Por: Alice Dantas

Tipos de Escopo em JS

1. Escopo Global

- A variável pode ser acessada de qualquer parte do código.
- Criada fora de funções ou blocos.

```
let nome = "Alice";

function dizerOla() {
  console.log("Olá " + nome); // consigo acessar aqui
}

dizerOla(); // Olá Alice

console.log(nome); // Alice (acessível fora da função também)
```

2. Escopo Local de Função

- Variável criada **dentro da função** só existe lá dentro.
- Não pode ser acessada fora da função.

3. Escopo de Bloco

- Criado com { } (exemplo: if, for, while).
- Com let e const, a variável só existe dentro do bloco.
- Com var, a variável **ignora** esse escopo (mais perigoso).

```
if (true) {
  let a = 10;
  var b = 20;
  const c = 30;
  console.log(a, b, c); // 10 20 30
};

console.log(b); // funciona (var "vazou")
console.log(a); // erro
console.log(c); // erro
```

4. Escopo Léxico

- Funções "lembram" o lugar onde foram criadas.

```
function externa() {
  let cor = "azul";

  // criando uma função dentro de outra
  function interna() {
    console.log("A cor é " + cor); // consegue acessar
  }
  interna(); // chamada da função interna dentro da função externa
}

externa(); // A cor é azul
```

Hoisting em JS

Hoisting vem de *hoist*, que significa "erguer" ou "içar". No JavaScript, durante a fase de **compilação** (antes da execução), o interpretador **move as declarações de variáveis** e **funções declaradas para o topo do escopo atual**.

> Em outras palavras: o JavaScript primeiro *lê e reserva espaço na memória* para variáveis e funções declaradas antes de executar o código linha a linha. Logo, é possível chamar uma função declarada antes mesmo de criá-la.