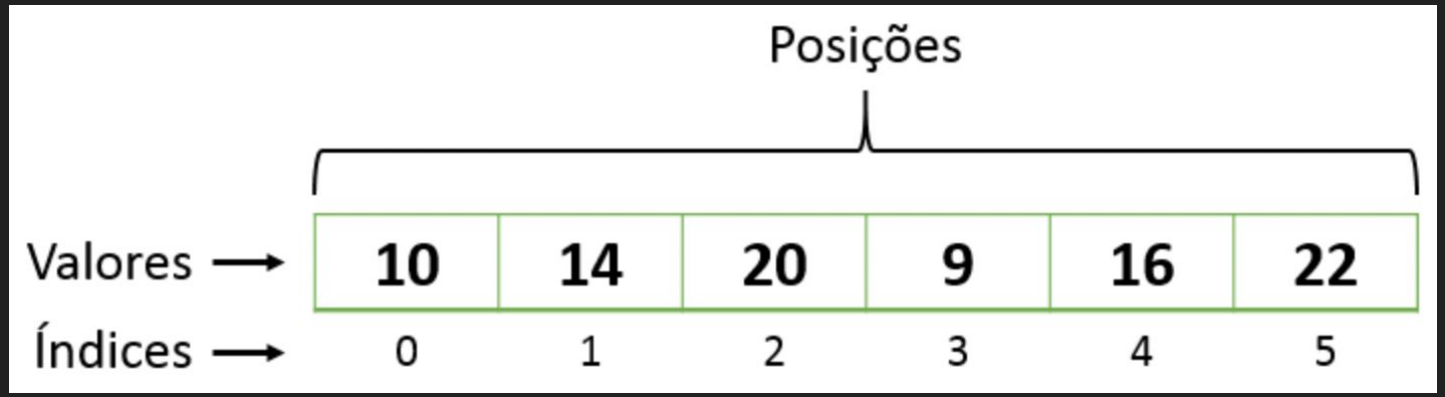
ROGÉRIO SILVA
@rogerio410
Desenvolvedor de
Software & Professor

Coleções

Vetores e Matrizes (Listas, Arrays, etc..)

Atualizado em 29/maio/2023

Organização de Vários valores e uma mesma estrutura de dados: indexadas ou chaveadas



Uma só **"variável"**
apontar para vários **"valores"**
por meio de **"índices"**

lista[index]

ou seja,

lista = [1, 5, 6, 19]

lista[0] é o primeiro valor → 1

Vetores são mutáveis (geralmente)

Leitura e Escrita

Operações com Vetores

criação, inicialização, fatias(slice), preenchimento, remoção(splice)

Criar Vetor

```
1
2   // Criar vetor a partir de dados
3   const carros = ['Argo', 'Gol', 'Onix']
4
5   // Criar um Vetor vazio com 10 posições (undefined)
6   const lista = new Array(10)
7
8   // Criar vetor com um elemento
9   const valores = [10]
10
11  // Criar um vetor a partir de outro vetor (copiar)
12  const dados = Array.from([2, 4, 5])
13
14  // Criar um vetor a partir de outro vetor (alterando)
15  const quadrados = Array.from(dados, (vlr, idx) => vlr*2)
16
17  // Criar um vetor a partir do "nada", e gerar seus valores
18  const numeros = Array.from({length: 5}, (valor, index) => index)
19
```

Saídas

```
❯ node criar_vetores.js
[ 'Argo', 'Gol', 'Onix' ]
[ <10 empty items> ]
[ 10 ]
[ 2, 4, 5 ]
[ 4, 8, 10 ]
[ 0, 1, 2, 3, 4 ]
```

Tip: Crie suas funções para Criar Vetor

Percorrendo uma Vetores

while, for, for in, for of, foreach, ...

```js
const carros = ['Argo', 'Gol', 'Onix']

    // Tradicional FOR (mais importante para vc)
    for (let i = 0; i < carros.length; i++){
        console.log('Posicao', i, 'Carro', carros[i])
    }

    // Tradicional WHILE
    let i = 0
    while (i < carros.length){
        console.log('Posicao', i, 'Carro', carros[i])
        i++
    }

    // Iterando nos índices (FOR..IN...)
    for (let i in carros){
        console.log('Posicao', i, 'Carro', carros[i])
    }

    // Iterando nos valores (FOR..OF...)
    for (let carro of carros){
        console.log('Carro', carro)
    }

    // Iterando índices e valores com FOR OF + entries
    for (let [i, carro] of carros.entries()){
        console.log('Posicao', i, 'Carro', carro)
    }
```

