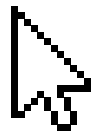
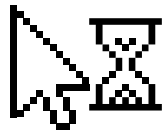
Computer Science :
JavaScript



Advanced 2: Iterators

Iterators

Iterators are methods that use callback functions as inputs and are called on for arrays to manipulate them.



.forEach()

.forEach(): will execute the same code for each element in the list.

Ex:

```
const groceries = [‘apples’, ‘milk’, ‘onions’];

groceries.forEach(function(items){
  console.log(‘I need’ + items)} )
// Prints: I need apples I need milk I need onions
```

.map()

.map(): Returns a new array based on the block code.

Ex:

```
const numbers = [1, 5, 8, 3, 18];

const newNum = numbers.map(num => {
  return num * 5 })

console.log(newNum) //Prints: [5, 25, 40, 15, 90]
```

.filter()

.filter(): It takes in a true or false function, and returns a new array and filters out certain elements that do not meet the conditions.

Ex:

```
const numbersList = [1, 6, 8, 2, 17];

const newList = numbersList.filter(num => {
  return num > 4 })

console.log(newList) //Prints: [6, 8, 17]
```

.findIndex()

.findIndex(): Provides the location of an element in an array. It will return the first element that evaluates true.

Ex:

```
const numberBowl = [3, 7, 36, 15, 4];

const newNumbers = numberBowl.findIndex(num => {
  return num > 7 })

console.log(newNumbers)//Prints: 2
```

.reduce()

.reduce(): Reduces the array to a single number based on the block code.

Ex:

```
const numList = [1, 2, 4, 10];

const sumList = numList.reduce(
(accumulator, currentValue) => {
  return accumulator + currentValue }, 100)
 //100 is the second argument so currentValue
console.log(sumList)//Prints: 117
```