

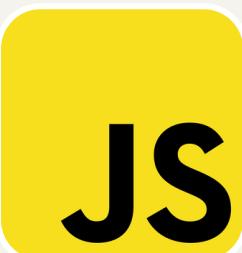


Estrutura de dados e funções básicas

Array

Conceito inicial e geral

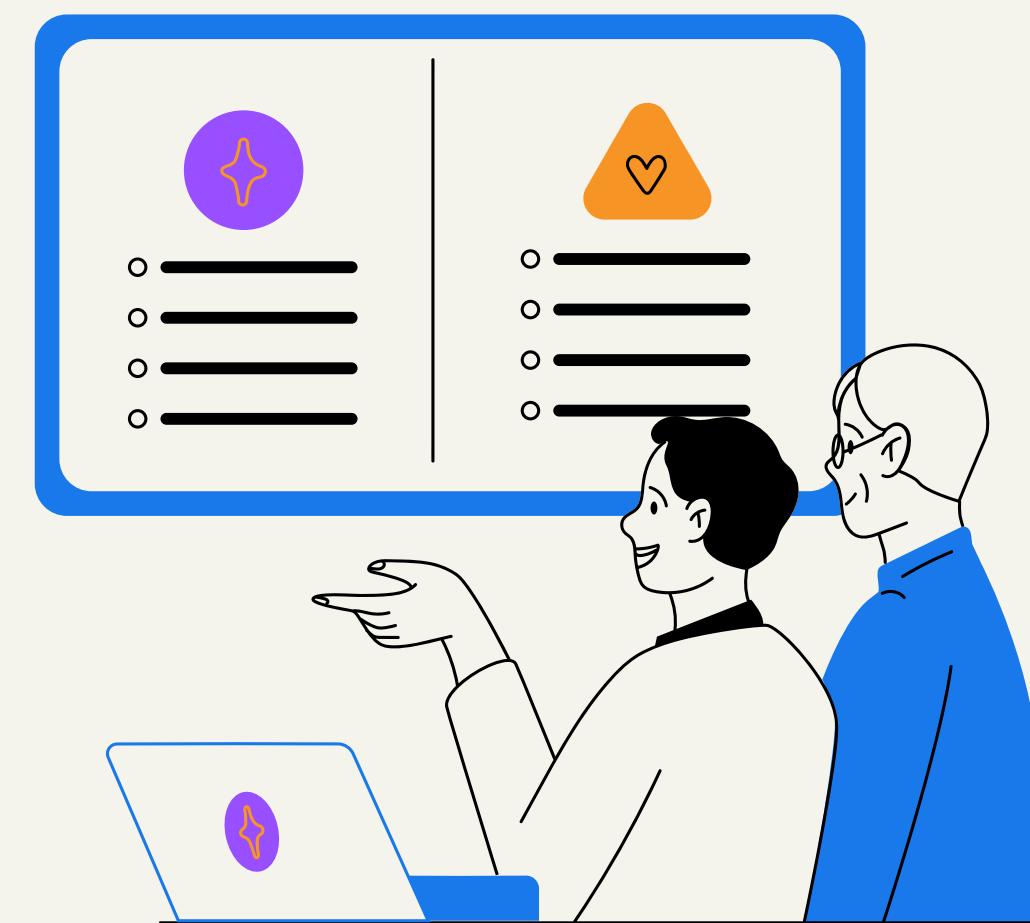
- Arrays e listas são estruturas de dados que permitem armazenar múltiplos valores em uma única variável.
- Eles são amplamente utilizados para organizar e processar conjuntos de dados.



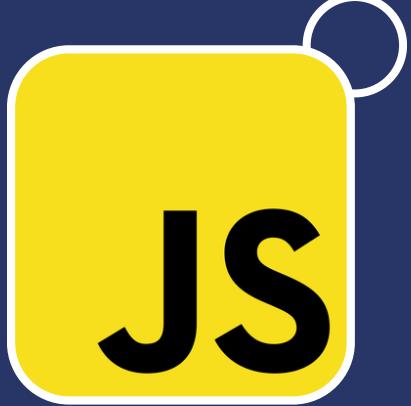
```
let frutas = ["Maçã", "Banana"];
```



```
frutas = ["maçã", "banana", "laranja"]
```



Array e Listas



Lista em JavaScript

- Em JavaScript, um array é uma coleção ordenada de valores, acessados por meio de índices numéricos, começando com 0.
- Acessar elementos

```
...
// Criando um array vazio
let meuArray = [];

// Inicializando um array com valores
let frutas = ["maçã", "banana", "laranja"];

// Acessando elementos do array pelo índice
console.log(frutas[0]); // Saída: "maçã"

// Modificando elementos do array
frutas[1] = "uva";

// Adicionando elementos ao final do array
console.log(frutas[1]); // Saída: "uva"
```



Listas em Python

- Uma lista é uma coleção ordenada de itens, onde cada item é identificado por um índice. Listas em Python são representadas por colchetes [] e os elementos dentro da lista são separados por vírgulas.

