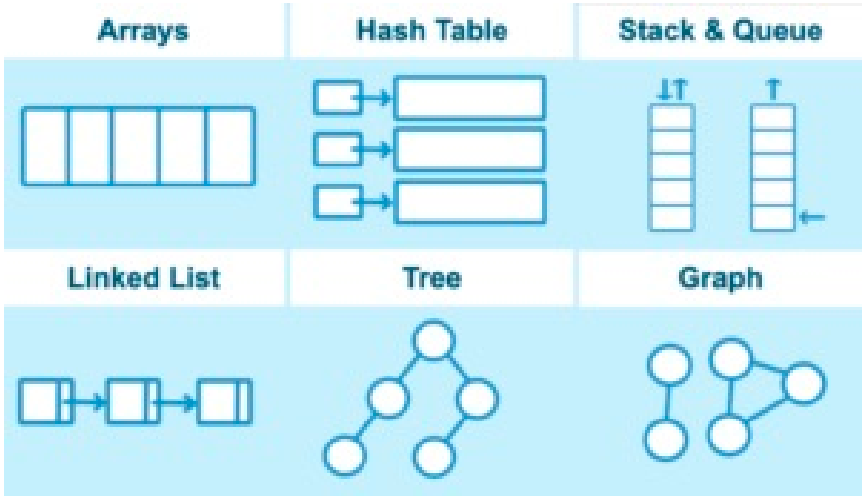
Data structures an Algorithms

Operations in an Array

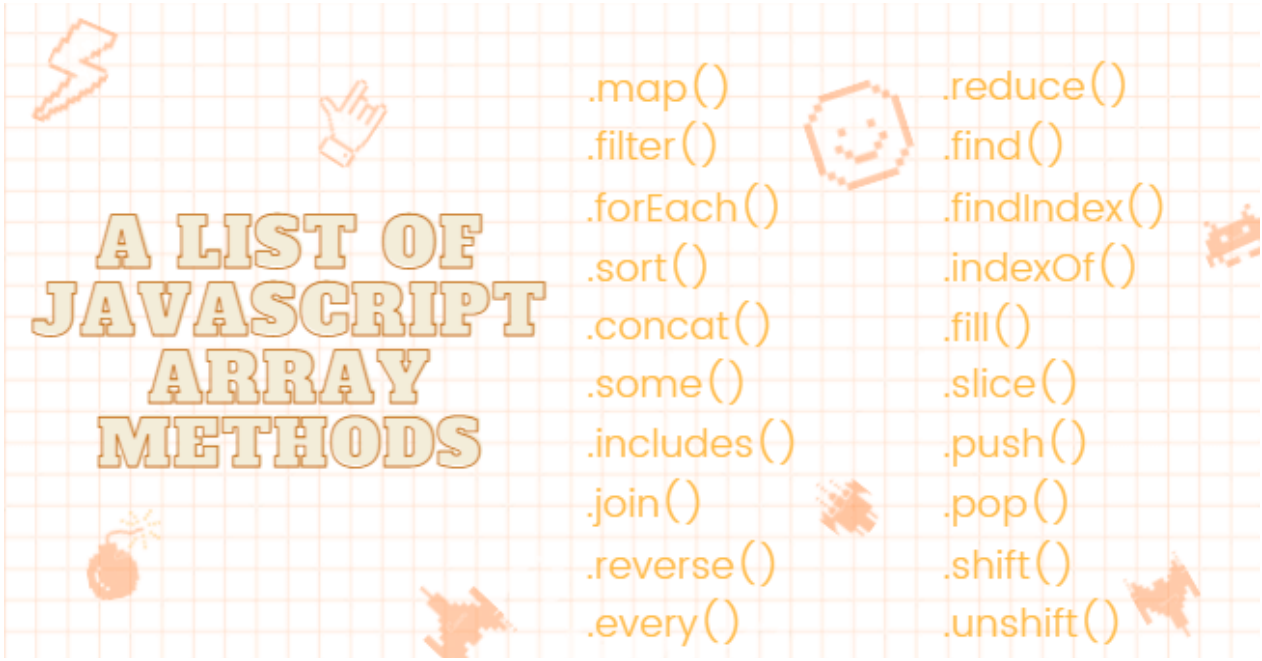
- Declaring, instantiating, initializing an Array
- insertion a value (push)
- Traversing a given value (for loop, for each map)
- Accessing a given cell (with index value)
- Searching a given value (find, filter, includes)
- deletion a given value (pop, shift,splice)

