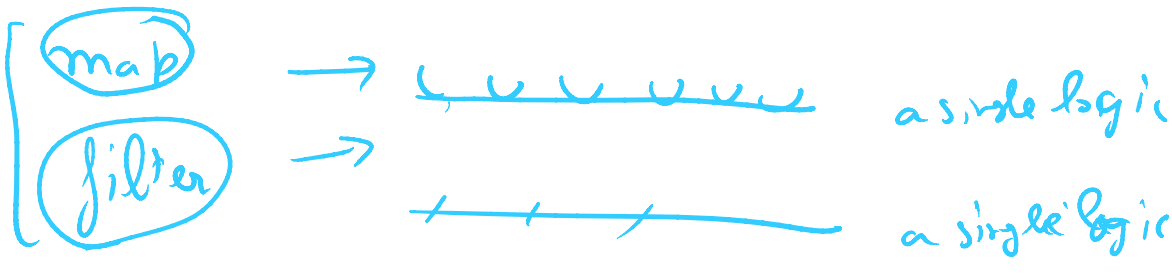
Array Methods

14 September 2022 20:04



for each

forloop

Arrays

arr.forEach(function (ele, index, arr) {

B.L

})

for (let i = 0 ; i < arr.length ; i++) {

i; arr[i] , arr

B.L → break

}

arr.length

arr.forEach(function (ele, index, arr) {

B.L

})

3)

```
for (let i=0; i<arr.length; i++) {
  arr[i], i, arr
  break
}
```

for loop → break

1, 2, 3 | →

If → false

1 2 3 4 5 6

Reduce

Write a for loop to sum an Array.

[1, 2, 3, 4] → 10

Reduce

↓

method

Arrays

. reduce (function (P.N, N.E) {

3, initial Num)

[1, 2, 3, 4]

. reduce (function , initial Element)

↓

function (P.E, C.E, (I) {

return C.E

}

[1, 2, 3, 4]

4

3 3

1 2

0

Sum 1

← reduce (function (P.E, C.E) {

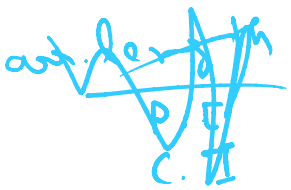
return sum + C.E

10

return sum + C.E

1
3
6
10

3, 0)

