ARRAY METHODS

ARRAY.MAP(CALLBACK)

ARRAY.MAP(CALLBACK)

What does it do?

ARRAY.MAP(CALLBACK)

What does it do?

The map method will iterate through every item in the array that it is being called from.

ARRAY.MAP(CALLBACK)

What does it do?

The map method will iterate through every item in the array that it is being called from.

It will invoke the callback function for each round in the iteration.

ARRAY.MAP(CALLBACK)

What does it do?

The map method will iterate through every item in the array that it is being called from.

It will invoke the callback function for each round in the iteration.

It will return a new array. The elements in this new array will be whatever the callback function returns with each iteration.

ARRAY.MAP(CALLBACK)

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.map()
```

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.map( function() {
        …
})
```

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.map( function( element, index, array ) {
        …
})
```

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.map( function( element, index, array ) {
        …
})
```

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.map( function(element, index, array ) {
        …
})
```

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.map( function(element, index, array ) {
        …
})
```

1

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.map( function(element, index, array ) {
        …
})
```

2

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.map( function(element, index, array ) {
        …
})
```

3

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.map( function(element, index, array ) {
       …
})
```

4

ARRAY.MAP(CALLBACK)

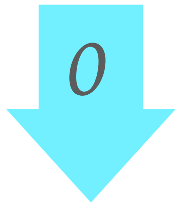
```
let arr = [1, 2, 3, 4, 5]

arr.map( function(element, index, array ) {
        …
})
```

5

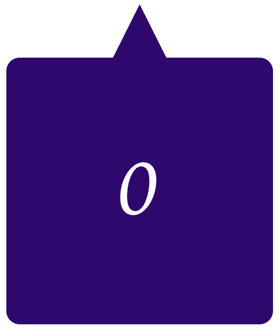
ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.map( function(element, index, array ) {
        …
})
```

ARRAY.MAP(CALLBACK)



```
let arr = [1, 2, 3, 4, 5]

arr.map( function( element, index, array ) {
        …
})
```



ARRAY.MAP(CALLBACK)

1

```
let arr = [1, 2, 3, 4, 5]

arr.map( function( element, index, array ) {
      …
})
```

1

ARRAY.MAP(CALLBACK)

2

```
let arr = [1, 2, 3, 4, 5]

arr.map( function( element, index, array ) {
        …
})
```

2

ARRAY.MAP(CALLBACK)

3

```
let arr = [1, 2, 3, 4, 5]

arr.map( function( element, index, array ) {
        …
})
```

3

ARRAY.MAP(CALLBACK)

4

```
let arr = [1, 2, 3, 4, 5]

arr.map( function( element, index, array ) {
        …
})
```

4

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.map( function(element, index, array ) {
        …
})
```

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.map( function( element, index, array ) {
        …
})
```

ARRAY.MAP(CALLBACK)

```
 let arr = [1, 2, 3, 4, 5]
arr.map( function( element, index, array ) {
        …
 })
```

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.map( function( element, index, array ) {
        …
})
```

arr

IN ACTION!

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]



 arr.map( function(element, index, array) {

 })
```

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]



let newArr = arr.map( function(element, index, array) {

             })
```

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

```
[                  ]
```

```
let newArr = arr.map( function(element, index, array) {
                          return element + 2
                     })
```

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

```
[              ]
```

```
let newArr = arr.map( function(element, index, array) {
                 return element + 2
             })
```

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

    [               ]

let newArr = arr.map( function(element, index, array) {
                  return element + 2
                })
```

1

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

[3]

```
let newArr = arr.map( function(element, index, array) {
                 return element + 2
               })
```

1

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

[3]

```
let newArr = arr.map( function(element, index, array) {
                        return element + 2
                    })
```

2

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

[3, 4]

```
let newArr = arr.map( function(element, index, array) {
                  return element + 2
              })
```

2

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

```
[ 3, 4           ]
```

```
let newArr = arr.map( function(element, index, array) {
                    return element + 2
                 })
```

3

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

```
[ 3, 4, 5        ]
```

```
let newArr = arr.map( function(element, index, array) {
                  return element + 2
              })
```

3

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

```
[ 3, 4, 5       ]
```

```
let newArr = arr.map( function(element, index, array) {
                 return element + 2
              })
```

4

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

```
[ 3, 4, 5, 6    ]
```

```
let newArr = arr.map( function(element, index, array) {
                    return element + 2
                })
```

4

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

```
[ 3, 4, 5, 6    ]
```

```
let newArr = arr.map( function(element, index, array) {
                      return element + 2
                  })
```

5

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

    [ 3, 4, 5, 6, 7 ]

let newArr = arr.map( function(element, index, array) {
                  return element + 2
                })

