```
//REDUCE():
//O método "reduce()" - reduzir - é mais um método do array que utilizamos quando desejamos operar o valor de um índice do array pelo próximo valor, o resultado é acumulado e depois é operado sobre o valor seguinte, o resultado é acumulado e depois é operado sobre o valor seguinte, e isso se segue até o que array termine.
//No final, reduce() não irá retornar um array, ele vai retornar um valor de acordo com o tipo de dado que operamos sobre o reduce()...
//Para fazer esse tipo de operação o reduce() recebe 2 parâmetros, onde o 1º é uma função callback que vai receber 2 parâmetros, onde o primeiro é acumulador e o segundo é o valor atual da iteração que será operado sobre o acumulador e o 2º parâmetro de reduce() é um valor inicial á nossa escolha, esse 2º parâmetro é opcional, se não passarmos ele, o reduce simplesmente vai iniciar á partir do índice 0 do array...
//Vejamos como utilizar o reduce:

//USANDO REDUCE PARA FAZER UM FATORIAL:
const nums = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] //Vamos usar esse array para fatorar o número 10...
const fatorial = nums.reduce(function(acumulador, atual) { //perceba que dessa vez não colocamos o 2º parâmetro de reduce, então ele vai calcular o array á partir do primeiro índice até o final...
    console.log(`Valor acumulado: ${acumulador}; Iterável atual: ${atual}`)
    return acumulador * atual //Perceba que pegarmos o valor acumulado e ele é calculado pelo próximo atual, gerando o fatorial...
})

console.log('1)')
console.log(`O fatorial de ${nums[nums.length - 1]} é`,  fatorial)
console.log(typeof fatorial) //No final ele não retorna um array, mas sim um valor de acordo com o tipo que foi operado...



//USANDO O 2º PARÂMETRO DE REDUCE() PARA GERAR UM VALOR INICIAL...
const fatora = (acumulador, atual) => {
    console.log(`Valor acumulado: ${acumulador}; Iterável atual: ${atual}`)
    return acumulador * atual
} //Criamos uma callback para não ter que criá-la literalmente no reduce()

console.log('\n2)')
const fatorialComSegundoParam = nums.reduce(fatora, 0) //Perceba que o reduce() agora além de receber a callback também recebeu o parâmetro 0 (zero), o que significa que antes de calcular pelo valor 1, que era o inícial, ele calculou por 0...
```

```
console.log(`O fatorial de ${nums[nums.length - 1]} é`, fatorialComSegundoParam) //Como todo número multiplicado por 0 é 0, o
fatorial acabou gerando 0...



//USANDO REDUCE() PARA RETORNAR VALORES DE UM ARRAY:
const alunos = [ //Queremos saber nessa lista se todo mundo é bolsista e também, se houver bolsistas quantos são...
    {nome: 'João', nota: 7.3, bolsista: false},
    {nome: 'Maria', nota: 9.2, bolsista: true},
    {nome: 'Pedro', nota: 9.8, bolsista: false},
    {nome: 'Ana', nota: 8.7, bolsista: true}
]

const valoresDasBolsas = a => a.bolsista //Função que mostra todos os valores dos bolsistas...
const alguemNaoEhBolsista = (acumulador, atual) => acumulador && atual //Se algum aluno for false o resultado retorna false...
const soBolsistas = (acumulador, atual) => acumulador + atual //Função que verifica quantos bolsistas temos...

const todosSaoBolsistas = alunos.map(valoresDasBolsas).reduce(alguemNaoEhBolsista) //Aqui a função map() nos traz um array com
valores true e false, e o reduce() faz a comparação por através de &&, se alguém for false o valor retornado é false...
const quantosSaoBolsistas = alunos.map(valoresDasBolsas).reduce(soBolsistas) //Aqui a função map() nos traz um array com valores true
e false, e o reduce() faz a soma de valores true, lembrando que true equivale a "1" e false a "0"...

console.log(`\n3)
Todos os alunos são bolsistas?: ${todosSaoBolsistas}`)
console.log(`Quantos alunos não são bolsistas: ${quantosSaoBolsistas}`)
```

RESULTADO NO CONSOLE...

```
[Running] node "c:\Users\Almoxarifado\Documents\javascript\arquivos_das_aulas\112-
reduce()__Gerando_Novos_Tipos_a_Partir_de_um_Array.js"
Valor acumulado: 1; Iterável atual: 2
Valor acumulado: 2; Iterável atual: 3
Valor acumulado: 6; Iterável atual: 4
Valor acumulado: 24; Iterável atual: 5
Valor acumulado: 120; Iterável atual: 6
Valor acumulado: 720; Iterável atual: 7
```

```
Valor acumulado: 5040; Iterável atual: 8
Valor acumulado: 40320; Iterável atual: 9
Valor acumulado: 362880; Iterável atual: 10
1)
O fatorial de 10 é 3628800
number

2)
Valor acumulado: 0; Iterável atual: 1
Valor acumulado: 0; Iterável atual: 2
Valor acumulado: 0; Iterável atual: 3
Valor acumulado: 0; Iterável atual: 4
Valor acumulado: 0; Iterável atual: 5
Valor acumulado: 0; Iterável atual: 6
Valor acumulado: 0; Iterável atual: 7
Valor acumulado: 0; Iterável atual: 8
Valor acumulado: 0; Iterável atual: 9
Valor acumulado: 0; Iterável atual: 10
O fatorial de 10 é 0

3)
Todos os alunos são bolsistas?: false
Quantos alunos não são bolsistas: 2

[Done] exited with code=0 in 0.112 seconds
```