Uses of Arrays

- Create Hash Tables
- Create Stacks
- Create Queues



Data structures an Algorithms



array.sort()

sort() method is used to arrange the elements of the given array either in ascending or descending order, by default it arranges the elements in the ascending order.

```
let arr = [4, 2, 5, 1, 3]
let result = arr.sort()

console.log('Orginal array',arr) // "Orginal array", [1, 2, 3, 4, 5]
console.log('New return array',result) // "New return array", [1, 2, 3, 4, 5]

// you can also provide function expressions to determine the order
let arr1 = [6, -3, -10, 0, 2, 8]

// to sort in ascending order
arr1.sort((a,b) => a - b)

console.log(arr1) // [-10, -3, 0, 2, 6, 8]

// to sort in descending order
arr1.sort((a,b) => b - a)

console.log(arr1) // [8, 6, 2, 0, -3, -10]
```

Data structures an Algorithms

array.concat()

concat() method is used to merge two or more arrays and return a new array. It does not affect the original arrays and returns the new array of the sum of all the array lengths.

```
let arr = [1,2,3,4,5]
let arr1 = [6,7,8,9,10]
let arr2 = [11,12,13,14,15]

// merging two arrays
let result1 = arr.concat(arr1)
console.log(arr) // [1, 2, 3, 4, 5]
console.log(arr1) // [6, 7, 8, 9, 10]
console.log(result1) // [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]


//merging two and more arrays
let result2 = arr.concat(arr1,arr2)
console.log(arr) // [1, 2, 3, 4, 5]
console.log(arr1) // [6, 7, 8, 9, 10]
console.log(arr2) // [11, 12, 13, 14, 15]
console.log(result2) // [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]
```

array.some()

concat() method is used to merge two or more arrays and return a new array. It does not affect the original arrays and returns the new array of the sum of all the array lengths.

```
let arr = [2,5,7,4,6,1]

let result = arr.some(item => item > 8)
let result2 = arr.some(item => item % 3 == 0)

console.log(arr) // [2, 5, 7, 4, 6, 1]
console.log(result) // false
console.log(result2) // true
```

Data structures an Algorithms

array.includes()

includes() method is used to determine that a certain value is present among the elements in the given array or not and returns true or false accordingly. It also does not affect the original array.

```
let arr = [2,5,7,4,6,1]

let result = arr.includes(4)
let result2 = arr.includes(9)

console.log(arr) // [2, 5, 7, 4, 6, 1]
console.log(result) // true
console.log(result2) // false
```

array.join()

join() method is used to return a new string, after concatenating all the elements of the given array separated with a specified separator and It does not affect the original array.

```
let arr = ['A','p','p','l','e']
let result = arr.join() // giving no seprator
let result1 = arr.join('') // giving empty seprator

console.log(arr) // ["A", "p", "p", "l", "e"]
console.log(result) // "A,p,p,l,e"
console.log(result1) // "Apple"

let arr1 = ['This','is','so','awesome']

let result2 = arr1.join(' ') // giving space as a seprator
let result3 = arr1.join('-') // giving - as a seprator

console.log(arr1) // ["This", "is", "so", "awesome"]
console.log(result2) // "This is so awesome"
console.log(result3) // "This-is-so-awesome"
```

Data structures an Algorithms

array.includes()

includes() method is used to determine that a certain value is present among the elements in the given array or not and returns true or false accordingly. It also does not affect the original array.

```
let arr = [2,5,7,4,6,1]

let result = arr.includes(4)
let result2 = arr.includes(9)

console.log(arr) // [2, 5, 7, 4, 6, 1]
console.log(result) // true
console.log(result2) // false
```