                                  5
```

ARRAY.MAP(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

```
[ 3, 4, 5, 6, 7 ]
```

```
let newArr = arr.map( function(element, index, array) {
                return element + 2
            })
```

ARRAY.FILTER(CALLBACK)

ARRAY.FILTER(CALLBACK)

What does it do?

ARRAY.FILTER(CALLBACK)

What does it do?

The filter method will iterate through every item in the array that it is being called from.

ARRAY.FILTER(CALLBACK)

What does it do?

The filter method will iterate through every item in the array that it is being called from.

It will invoke the callback function for each round in the iteration.

ARRAY.FILTER(CALLBACK)

What does it do?

The filter method will iterate through every item in the array that it is being called from.

It will invoke the callback function for each round in the iteration.

It will return a new array. The callback method will return either ***truthy*** *or* ***falsy****.*

If the return value is truthy, the current element will be included in the new array. If falsy is returned, the current element will not be included in the new array.

ARRAY.FILTER(CALLBACK)

ARRAY.FILTER(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

ARRAY.FILTER(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.filter()
```

ARRAY.FILTER(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.filter( function() {
        …
})
```

ARRAY.FILTER(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.filter( function(element, index, array) {
        …
})
```

IN ACTION!

ARRAY.FILTER(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]




 arr.filter( function(element, index, array) {

 })
```

ARRAY.FILTER(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]



let newArr = arr.filter( function(element, index, array)

                })
```

ARRAY.FILTER(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

    [     ]

let newArr = arr.filter( function(element, index, array)
                  return element > 3
              })
```

ARRAY.FILTER(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

[    ]

let newArr = arr.filter( function(element, index, array)
                              return element > 3
                         })
```

ARRAY.FILTER(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

[ ]

```
let newArr = arr.filter( function(element, index, array)
                 return element > 3
             })
```

False

ARRAY.FILTER(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

[   ]

```
let newArr = arr.filter( function(element, index, array)
                return element > 3
            })
```

False

ARRAY.FILTER(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

  [     ]

let newArr = arr.filter( function(element, index, array)
                      return element > 3
                  })
```

False

ARRAY.FILTER(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

```
[     ]
```

```
let newArr = arr.filter( function(element, index, array)
                 return element > 3
              })
```

True

ARRAY.FILTER(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

[4   ]

```
let newArr = arr.filter( function(element, index, array)
                    return element > 3
                })
```

True

ARRAY.FILTER(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

[4]

```
let newArr = arr.filter( function(element, index, array)
                   return element > 3
               })
```

True

ARRAY.FILTER(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

[4, 5]

```
let newArr = arr.filter( function(element, index, array)
                      return element > 3
                 })
```

True

ARRAY.FILTER(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

     [4, 5]

let newArr = arr.filter( function(element, index, array)
                   return element > 3
                })
```

ARRAY.REDUCE(CALLBACK)

ARRAY.REDUCE(CALLBACK)

What does it do?

ARRAY.REDUCE(CALLBACK)

What does it do?

The reduce method will iterate through every item in the array that it is being called from.

ARRAY.REDUCE(CALLBACK)

What does it do?

The reduce method will iterate through every item in the array that it is being called from.

It will invoke the callback function for each round in the iteration.

ARRAY.REDUCE(CALLBACK)

What does it do?

The reduce method will iterate through every item in the array that it is being called from.

It will invoke the callback function for each round in the iteration.

It will return a single value which is the resulting value of the total parameter.

ARRAY.REDUCE(CALLBACK)

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.filter()
```

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.filter( function() {
        …
})
```

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]

arr.filter( function(total, element, index, array ) {
        …
})
```

IN ACTION!

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]




 arr.reduce( function(total, element, index, array) {

 })
```

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]



   Let sum =  arr.reduce( function(total, element, index,

              })
```

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]



   Let sum =  arr.reduce( function(total, element, index,
                    return total + element
               })
```

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]



   Let sum =  arr.reduce( function(total, element, index,
                  return total + element
              })
```

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

```
Let sum =  arr.reduce( function(total, element, index,
                return total + element
           })
```

0

1

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

1

```
Let sum =  arr.reduce( function(total, element, index,
                  return total + element
               })
```

0

1

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

1

```
   Let sum =  arr.reduce( function(total, element, index,
                    return total + element
              })
```

1

2

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

3

```
Let sum =  arr.reduce( function(total, element, index,
                     return total + element
               })
```

1

2

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

3

```
   Let sum =  arr.reduce( function(total, element, index,
                     return total + element
              })
```

3

3

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

6

```
   Let sum =  arr.reduce( function(total, element, index,
                  return total + element
              })
```

3

3

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

6

```
Let sum =  arr.reduce( function(total, element, index,
                 return total + element
           })
```

6

4

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

10

```
  Let sum =  arr.reduce( function(total, element, index,
                   return total + element
                })
```

6

4

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

10

```
  Let sum =  arr.reduce( function(total, element, index,
                 return total + element
             })
```

10

5

ARRAY.REDUCE(CALLBACK)

```
let arr = [1, 2, 3, 4, 5]
```

15

```
  Let sum =  arr.reduce( function(total, element, index,
                 return total + element
             })
```

10

5