

# Exercícios: .entries() + Destructuring em JavaScript

---

Exercícios voltados para fixar o uso de `entries()` com `for...of` e destructuring.

## Exercício 1

Dado o array const colors = ['red', 'green', 'blue'], use for...of com .entries() e destructuring para imprimir o índice e o nome da cor (ex: 1: red).

Solução:

```
const colors = ['red', 'green', 'blue'];
for (const [i, color] of colors.entries()) {
  console.log(`${i + 1}: ${color}`);
}
```

## Exercício 2

Dado o array const fruits = ['apple', 'banana', 'cherry'], imprima apenas os itens em maiúsculas com seus índices (ex: 1: APPLE).

Solução:

```
const fruits = ['apple', 'banana', 'cherry'];
for (const [i, fruit] of fruits.entries()) {
  console.log(`${i + 1}: ${fruit.toUpperCase()}`);
}
```

## Exercício 3

Com o array const langs = ['JS', 'Python', 'C++'], use destructuring para gerar uma nova array com strings formatadas assim: ['1: JS', '2: Python', '3: C++'].

Solução:

```
const langs = ['JS', 'Python', 'C++'];
const formatted = [];
for (const [i, lang] of langs.entries()) {
  formatted.push(`${i + 1}: ${lang}`);
}
console.log(formatted);
```

## Exercício 4

Dado const items = ['pen', 'notebook', 'eraser'], crie uma função que recebe esse array e retorna um array de objetos com {index: n, value: 'item'}

Solução:

```
function mapItems(arr) {
  const result = [];
  for (const [i, val] of arr.entries()) {
    result.push({ index: i, value: val });
  }
  return result;
}
console.log(mapItems(['pen', 'notebook', 'eraser']));
```

## Exercício 5

Com const tech = ['HTML', 'CSS', 'JS'], use destructuring com entries para imprimir somente os itens com índice ímpar.

Solução:

```
const tech = ['HTML', 'CSS', 'JS'];
for (const [i, el] of tech.entries()) {
  if (i % 2 !== 0) console.log(`${i}: ${el}`);
}
```

## Exercício 6

Dado const grades = [10, 6, 8], use entries + destructuring para calcular a média ponderada dos índices como pesos.

Solução:

```
const grades = [10, 6, 8];
let sum = 0, weight = 0;
for (const [i, grade] of grades.entries()) {
  sum += grade * (i + 1);
  weight += (i + 1);
}
console.log(sum / weight);
```

### **Exercício 7**

Dado const prices = [5, 10, 15], use entries + destructuring para aplicar um desconto de 10% aos índices pares.

Solução:

```
const prices = [5, 10, 15];
for (const [i, price] of prices.entries()) {
  const finalPrice = i % 2 === 0 ? price * 0.9 : price;
  console.log(`Item ${i}: ${finalPrice}`);
}
```

### **Exercício 8**

Crie uma função que imprime todos os elementos de um array com seu índice, mas começa a contagem no 100.

Solução:

```
function printFrom100(arr) {
  for (const [i, el] of arr.entries()) {
    console.log(`${i + 100}: ${el}`);
  }
}
printFrom100(['a', 'b', 'c']);
```

### **Exercício 9**

Use destructuring com entries para transformar ['x', 'y', 'z'] em objeto {0: 'x', 1: 'y', 2: 'z'}

Solução:

```
const letters = ['x', 'y', 'z'];
const obj = {};
for (const [i, el] of letters.entries()) {
  obj[i] = el;
}
console.log(obj);
```

### **Exercício 10**

Dado const tasks = ['code', 'test', 'deploy'], imprima 'Step 1: code', 'Step 2: test' etc.

Solução:

```
const tasks = ['code', 'test', 'deploy'];
for (const [i, task] of tasks.entries()) {
  console.log(`Step ${i + 1}: ${task}`);
}
```