

# Javascript

Programa de atração de formação de talentos  
Bradesco / Visionaire

**Aula 02 – Outros Tipo**

**Mark Joselli**

**Mark.Joselli@pucpr.br**



# Atividades

1. Crie as variáveis peso e altura leia e as inicialize com seu peso e sua altura. Calcule o valor do IMC ( $\text{peso} / \text{altura}^2$ ). Associe o valor verdadeiro a variável obeso caso o valor do IMC seja maior ou igual a 30. Imprima o texto, substituindo os valores em itálico pelas respectivas variáveis: O valor do IMC para *peso* quilos e *altura* metros é de *imc*.

2. Leia um número e aplique sobre ele a conjectura de Collatz. Ela diz que uma sequência pode ser feita com base na seguinte regra:

- Se o número  $n$  for par, o próximo é  $n / 2$
- Se for ímpar é  $3n+1$
- A sequência termina em 1

3. Chico tem 1,50 metro e cresce 2 centímetros por ano, enquanto Zé tem 1,10 metro e cresce 3 centímetros por ano. Construa um algoritmo que calcule e imprima as alturas de Chico e Zé até que Zé seja maior que Chico

# Listas (arrays)

É possível associar um conjunto de valores em listas

```
const tamanhos = [38, 39, 40, 41];
```

Um elemento pode ser acessado por índice, sendo o primeiro índice 0:

```
console.log(tamanhos[1]); //Imprime 39
```

```
tamanhos[2] = 100;
```

```
console.log(tamanhos);    //Imprime [38, 39, 100, 41]
```

# Listas (arrays)

Diferente de outras linguagens, os elementos de uma lista **não precisam** ser do mesmo tipo:

```
const elems = [38, "vinícius", null, [2,3], false];
```

Também é possível criar uma lista vazia:

```
const vazia = [];
```

O comando **push** adiciona um elemento ao final da lista. A lista crescerá se necessário.

# Listas

A lista implementa vários comandos úteis. Alguns deles são:

- **length**: Retorna o tamanho do array
  - **push / unshift**: Adiciona um item ao final / início
  - **pop / shift**: Remove um item do final / início
  - **splice(pos, n)**: Remove *n* elementos a partir do índice *pos*
  - **slice(inicio, fim)**: Copia o array da posição *inicio* até *fim*. Caso o fim não seja fornecido copia até o final. Caso nenhum seja fornecido, copia o array todo.
  - **indexOf**: Retorna o índice de um elemento
- 
- Veremos mais comandos nas próximas aulas

# for of (para cada)

- Há um tipo de iteração exclusivo para listas:

```
let tamanhos = [38, 39, 40, 41];  
for (const tamanho of tamanhos) {  
    console.log(`Tamanho: ${tamanho}`);  
}
```

Este comando foi criado para substituir o `for in`, que percorria a lista retornando seus índices, ao invés dos valores.

# Objetos

- Objetos permitem agrupar mais valores em uma única variável

```
const propriedade = "altura";  
let pessoa = {  
  nome: "Elias",  
  "idade": 41,  
  [propriedade]: 1.75  
};
```

# Objetos

- Os propriedades podem ser acessadas de duas formas:

```
console.log(pessoa.nome);  
console.log(pessoa["altura"]);
```

- Valores e propriedades podem ser adicionados a qualquer momento:

```
let campo = "sexo";  
pessoa.peso = 76;  
pessoa[campo] = "masculino";
```



# Objetos

- Os propriedades podem ser acessadas de duas formas:

```
console.log(pessoa.nome);  
console.log(pessoa["altura"]);
```

- Valores e propriedades podem ser adicionados a qualquer momento:

```
let campo = "sexo";  
pessoa.peso = 76;  
pessoa[campo] = "masculino";
```

Dica: Com a sintaxe dos []  
você pode usar variáveis!

# Objetos

- Você pode testar se uma propriedade existe ou não através do operador in

```
if ('cpf' in pessoa) {  
    //usa o pessoa.cpf  
}
```

Você pode excluir uma propriedade utilizando delete

```
delete pessoa.peso;
```

# Objetos

- É possível iterar sobre as chaves, valores ou até sobre os pares de um objeto através das funções:
  - *Object.keys*
  - *Object.values*
  - *Object.entries*

Exemplo:

```
for (const key of Object.keys(pessoa)) {  
    console.log(key);  
}
```

# Objetos

- Cuidado: Variáveis que operam sobre objetos trabalham com **referências**

```
let pessoa = {  
  nome: "Elias"  
};
```

```
let outraPessoa = pessoa;  
outraPessoa.nome = "Marcelo";
```

```
console.log(pessoa.nome); //Imprime Marcelo
```

# Objetos

- Obviamente, nada te impede de criar listas de objetos:

```
let pessoas = [  
  { nome: "Elias", idade: 21 },  
  { nome: "Lucas", idade: 18 },  
  { nome: "Pedro", idade: 23 },  
];
```

```
console.log(pessoas[1].nome); //Imprime Lucas
```

# Atividades



1. Crie uma lista com alguns valores e imprima:
  - Os valores positivos
  - A média de todos os valores
  
2. Crie uma lista com 15 pessoas lendo seu nome e cor.
  - As cores podem ser “branco”, “negro”, “pardo” e “outro”
  - Em seguida, mostre quantas pessoas tem cada cor
  - Tente resolver esse problema sem usar uma cadeia de “ifs” ou switch
  
3. Crie o jogo de adivinhar um número de 1 até 100.
  - Caso ele entre um número maior escreva “Maior”
  - Caso ele entre um número menor escreva “Menor”
  - Caso ele entre com um número fora do intervalo ( 0 ou maior que 100) fale “desistiu?” e acabe o jogo
  - O jogo deve perguntar até o usuário desistir ou falar o valor correto.

# Dúvidas?

- Materiais:
  - Livro Eloquent Javascript:
    - 3ª edição (inglês):  
<https://eloquentjavascript.net/>
    - 2ª edição (português):  
<https://github.com/braziljs/eloquente-javascript>
  - The Modern Javascript Tutorial:  
<https://javascript.info/>