- Acessar elementos

```
...
# Criando uma lista vazia
minha_lista = []

# Inicializando uma lista com valores
frutas = ["maçã", "banana", "laranja"]

# Acessando elementos da lista pelo índice
print(frutas[0]) # Saída: "maçã"

# Modificando elementos da lista
frutas[1] = "uva"

# Acessando elementos da lista pelo índice
print(frutas[1]) # Saída: "uva"
```



Listas x Arrays

- Tamanho dinâmico vs. tamanho fixo
- Tipos de dados

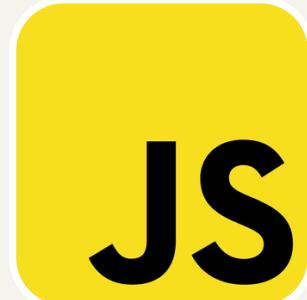
- **Lista:**

- Você pode adicionar ou remover elementos conforme necessário sem se preocupar com o tamanho inicial. E seus dados podem variar.

- **Array:**

- precisa especificar o tamanho quando o cria e não pode facilmente alterá-lo posteriormente.
Tipo de dados determinados e fixos.

Propriedades e Métodos da Lista:



- length

```
'length'

const numeros = [1, 2, 3, 4, 5];
console.log(numeros.length); // Saída: 5
```

- pop()

```
'pop()'

const frutas = ['maçã', 'banana', 'laranja'];
frutas.pop();
console.log(frutas); // Saída: ['maçã',
'banana']
```

- concat()

```
'concat()'

const frutas1 = ['maçã', 'banana'];
const frutas2 = ['laranja', 'uva'];
const todasAsFrutas = frutas1.concat(frutas2);
// todasAsFrutas é ['maçã', 'banana',
'laranja', 'uva']
```

- push()

```
'push()'

const frutas = ['maçã', 'banana', 'laranja'];
frutas.push('uva');
console.log(frutas); // Saída: ['maçã',
'banana', 'laranja', 'uva']
```

- splice()

```
'splice()'

const frutas = ['maçã', 'banana', 'laranja'];
frutas.splice(1, 1, 'uva'); // Remove a banana
e insere a uva no lugar
// frutas agora é ['maçã', 'uva', 'laranja']
```

Propriedades e Métodos da Lista:



- Descoberta de elementos
- indexOf()
- includes()

```
••• 'indexOf()  
  
const frutas = ['maçã', 'banana', 'laranja'];  
const indice = frutas.indexOf('banana');  
// indice é 1 (índice da 'banana' no array)
```

```
••• 'includes()  
  
const frutas = ['maçã', 'banana', 'laranja'];  
const temBanana = frutas.includes('banana');  
// temBanana é true
```

Propriedades e Métodos das Listas:



- len()

```
... len()  
  
minha_lista = [1, 2, 3, 4, 5]  
tamanho = len(minha_lista)  
print(tamanho) # Saída: 5
```

- pop()

```
... pop()  
  
minha_lista = [1, 2, 3, 4, 5]  
elemento_removido = minha_lista.pop()  
print(minha_lista) # Saída: [1, 2, 3, 4]  
print(elemento_removido) # Saída: 5
```

- + (concatenação)

```
... +  
  
lista1 = [1, 2, 3]  
lista2 = [4, 5, 6]  
resultado = lista1 + lista2  
print(resultado) # Saída: [1, 2, 3, 4, 5, 6]
```

- append()

```
... append()  
  
minha_lista = [1, 2, 3]  
minha_lista.append(4)  
print(minha_lista) # Saída: [1, 2, 3, 4]
```

- : (slicing)

```
... : (slicing)  
  
minha_lista = [1, 2, 3, 4, 5]  
parte_da_lista = minha_lista[1:4] # Obtém  
elementos de índice 1 a 3  
print(parte_da_lista) # Saída: [2, 3, 4]
```

Propriedades e Métodos das Listas



- Descoberta de elementos
- index()
- in

```
...  
index()  
  
minha_lista = [1, 2, 3, 4, 5]  
indice = minha_lista.index(3)  
print(indice) # Saída: 2 (índice do elemento  
3 na lista)
```

```
...  
in  
  
minha_lista = [1, 2, 3, 4, 5]  
elemento = 3  
if elemento in minha_lista:  
    print("Elemento encontrado na lista.")  
else:  
    print("Elemento não encontrado na lista.")
```

Revisão

while()

```
const frutas = ["Banana", "Uva", "Jaca", "Jenipapo"];
let acabar = prompt("Quer comprar? (Para acabar aperte 0)");

while (acabar != 0) {
    console.log("Vou comprar mais frutas")

    acabar = prompt("Quer continuar a compra? (Para acabar aperte 0)");
}
```

```
frutas = ["Banana", "Uva", "Jaca", "Jenipapo"]
acabar = input("Quer comprar? (Para acabar aperte 0)")

while acabar != 0:
    console.log("Vou comprar mais frutas")
    acabar = input("Quer continuar a compra? (Para acabar aperte 0)")
```

for()

```
1 const frutas = ["Banana", "Uva", "Jaca", "Jenipapo"];
2
3 for (let contador = 0; contador < frutas.length; contador++) {
4     console.log(frutas[contador])
5 }
```