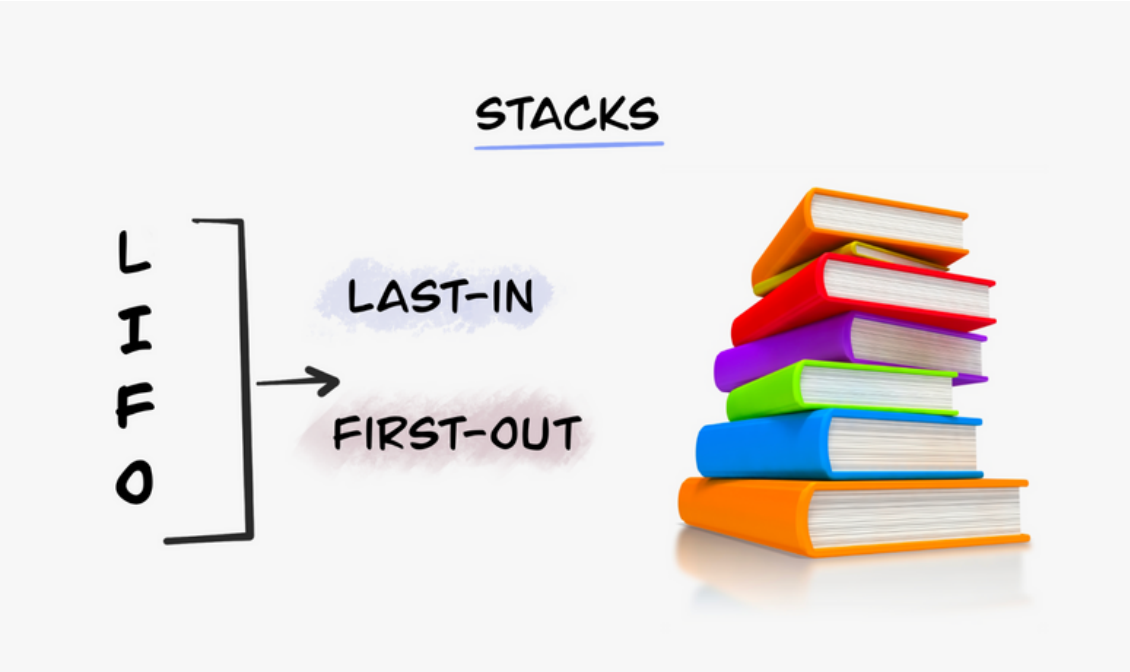
array.join()

join() method is used to return a new string, after concatenating all the elements of the given array separated with a specified separator and It does not affect the original array.

```
let arr = ['A','p','p','l','e']
let result = arr.join() // giving no seprator
let result1 = arr.join('') // giving empty seprator

console.log(arr) // ["A", "p", "p", "l", "e"]
console.log(result) // "A,p,p,l,e"
console.log(result1) // "Apple"

let arr1 = ['This','is','so','awesome']

let result2 = arr1.join(' ') // giving space as a seprator
let result3 = arr1.join('-') // giving - as a seprator

console.log(arr1) // ["This", "is", "so", "awesome"]
console.log(result2) // "This is so awesome"
console.log(result3) // "This-is-so-awesome"
```

