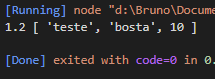
**Métodos e Atributos de Arrays**

**Lenght:** atributo que retorna o número de elementos dentro de um array, incluindo os elementos vazios;

**Ex.:**

**Métodos para adicionar e remover elementos**

**POP(index):** remove e retorna o último elemento;

******Ex.:**

**PUSH(valor1, valor2, ...):** adiciona um ou vários elementos no final do array.

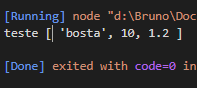
* Retorna o **comprimento do novo array;**
* A ordem que ficarão depende da ordem que foram passados como argumento;

**Ex.:**

l

**SHIFT():** remove o primeiro elemento de um array e ajeita os index dos outros elementos.

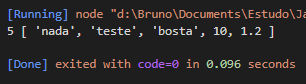
* Retorna o elemento que foi removido

**Ex.:**

****

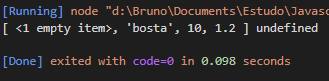
**UNSHIFT(valor1, valor2, valor3, ...):** adiciona um ou vários valores no começo do array

* Retorna o comprimento do novo array

**Ex.:**

**Delete array[index]:** remove o elemento naquele index.

* O elemento deletado será substituído por um **undefined;**
* Essa forma deixa buracos no array;

******Ex.:**

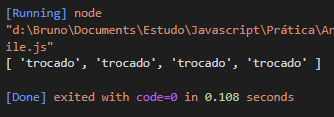
**Splice(where, delete, element1, element2, …):** é usado para adicionar ou remover elementos.

* **Where(obrigatório):** o index onde será adicionado/excluído o elemento
* **Delete(obrigatório):** quantos elementos serão deletados a partir do index especificado em where;
* **Element1, element2(opcional):** quais elementos serão adicionados
* Retorna um array com os elemento(s) deletado(s)

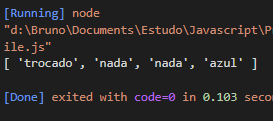
**Substituindo elementos dentro de um array**

**Fill(item, start, end):** substitui elementos dentro de um array

* **Item:** o item que queremos preencher o array;
* **Start(opcional):** index inicial. Valor padrão é 0;
* **End(opcional):** index final. Valor padrão é o final do array.

**Ex.:**



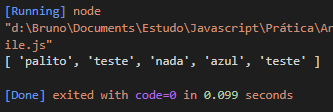


**copyWithin(target, start, end):** pega os elementos de um array é coloca em outra posição, substituindo o elemento naquela posição.

* **Target:** diz o index a partir do qual as cópias serão inseridas.
* **Start:** diz o index a partir do qual a cópia começa. O padrão é o começo do array.
* **End:** diz o index a partir do qual a cópia termina. O padrão é o final do array.

**Ex.:** 0 => diz que as cópias serão inseridas a apartir da posição 0; 2 => diz que a cópia começará da posição 2 até o final



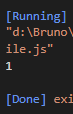
**Ex. 2:**

****

**Verificando elementos dentro de um array**

**indexOf(item, start):** retorna o index da primeira ocorrência do item que você informar.

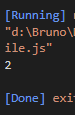
* **Item:** o item que queremos procurar;
* **Start:** diz por qual index começar.
* Se o elemento não for encontrado, **retorna -1**.

**Ex.:**



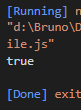
**lastIndexOf(item, start):** retorna o index da última ocorrência de um elemento

* **Item:** o item que queremos procurar;
* **Start:** diz por qual index começar.
* Se o elemento não for encontrado, **retorna -1**.

**Ex.:**

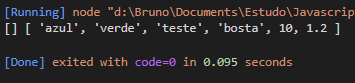


**Includes(item, start):** verifica se um determinando elemento está no array

* Item: o item que queremos verificar;
* Start: por qual index começarmos a procura;
* Se o item estiver no array, retorna **true**; caso contrário, retorna **false**.

**Ex.:**

**Adicionado elemento com splice:**

**Ex.:**

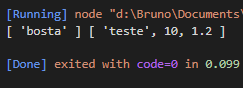
**Where = 0 =>** os elementos serão adicionados no index a apartir do index 0

**Delete = 0 =>** nenhum elemento será deletado a partir do index 0

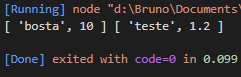
**‘azul’ e ‘verde’ =>** os elementos que serão adicionados a partir do index 0

**Removendo elemento com splice:**

* É recomendável usar splice para deletar elementos ao invés do delete porque ele(splice) não deixa **‘buracos’** no array original!

**Ex.:**



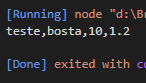
**Ex. 2:** removendo vários elementos de uma vez



**Método para transformar um Array em uma String**

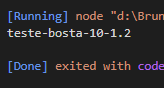
**toString():** transforma todos os elementos de um array em uma string;

* Ficam separados por vírgula;
* Não é possível especificar o separador.

**Ex.:**

**Join(separador):** faz o mesmo que o método toString(), mas:

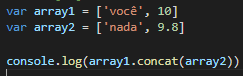
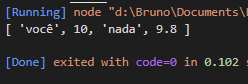
* É possível escolher o separador
* Se não for informado, o separador será a vírgula

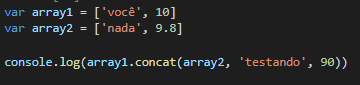
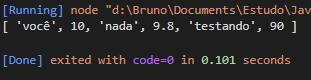
**Ex.:**

**Métodos para juntar Arrays**

**Concat(valor ou array):** é possível juntar os elementos de vários arrays ou strings, números, objetos etc

* Retorna um **novo array** com todos os elementos adicionados

**Ex.:** juntando dois arrays

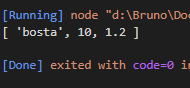
**Ex. 2**: juntando arrays, string e números

**Método para separar Arrays**

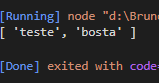
**Slice(start, end):** separa um array

* Retorna um novo array com os elementos separados;
* **Start:** o index inicial;
* **Lenght(opcional):** quantos elementos farão parte do novo array. Se não for informado, pega todos desde do start até o último.

**Ex.:** separando o array desde do elemento com index 1 até o final

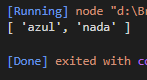


**Ex. 2:** separando desde do elemento com index 0 num total de 2 elementos.

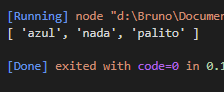


**Métodos para Organizar um Array**

**Sort():** organiza todos os elementos em **ordem alfabética**

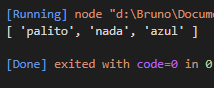
**Ex.:**

**Reverse():** reverte a ordem de todos os elementos em um array.

**Ex.:**

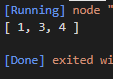


**Ex. 2:** se você quiser organizar em **ordem alfabética decrescente**, primeiro use o sort() depois o reverse()

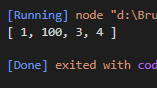


**Organizando números**

Naturalmente, o método sort() não organiza os números em ordem crescente, a não ser que seja um **número de apenas um algarismo**, aí não teríamos problemas.

**Ex.:**

**Problema:** Se o **número tiver mais de um algarismos**, teremos problemas porque esse método só analisa o primeiro algarismo de cada número

**Ex.:** perceba que o número 100 é o maior de todos e deveria ficar no final, mas ele começa com ‘1’, então seria considerado o segundo menor número depois do ‘1’