uma função somarTudo que use **rest parameter** e some todos os números recebidos.
12. Crie uma função multiplicarTudo que use **rest parameter** e multiplique todos os números recebidos.
13. Crie uma função apresentarPessoa que receba nome, idade e cidade, retornando uma string formatada (“Nome: <nome>, Idade: <idade>, Cidade: <cidade>)

Arrow Functions – Sintaxe

14. Crie uma arrow function chamada “quadrado” que receba um número e retorne o quadrado dele.
15. Crie uma arrow function chamada “saudacaoCurta” que receba um nome e retorne: "Oi, <nome>".
16. Transforme a função media3 (do exercício 4) em uma arrow function.

17. Transforme a função maiorNumero (do exercício 5) em uma arrow function.

Arrow Functions com Arrays

18. Dado o array numeros = [1, 2, 3, 4, 5], use **map** com arrow function para gerar um novo array com o triplo de cada número.
19. Dado o array frutas = ["maçã", "uva", "banana"], use **map** com arrow function para gerar um novo array com todas em maiúsculo.
20. Dado o array numeros = [10, 3, 25, 8, 15, 2], use **filter** com arrow function para gerar um novo array contendo apenas os números maiores que 10.
21. Dado o array nomes = ["Ana", "Pedro", "João", "Carolina", "Lu"], use **filter** para gerar um novo array contendo apenas os nomes com mais de 4 letras.
22. Dado o array valores = [5, 10, 15, 20], use **reduce** para calcular a **soma total** de todos os elementos.
23. Dado o array palavras = ["JS", "é", "muito", "legal"], use **reduce** para concatenar os elementos em uma única string.

Desafio

1. Crie uma função de expressão chamada “inverterTexto” que receba a string “**O rato roeu a roupa do rei de Roma**” e retorne invertida. (utilize os métodos join() e reverse() do parâmetro informado)
2. Dado o array de produtos:

```
const produtos = [
  { nome: "Mouse", preco: 80, categoria: "periférico" },
  { nome: "Teclado", preco: 120, categoria: "periférico" },
  { nome: "Notebook", preco: 3500, categoria: "computador" },
  { nome: "Cadeira Gamer", preco: 900, categoria: "móvel" },
  { nome: "Monitor", preco: 1200, categoria: "periférico" }
];
```

Crie um script que utilize filter para gerar um novo array contendo apenas os produtos da categoria "**periférico**" **E QUE** o preço seja maior que 100.

Depois, exiba o resultado no console.

Abaixo o resultado esperado

```
[
  { nome: "Teclado", preco: 120, categoria: "periférico"},
  { nome: "Monitor", preco: 1200, categoria: "periférico"}
]
```