Data structures an Algorithms

Stack



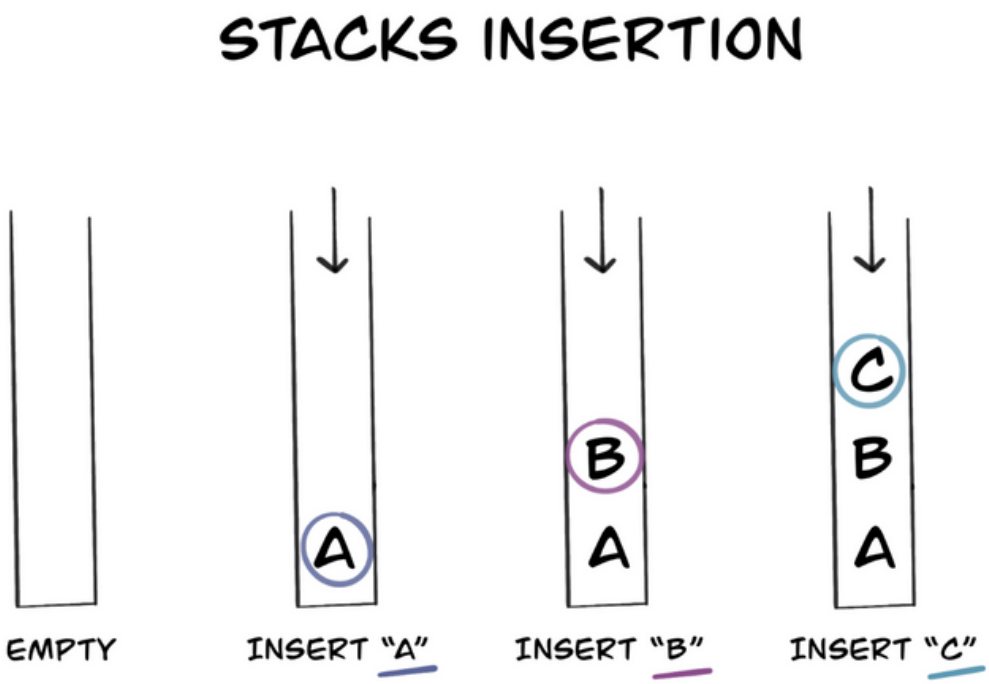
- Stack is a linear data structure.
- It follows LIFO: Last In First Out
- The addition/removal of elements can happen only at the one end.
- The element added in the last will be removed first.
- Real-world Examples:
    - Pile of dishes
    - Stack of books

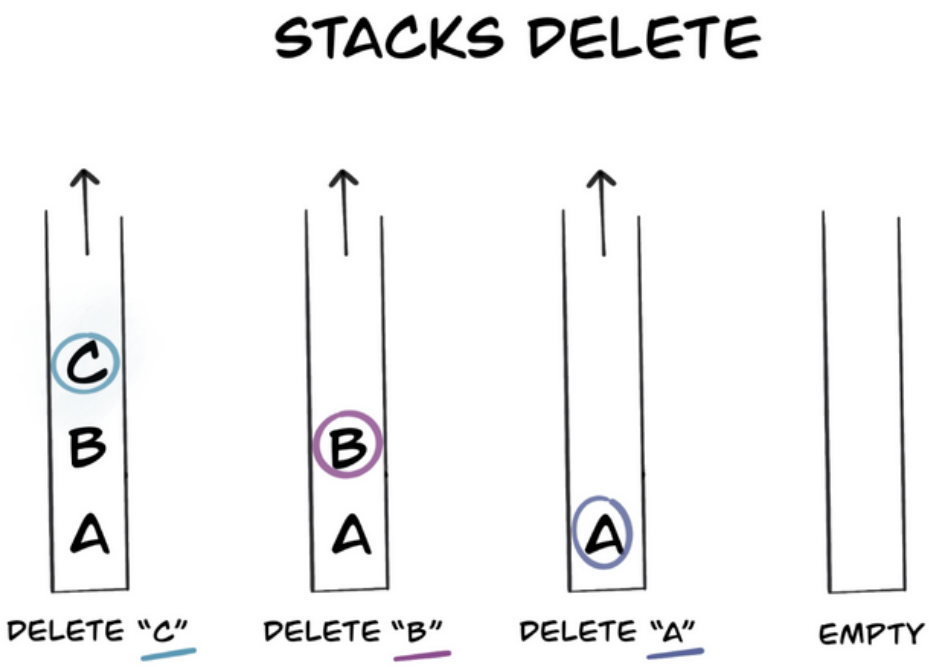
- Tech Examples:
    - Undo functionality: The last written element is removed first. Followed by second last and so on.

Data structures an Algorithms

Operations in Stacks

- Declaring, instantiating, initializing a stack.
- Push: Inserting a value to the top of the stack.
- Pop: Removing a value from the top of the stack.
- Peek: Viewing the topmost element.





KEEP LEARNING AND HAPPY CODING