Escopo de Função - Tópico 05

1- Por qual motivo o código abaixo retorna com erros?

```
{
var car = 'preto';
const marca = 'Flat';
let portas = 4;
}
console.log(var, marca, portas);
```

Correção: O erro ocorre porque **var** é uma palavra reservada para declaração de variáveis,
não pode ser usada como valor no console.log().
A correção seria usar o nome da variável **car**.
console.log(car, marca, portas);
E somente var fura escopo, mudado também as chaves para depois do comando

```
console.log().
}
```

2- Como corrigir o erro abaixo?

```
function somarDois(x) {
    const dois = 2;
    return x + dois;
}

function dividirDois(x) {
    return x + dois;
}

somarDois(4);
dividirDois(6);
```

Correção:

```
function somarDois(x) {
    const dois = 2;
    return x + dois;
}

function dividirDois(x) {
    const dois = 2;
    return x / dois;
}

dividirDois(6);
somarDois(4);
```

3- O que fazer para total retornar 500?

```
var numero = 50;

for(var numero = 0; numero < 10; numero++) {
    console.log(numero);

}
const total = 10 * numero;

console.log(total);
```

Solução 1: Usar uma variável diferente no loop.

```
var numero = 50;

for(var i = 0; i < 10; i++) {
    console.log(i);
}

const total = 10 * numero;
console.log(total);
```

Observações:

Variáveis declaradas dentro de funções não são acessadas fora das mesmas.

Declarar variáveis sem a palavra chave var, const ou let, cria uma variável que pode se acessar em qualquer escopo (global).
Isso é um erro. 'use strict' impede isso.

PAI
Variáveis declaradas no escopo pai da função, conseguem ser acessadas pelas funções.

ESCOPO DE BLOCO
Variáveis criadas com var varam o bloco. Por isso, com a introdução do ES6 a melhor forma de declararmos uma variável é utilizando const e let, pois estas respeitam o escopo de bloco.

{ } CRIAR UM BLOCO
Chaves { } criam um escopo de bloco, não confundir com a criação de objetos = { }.