JavaScript & Module ③ (Array)

__/__/____

① push() - Adds one or more elements to the end of an array & returns the new length of the array

```
let arr = [1, 2, 6];
arr.push (3, 4, 5);
console.log (arr);          // [1, 2, 6, 3, 4, 5]
```

② pop() - Removes the last element from an array & returns that element.

```
let arr = [1, 2, 6];
arr.pop ();
console.log (arr);          // [1, 2]
```

③ shift() - Removes the first element from an array and returns the element.

```
const arr = [1, 2, 3];
const arr2 = arr.shift();
console.log (arr 2);        // [2, 3]
Console.log (arr);          // [~~1, 2, 3~~]
```

④ unshift() - Adds one or more element to the beginning of an array and return the new length of an array.

```
const arr = [1, 2, 3];
arr.unshift (0, -1);
console.log (arr);          // [0, -1, 1, 2, 3]
```

⑤ find.js — Returns the value of first element in array that satisfies the provided testing function, otherwise undefined is return.

```
const numbers = [1, 2, 3, 4, 5];
const X = numbers.find((num) => num > 3);
console.log(X);          // 4
```

⑥ some() — Test whether at least one element in the array passes the test implemented by the fnc. It returns true if any element passes the test, otherwise false.

```
const numbers = [1, 2, 3, 4, 5];
const hasEven = numbers.some((num) => num%2==0);
console.log(hasEven);    // True
```

⑦ every() — Test whether all elements in the array pass the test implemented by the provided fnc. It returns true if all elements pass the test, otherwise false.

```
const numbers = [1, 2, 3, 4, 5];
const allEven = numbers.every((num) => num%2==0);
console.log(allEven);    // False
```

__/__/____

⑧ sort() - Sorts the element of an array in place & returns the sorted array. The default sort order is built upon converting the elements into strings, then comparing their sequences of UTF-16 code units values.

```
const num = [100, 20, 30, 10];
num.sort ((a, b) => a - b);
console.log (num);          // [10, 20, 30, 100]
```

⑨ includes() - determines whether an array includes a certain element, returning true or false as appropriate.

```
const num = [1, 2, 3];
const include3 = num.includes(3);
console.log (include3);          // true
```

⑩ Slice() - returns a shallow copy of a portion of an array into a new array object selected from start to end (end not included). The original will not be modified.

```
const num = [1, 2, 3, 4, 5];
const sliced = num.slice (0, 2);
console.log (sliced);          // [1, 3]
console.log (num);             // [1, 2, 3, 4, 5]
```

__/__/____

(11) map() - Creates a new array with the results of calling a provided function on every element in the calling array.

```
const numberss = [1, 2, 3];
const N = numbers.map((num) => num * 2);
console.log(N);          // [2, 4, 6]
```

(12) reduce() - executes a reducer function on each element of the array, resulting in a single output value.

```
const numbers = [1, 2, 3, 4, 5];
const sum = numbers.reduce((total, num) =>
    total + num, 0);
console.log(sum);        // 15
```

(13) filter() - creates a new array with all elements that pass the test implemented by the provided function.

```
const numbers = [1, 2, 3, 4, 5];
const hasEven = numbers.filter((num) => num % 2
    == 0);
console.log(hasEven);    // [2, 4]
```

(14) forEach() - executes a provided fnc once for each array element.

```
const numbers = [1, 2, 3];   numbers.forEach((num) =>
    console.log(num * 2));   // 2, 4, 6
```

__/__/____

(15) indexOf() - returns the 1st index at which a given element can be found in array, or -1 if it is not present.

```
const fruits = ['banana', 'apple', 'orange', 'grape'];
const appleIndex = fruits.indexOf('apple');
console.log(appleIndex);                    // 1
```

(16) lastIndexOf() - returns the last index at which a element can be found in the array, or -1 if it is not present

```
const fruits = ['apple', 'banana', 'grape'];
const bananaIndex1 = fruits.lastIndexOf('banana');
console.log(bananaIndex1);                  // 2
```

(17) reverse() - Reverses the order of element of an array in place. The 1st element becomes the last & vice-versa.

```
const numbers = [1, 2, 3];
numbers.reverse();
console.log(numbers);                       // [3, 2, 1]
```

(18) concat() - returns a new array that includes the elements from original array & additional elements

```
const A = [1, 2]; const B = [3, 4];
const C = A.concat(B);
console.log(C);                             // [1, 2, 3, 4]
```

__/__/__

(19) join()

- Joins all elements of an array into a string, the elements are seperated by a specified seperator string.

```
const Fruits = ['banana', 'apple', 'grape'];
const joined = Fruits.join(',');
console.log(joined); // 'banana,apple,grape'
```

(20) toString()

- returns a string representing that specified number or array & returns its element

```
const numbers = [1,2,3];
const string = numbers.toString();
console.log(string);          // '1,2,3'
```

'Challenges'

(A) Declare an array named "teaFlavours" that contains the string "green tea", "black tea", "oolong tea". Access the 1st element of the array and store it in a variable "FirstTea".

```
let teaFlavours = ["green tea", "black tea",
                   "oolong tea"];
let FirstTea = teaFlavours[0];
console.log(FirstTea);        // green tea
```

__/__/____

(B) Declare an array named "cities" containing "london", "Tokyo", "Paris", "New York". Access the 3rd element in the array & store it in the variable named "FavouriteCity".

```
let cities = ["london", "Tokyo", "Paris", "New York"];
let FavouriteCity = cities[2];
console.log(FavouriteCity);          // Paris
```

(C) You have an array named "teaTypes" containing "herbal tea", "white tea", "masala chai". Change the 2nd element of array to "Jasmine Tea"

```
let teaTypes = ["herbal tea", "white tea", "masala chai"]
teaTypes[1] = "Jasmine Tea";
console.log(teaTypes)
```

(D) Declare any array named "citiesVisited" containing "Mumbai" and "Sydney". Add "Berlin" to the array using the push method

```
let citiesVisited = ["Mumbai", "Sydney"];
citiesVisited.push("Berlin");
console.log(citiesVisited);          // ["Mumbai", "Sydney", "Berlin"];
```

[For more challenges check GIT REPO]