Arrays e Loops

Array

É um grupo de valores geralmente relacionados. Servem para guardarmos diferentes valores em uma única variável.

Array

```
var videoGames = ['Switch', 'PS5', 'XBox'];
videoGames[0] // Switch
videoGames[2] // Xbox
```

Métodos e Propriedades de uma Array

Methods

```
var videoGames = ['Switch', 'PS5', 'XBox'];

videoGames.pop(); // Remove o último item e retorna ele
videoGames.push('3DS'); // Adiciona ao final da array
videoGames.length; // 3
```

For Loop

O for loop possui 3 partes, **início** , **condição** e **incremento**

For

```
for (var numero = 0; numero < 10; numero++) {
    console.log(numero);
}
// Retorna de 0 a 9 no console
```

While Loop

While

```
var i = 0;
while (i < 10) {
    console.log(i);
    i++;
}
// Retorna de 0 a 9 no console
```

Arrays e Loops

Loops em arrays

```
var videoGames = ['Switch', 'PS5', 'XBox', '3DS'];
for (var i = 0; i < videoGames.length; i++) {
    console.log(videoGames[i]);
}
```

O que acontece no código anterior?

retorna cada item do array no console

Break

Break

```
var videoGames = ['Switch', 'PS4', 'XBox', '3DS'];
for (var i = 0; i < videoGames.length; i++) {
    console.log(videoGames[i]);
    if(videoGames[i] === 'PS4') {
        break;
    }
}
```

O loop irá parar caso encontro e palavra break

forEach

forEach é um método que executa uma função para cada item da Array.

É uma forma mais simples de utilizarmos um loop com arrays (ou array-like)

for each

```
var videoGames = ['Switch', 'PS5', 'XBox', '3DS'];
videoGames.forEach(function(item) {
    console.log(item);
});
// O argumento item será atribuído dinamicamente
```

Podemos passar os seguintes parâmetros `item` , `index` e `array`.

Exercício

- Crie uma array com os anos que o Brasil ganhou a copa

  1958, 1962, 1970, 1994, 2002
- Interaja com a array utilizando um loop, para cada ano mostrar no console a seguinte mensagem:

  O brasil ganhou a copa de ANO;
- Interaja com um loop nas frutas abaixo e pare ao encontrar Pera

`var frutas = ['Banana', 'Maçã', 'Pera', 'Uva', 'Melância']`

- Coloque a última fruta do array acima em uma variável, sem remover a mesma da array, depois imprima essa variável no console.

ESCOPO DE FUNÇÃO

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

Escopo evita conflitos

```
function mostrarCarro() {
  var carro = 'Fusca';
  console.log(carro);
}

mostrarCarro(); // Fusca no console
console.log(carro); // Erro, carro is not defined
```

VARIÁVEL GLOBAL (ERRO)

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

Erro

```
function mostrarCarro() {
  carro = 'Fusca';
  console.log(carro);
}

mostrarCarro(); // Fusca
console.log(carro); // Fusca
```

ESCOPO DE FUNÇÃO (PAI)

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

Pai

```
var carro = 'Fusca';

function mostrarCarro() {
  var frase = `Meu carro é um ${carro}`;
  console.log(frase);
}

mostrarCarro(); // Meu carro é um Fusca
console.log(carro);  // Fusca
```

ESCOPO DE BLOCO

Variáveis criadas com `var`, vazam o bloco. Por isso com a introdução do ES6 a melhor forma de declarmos uma variável é utilizando `const` e `let`, pois estas respeitam o escopo de bloco.

escopo

```
if(true) {
  var carro = 'Fusca';
  console.log(carro);
}
console.log(carro);
```

VAR VAZA O BLOCO

Mesmo com a condição falsa, a variável ainda será declarada utilizando hoisting e o valor ficará como undefined.

var

```
if(false) {
  var carro = 'Fusca';
  console.log(carro);
}
console.log(carro); // undefined
```

CONST E LET NO LUGAR DE VAR

A partir de agora vamos utilizar apenas `const` e `let` para declarmos variáveis.

let

```
if(true) {
  const carro = 'Fusca';
  console.log(carro);
}
console.log(carro); // erro, carro is not defined
```

{} CRIA UM BLOCO

Chaves `{}` criam um escopo de bloco, não confundir com a criação de objetos `= {}`

{}

```
{
  var carro = 'Fusca';
  const ano = 2018;
}
console.log(carro); // Carro
console.log(ano); // erro ano is not defined
```

Exercício

- Por qual motivo o código abaixo retorna com erros?

ex1

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

- Como corrigir o erro abaixo?

ex2

```
function somarDois(x) {
    const dois = 2;
    return x + dois;
}
function dividirDois(x) {
    return x + dois;
}
somarDois(4);
dividirDois(6);
```

- O que fazer para total retornar 500?

ex3

```
var numero = 50;

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

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