Percorrer Vetor

Saídas

```
❯ node percorrer_vetor.js
Posicao 0 Carro Argo
Posicao 1 Carro Gol
Posicao 2 Carro Onix
Posicao 0 Carro Argo
Posicao 1 Carro Gol
Posicao 2 Carro Onix
Posicao 0 Carro Argo
Posicao 1 Carro Gol
Posicao 2 Carro Onix
Carro Argo
Carro Gol
Carro Onix
Posicao 0 Carro Argo
Posicao 1 Carro Gol
Posicao 2 Carro Onix
```

```
1
2 const numeros = Array.from({length: 5}, (v, i) => i)
3 console.log(numeros)
4
5 numeros.push(9)
6 console.log(numeros)
7 // Appends new elements to the end of an array,
8 // and returns the new length of the array.
9
10 numeros.pop()
11 console.log(numeros)
12 // Removes the last element from an array and returns it.
13 // If the array is empty, undefined is returned
14 // and the array is not modified.
15
16 numeros.shift()
17 console.log(numeros)
18 // Removes the first element from an array and returns it.
19 // If the array is empty, undefined is returned
20 // and the array is not modified.
21
22 numeros.unshift(8, 10)
23 console.log(numeros)
24
```

Métodos JS em Vetores

push/unshift, pop/shift,
concat, join, sort, etc.

Saídas

```
❯ node metodos_vetores.js
[ 0, 1, 2, 3, 4 ]
[ 0, 1, 2, 3, 4, 9 ]
[ 0, 1, 2, 3, 4 ]
[ 1, 2, 3, 4 ]
[ 8, 10, 1, 2, 3, 4 ]
```

```
1
2 const novo_vetor = numeros.concat([-1, -10])
3 console.log(novo_vetor)
4 // Combines two or more arrays.
5 // This method returns a new array
6 // without modifying any existing arrays.
7
8
9 const novo_vetor2 = [...numeros, ...[-1, -10]]
10 console.log(novo_vetor2)
11 // Spread operator ...
12
13 console.log(numeros)
14
15 const texto = numeros.join('-')
16 console.log(texto)
17 // Adds all the elements of an array into a string,
18 // separated by the specified separator string.
19
20 numeros.sort()
21 console.log(numeros)
22 // Sorts an array in place.
23 // This method mutates the array and
24 // returns a reference to the same array.
25
26 numeros.sort((v1, v2) => {
27     if (v1 < v2) return -1
28     if (v1 > v2) return 1
29
30     return 0
31 })
32 console.log(numeros)
33
```

Métodos JS em Vetores

push/unshift, pop/shift, concat, join, sort, etc.

Saídas

```
[
  8, 10,  1,   2,
  3,  4, -1, -10
]
[
  8, 10,  1,   2,
  3,  4, -1, -10
]
[ 8, 10, 1, 2, 3, 4 ]
8-10-1-2-3-4
[ 1, 10, 2, 3, 4, 8 ]
[ 1, 2, 3, 4, 8, 10 ]
```

```
1
2 const alunos = ['Maria', 'Joana', 'Bia', 'João', 'Igor']
3
4 console.log(alunos)
5
6 const fatia = alunos.slice(0, 2)
7 console.log(fatia)
8 // Returns a copy of a section of an array
9
10 console.log(alunos)
11
12 const removidos = alunos.splice(1, 1)
13 console.log(alunos)
14 console.log(removidos)
15 // Removes elements from an array and, if necessary,
16 // inserts new elements in their place,
17 // returning the deleted elements.
18
```

Métodos JS em Vetores

slice e splice

Saídas

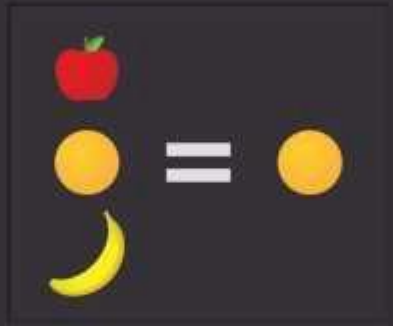
```
❯ node fatias_remover_valor_vetor.js
[ 'Maria', 'Joana', 'Bia', 'João', 'Igor' ]
[ 'Maria', 'Joana' ]
[ 'Maria', 'Joana', 'Bia', 'João', 'Igor' ]
[ 'Maria', 'Bia', 'João', 'Igor' ]
[ 'Joana' ]
```