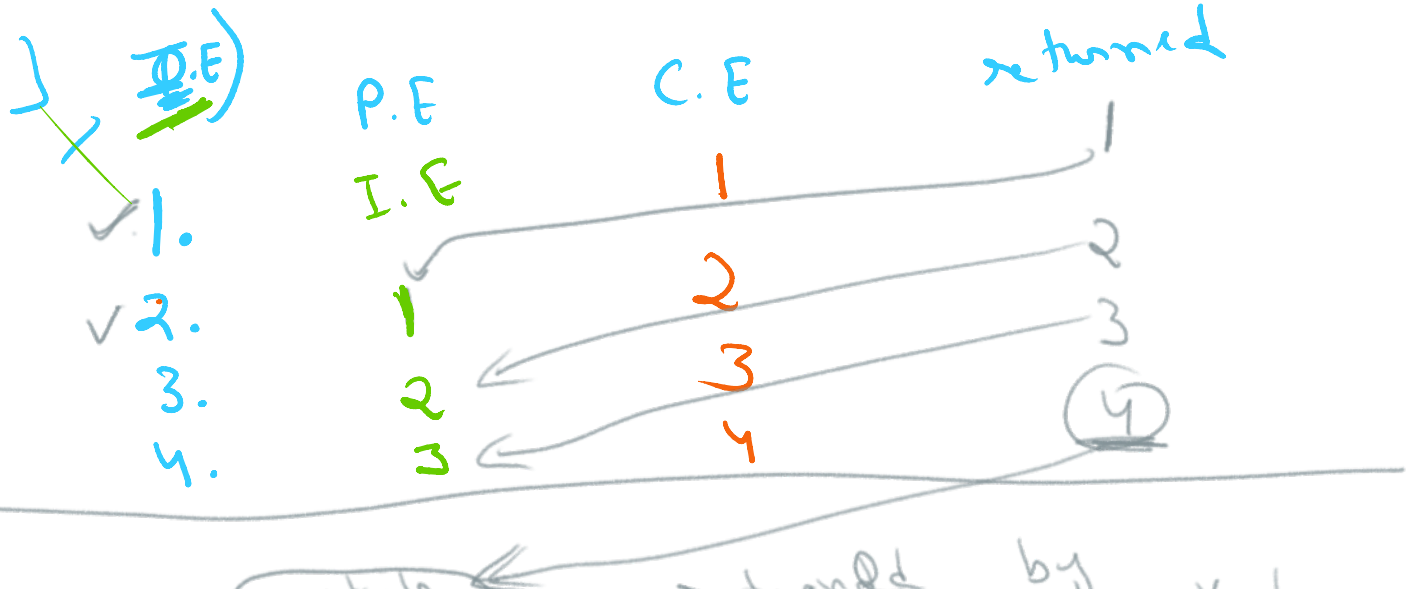


map → Array
filter → Array
forEach → nothing
reduce → a single element

[1, 2, 3, 4] 4

arr.reduce (function (P.E, C.E) {

return C.E



Outside reduce — returned by reduce method

```
arr.reduce ( function( P.E , C.E) {
    return P.E + C.E
```

1st time of execution

1) 'lakh'

2)

P.E , CE , returned

'lakh'

```
arr. reduce ( function (P.E, C.E) {
    return P.E + C.E
```

0 + 1
1 + 2
3 + 3
6 + 4

[1,2,3,4] 3, 0)

	P.E	C.E	Returned
1.	0	1	
	1	2	3
2.			6
3.	3	3	10
4.	6	4	

reduce

[1, 2, 3, 4] → 1 × 2 × 3 × 4

reduce

[1, 2, 3, 4] → (1+2) × 3 × 4

[1, 2, 3, 4]

.reduce(function (P.E, C.E) {

return P.E + C.E

1 + 2

3 + 3

6 + 4

})

	P.E	C.E	returned
1)			
2)	1	2	3
3)	3	3	6
4)	6	4	10

arr.reduce (function (P.E, C.E){

[1, 2, 3, 4]

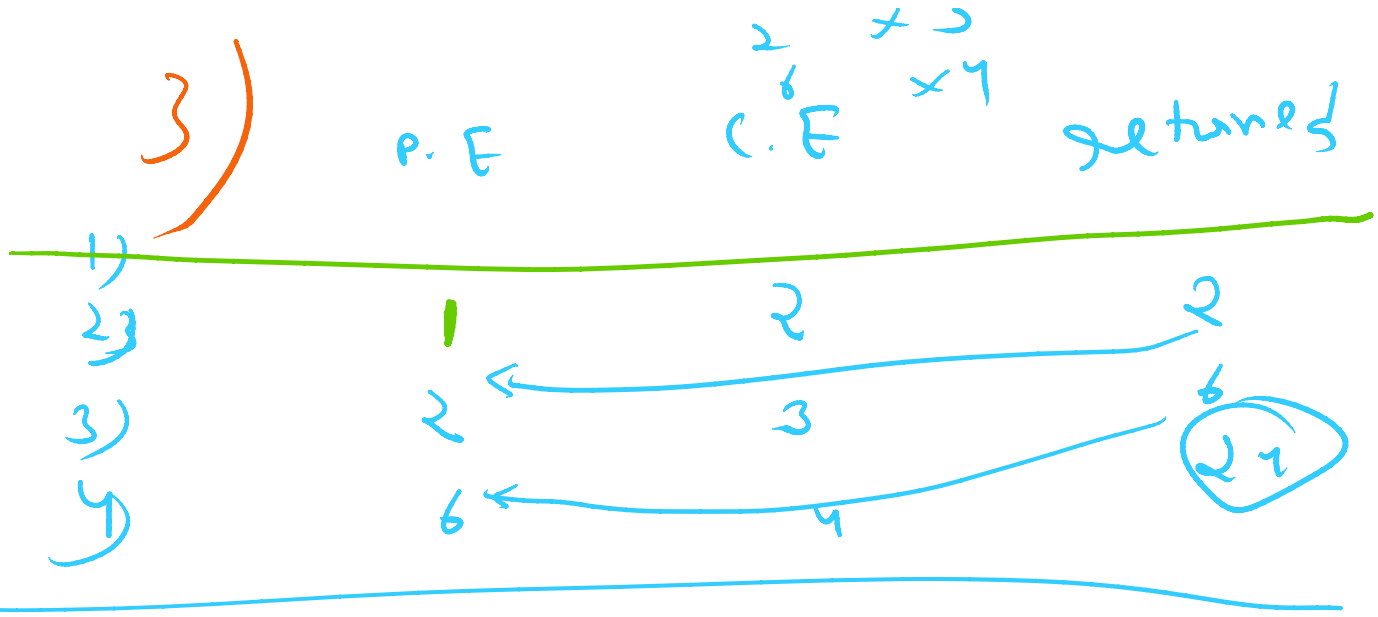
return P.E × C.E

1 × 2

2 × 3

6 × 4

> 1



2 types of for loop

for (let i of arr)

for (let i in arr)