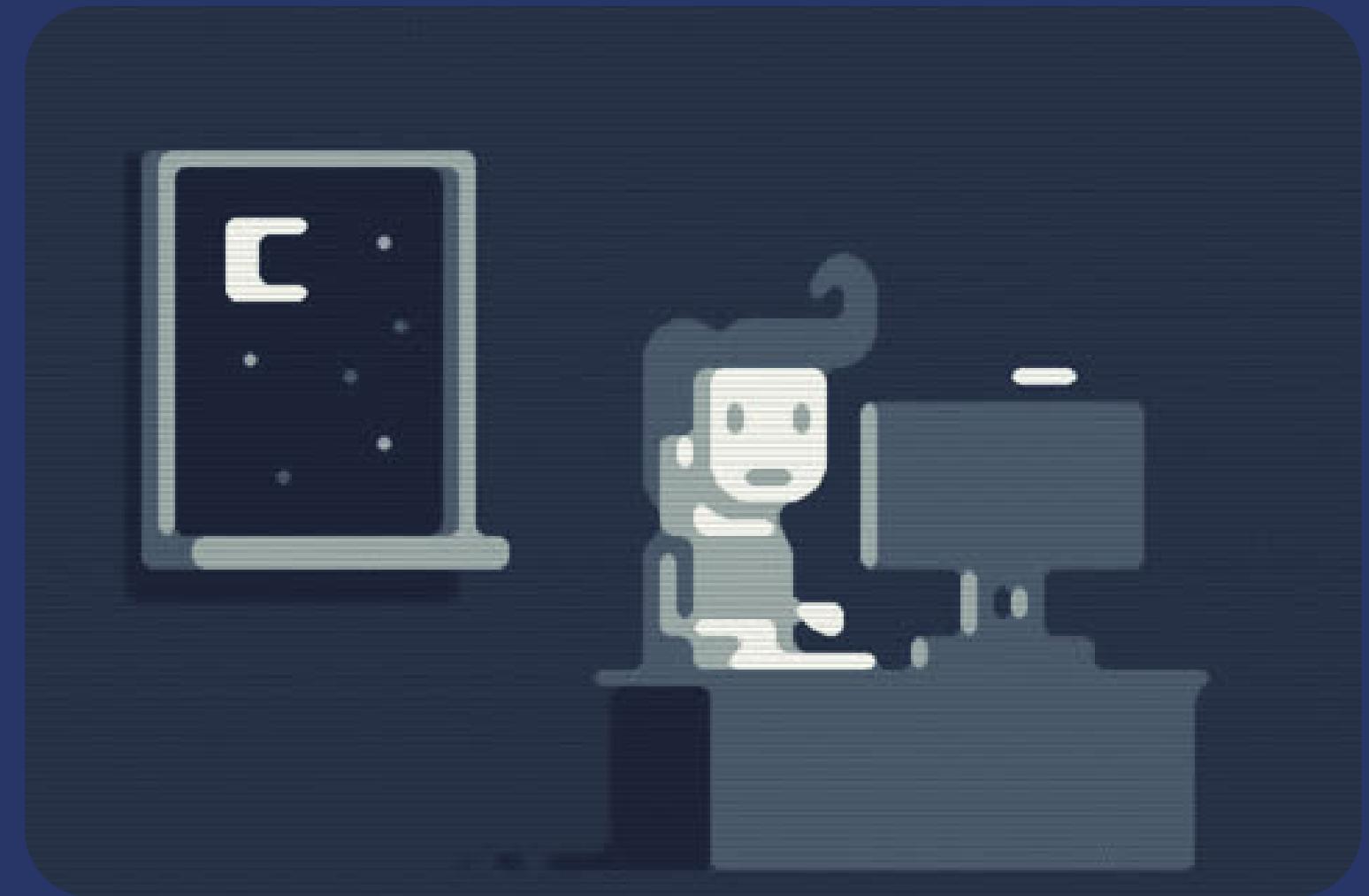
```
1
2 frutas = ["Banana", "Uva", "Jaca", "Jenipapo"]
3
4 for contador in range(len(frutas)):
5     print(frutas[contador])
```

Como funcionam?
Sua sintaxe
Exemplos

Exercícios

- Busca Simples no Vetor 01: [**Clique**](#)
- Média de um vetor: [**Clique**](#)
- Dois Vetores: Pares e Ímpares: [**Clique**](#)
- Substituição no Vetor: [**Clique**](#)
- Intruso: [**Clique**](#)
- Carrinho de compras: [**Clique**](#)

That's all folks!



Alagoas no Mapa da OBI