Operações Comuns (focadas em Problemas):

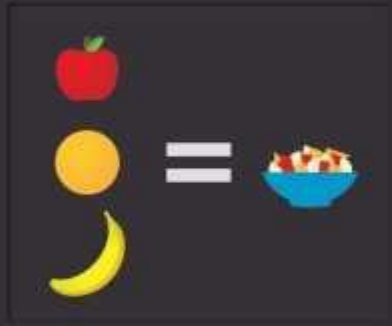
- Contar Ocorrências
- Somatórios
- Buscar
- Verificar se (não) contém
-

Map, Filter, Reduce

map()

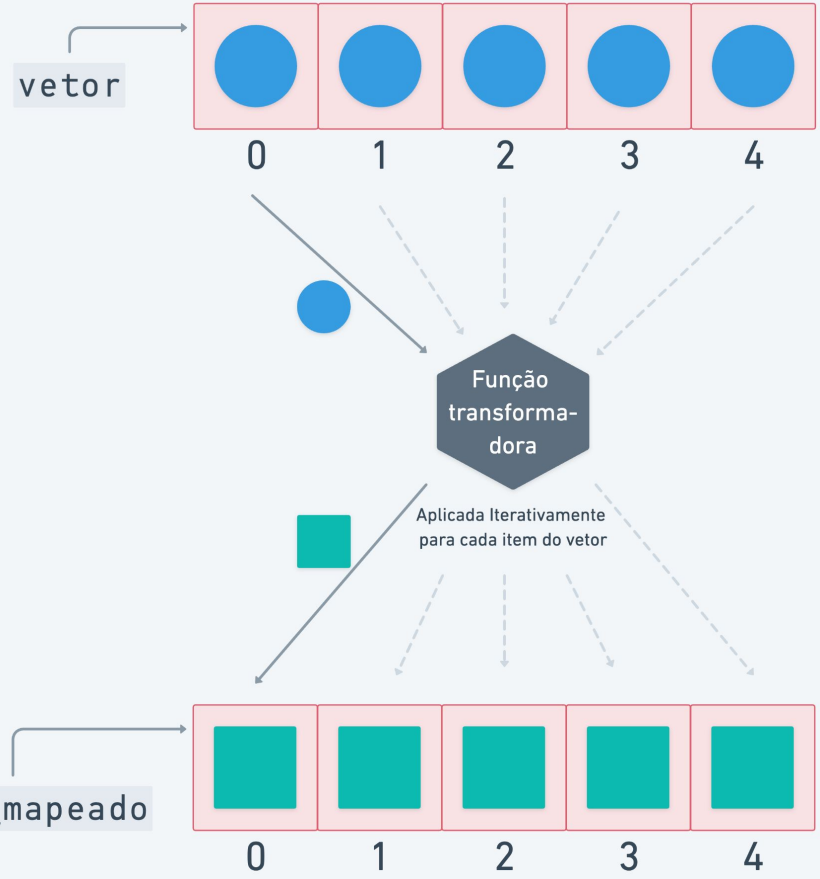


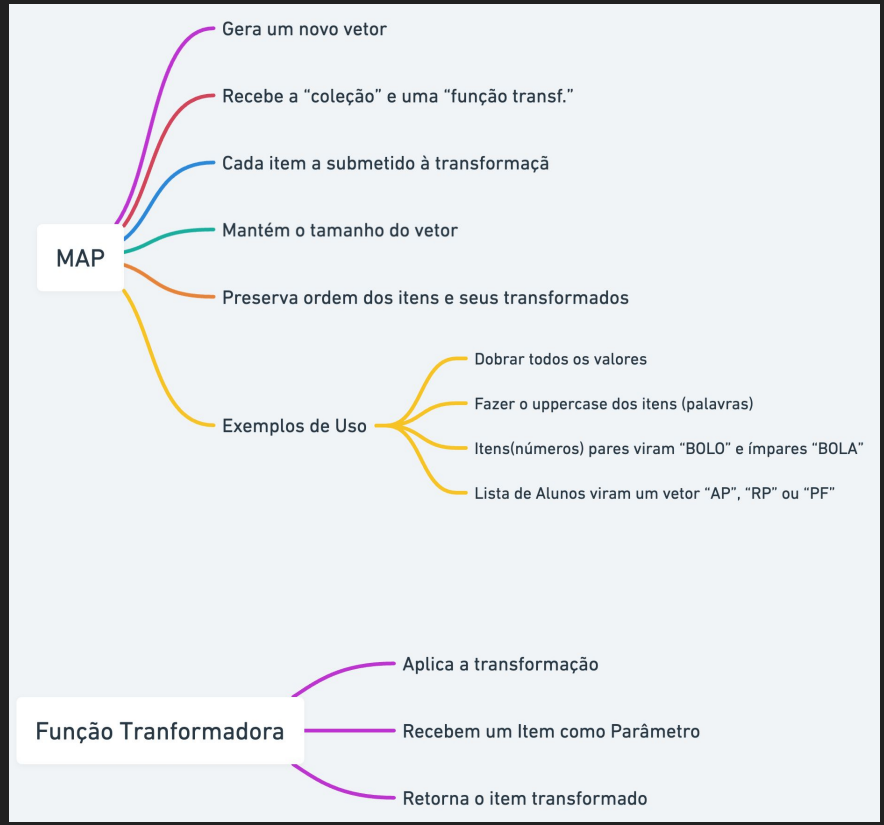
filter()



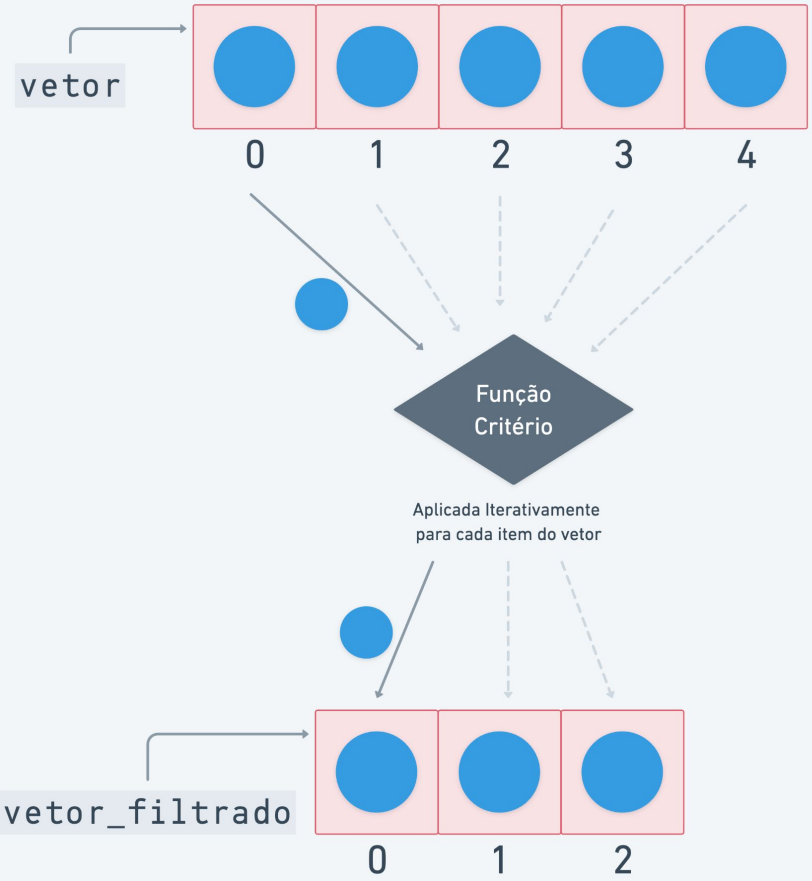
reduce()

