Array Methods

1) concat()

The concat() method of array is used to merge two or more arrays. It does not change the existing array.

```
eg:- const arr1 = ["Nandu"]
     const arr2 = ["vinod"]
     const name = arr1.concat(arr2)

        // output: ['Nandu', 'Vinod']
```

2) Every()

The every method executes a function for each array element. The every() method returns true if the function returns true for all elements. The every() method returns false if the function returns false for one element. The every() method does not execute the function for empty elements. It does not change original array.

eg: const ages = [25, 40, 35, 50]

```
Function checkAge (age) {
    return age>18
}

ages.every (checkAge)

// output : true
```

3) fill()

The fill() method overwrites the original array. start and end position can be specified. if not all elements will be filled.

```
eg: const fruit = ["apple", "mango", "orange"]

    fruit.fill ("grape")   => [grape , grape,

    fruit.fill ("grape", 2, 3)

    => ['apple', 'mango', 'grape']
```

4) Fi

4) Find()

The find() method return the value of the first element that passes a test. The find() method executes a function for each array element. The find() method return undefined if no elements are found. The find() method does not executes the function for empty elements. The find() method does not change the orginal array.

```
eg:- const age1 = [5,15,12,25,30]
     function findAge(age){
              return age>10
          }
          ~~age1~~ age1.find(findAge))

     // output : 15
```

5) FindIndex()

FindIndex() method executes a functi for each array elements. It return the index (position) of the first elem that posses a test. It return -1 if no match is found. It does not execute the function for empty arr elements. It does not change the orig array.

```
eg:- const age2 = [5,10,19,25]
        Function checkAge (age) {
              return age>18
        }
        age2.FindIndex(checkAge)

   // output : 2
```

6)

6) flat()

The flat() method concatenate sub-array elements.

eg:- const num1 = [[1,2],[3,4],[5,6]]

const num2 = num1.flat()

//output: [1,2,3,4,5,6]

7) includes()

The includes () method returns true if an array contains a specific value. It returns false if the value is not found. It is case sensitive.

eg:- const fruit = ['apple', 'orange', 'mango']

Fruit.includes('apple')

//output : true.

8) IndexOF()

The indexOf() method returns the first index (position) of a specific val ~~The indexOf~~ It returns -1 if the value is not found. It starts at a specified index and searches from left to right. By default the search starts at the first element and ends at the last. Negative start values counts from the last element.

```
1. const fruit = ['apple', 'mango', 'orange']

   ~~fruit include~~
   fruit.indexOf('apple')

   //output : 0
```

9) join ()

• join method return an array as a string. It does not change the original array. any seperator can be specified. The default is comma(,).

eg: const fruit = ['apple', 'orange', 'mango']

```
fruit.join()

// output : apple, orange, mango.
```

10) lastIndexOf ()

The lastIndexOf () method returns the last index of a specified value. It return -1 if the value is not found. It starts at a specific index and searches from right to left.

eg:- const fruit = ["apple", "orange", "apple"]

```
fruit. lastIndex of ("apple")

// output : 2
```

11) <u>pop()</u>

pop method removes the last elemen of an array. It changes the origin array : It return the removed elemen

const fruit = ["apple", "orange", "mango"]

fruit . pop()

// output : mango.

12) <u>push()</u>

Push method adds new items to th end of an array. It changes the length of the array. It return the new length.

eg: const fruit = ["apple", "orange", "mango"]
fruit.push("lemon")

// output : ["apple", "orange", "mango", "lemon
length = 4

13) Reverse()

The reverse () method reverse the order of the element in on array. It over-writes the original array.

```
eg:   Const Fruit = ["apple","oronge","mongo"]
              Fruit.reverse()

      //output : ["mongo","oronge","apple"]
```

14) unshift ()

The unshift () method adds new elements to the beginning of on array. It over-writes the original array.

```
eg: const fruit = ["apple", "oronge","mongo"]
            Fruit.unshift ("grape")

//output : ["grape","apple","oronge","mongo"]
```

15) shift ()

The shift () method removes th
first item of an array. It changes
the original array. It returns the shif
element.

eg: const fruit = ['apple', 'oronge', 'mong
Fruit.shift ()

//out put : ['orgonye', 'mongo']

16) slice ()

slice () method return selected element
in an array as a new array. It
select from a given start, up to a
(not inclusive) given end. It does not
chonge the original array.

eg:
const Fruit = ['Bonona', 'oronge', 'lemon', 'app
Fruit.slice (1, 3)
// output : ['oronge', 'lemon']

17) <u>some()</u>

The some() method checks if any array elements pass a test (provided a callback function). It executes the callback function once for each array element. It returns true (and stops) if the function return true for one of the array elements. It return false if the function return false for all of the array elements. It does not execute the function for empty array elements. It does not change the original array.

eg:
```
const ages = [12,18,16,25,30]
        ages.some(checkAge)
                function checkAge(age) {
                        return age>18
                }

//output : true.
```

~~every()~~

19) sort()

Sort() method sorts the elements an array. It sorts the elements as strings in alphabetical and ascending order. It overwrites the original array.

eg: const fruit = ["Banana", "orange", "lemon",

fruit.sort()

//output: ["apple", "banana", "lemon", "orange"]

20) splice()

splice() method adds and/or remove array elements. It overwrites the original array.

eg: const fruit = ["Banana", "orange", "Apple", "mango"]

fruit. splice (2, 0 ,"Lemon", "kiwi");

//output: [Banana, orange, lemon, kiwi , Apple,

21) toString()

The toString() method returns a string with array values separated by commas. It does not change the original array.

eg:- const Fruit = ['apple', 'orange', 'grape']
Fruit.toString()

// output: apple, orange, grape.

22). Filter()

To filter method creates a new array field with elements that pass a test provided by a function. It does not execute the function for empty element. It does not change the original array.

eg:- const ages = [15, 20, 40, 35, 12]
ages.filter (checkAdult)
function checkAdult (age) {
return age > 18
}

//output [20, 40, 35]

23) <u>reduce()</u>

The reduce() method executes a reduc
Function for array element. It retu
a single value the Function's accumul
result. It does not execute the Functi
For empty array elements. It does no
change the original array.

```
eg:- const numbers = [100, 50, 25]
         numbers.reduce(numReduce)

         Function numReduce (total, num){
//output : 25       return total - num
         }
```

24) <u>map()</u>

map() creates a new array from
calling a function for every arra
element. It does not execute fu
Function for empty elements. It doe
not changes the original array.

eg:- const numbers = [2,4,6,8]
number.map (numMult)

Function numMult (num){
return num * 10
}

//output : ~~50~~ 20, 40, 60, 80

25) <u>Foreach()</u>

Foreach method calls a function for each element in an array. It is not executed for empty elements.

eg: let tex = " ";
const fruit = ["apple, "orange", "mango"]
Fruit. ForEach (printFruit)
console. log (text)

Function printFruit (item, index){
return text += index+":"+ item