MAP: Transformar itens





FILTER: Selecionar alguns Itens



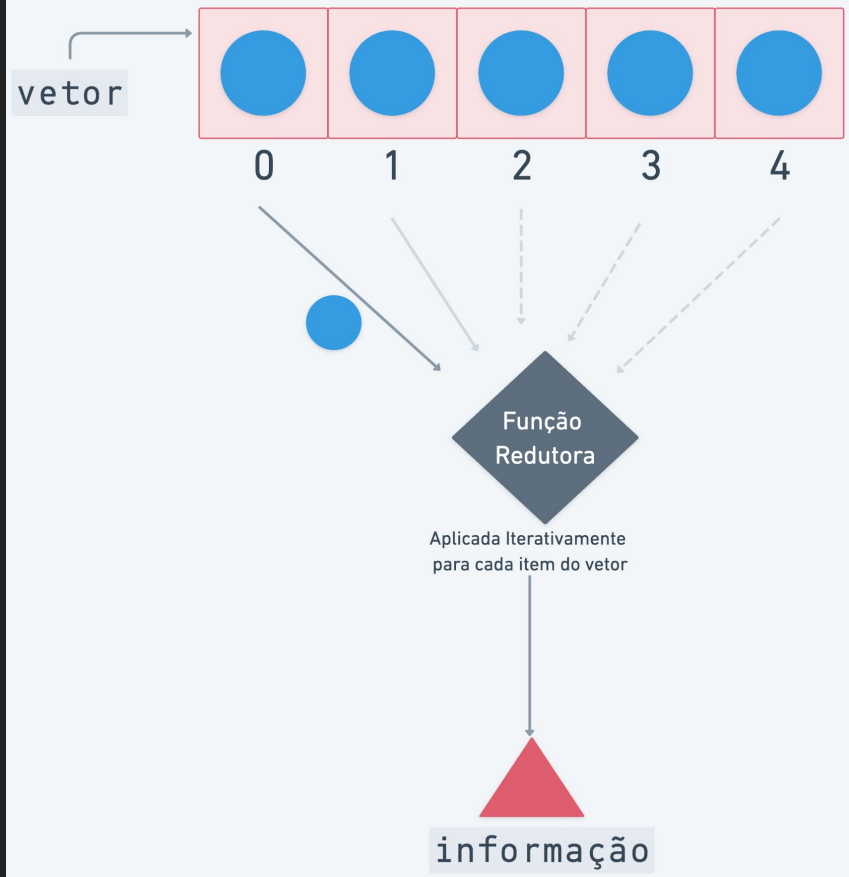
FILTER

- Gera um novo Vetor com alguns, nenhum ou todos os itens
- Usa um critério de aceite (função booleana)
- Não aplica qualquer transformação no itens selecionados
- Exemplos de Uso
  - Selecionar Números Pares
  - Obter só palavras que começam com vogal
  - Criar vetor de Alunos Aprovados
  - Filtrar apenas carros de até R$ 120k

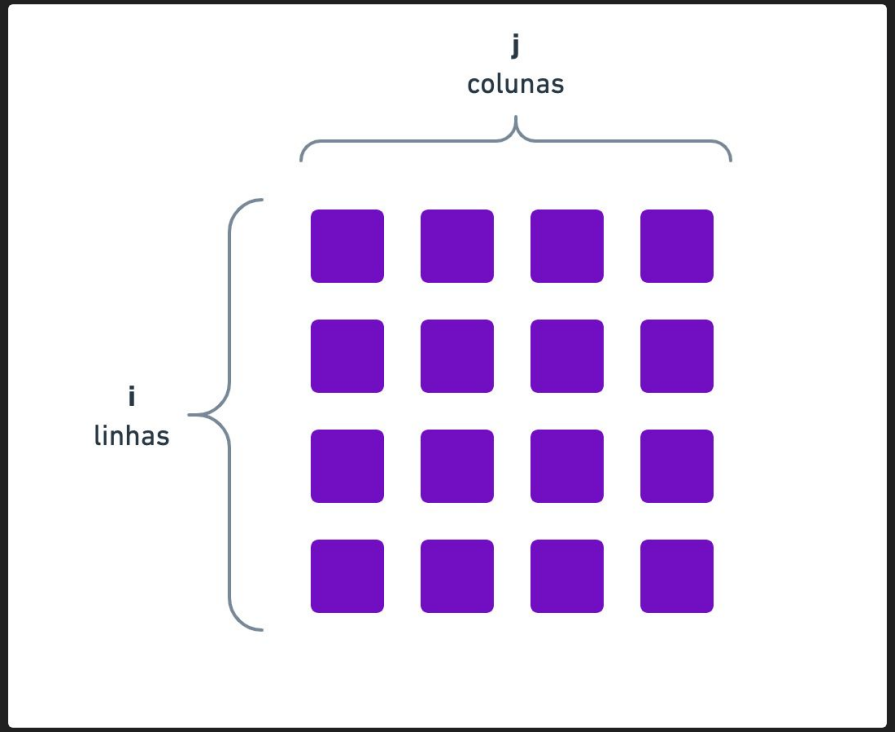
Função Critério

- Recebem um único argumento (item do vetor)
- Booleana: para cada item aplica um regra boolean para autorizar seleção (True)
- Pode aplicar qualquer regra porém deve retorna True ou False como resultado

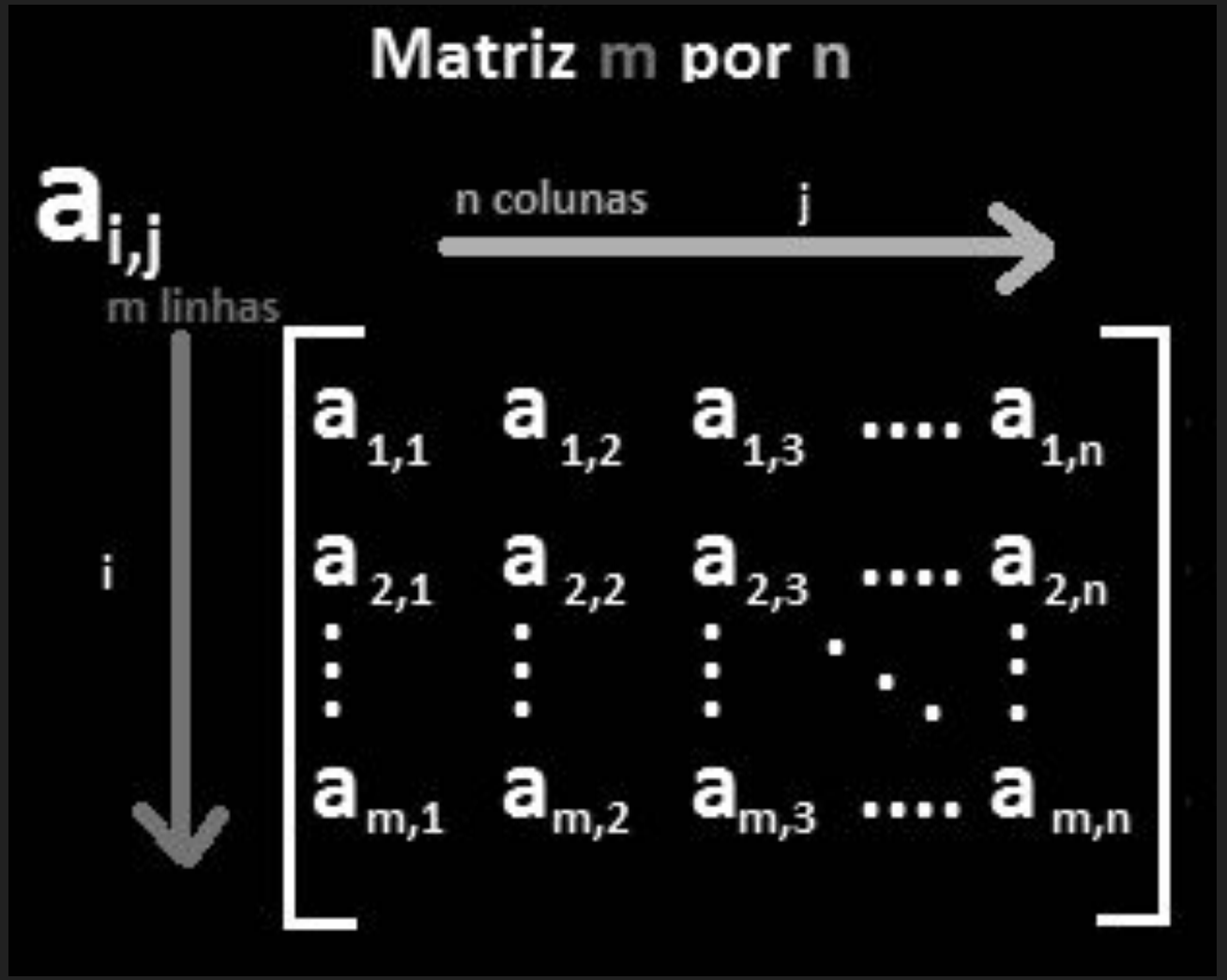
REDUCE: Obter uma informação



Vetor Multidimensional: Matriz
Vetor de vetores



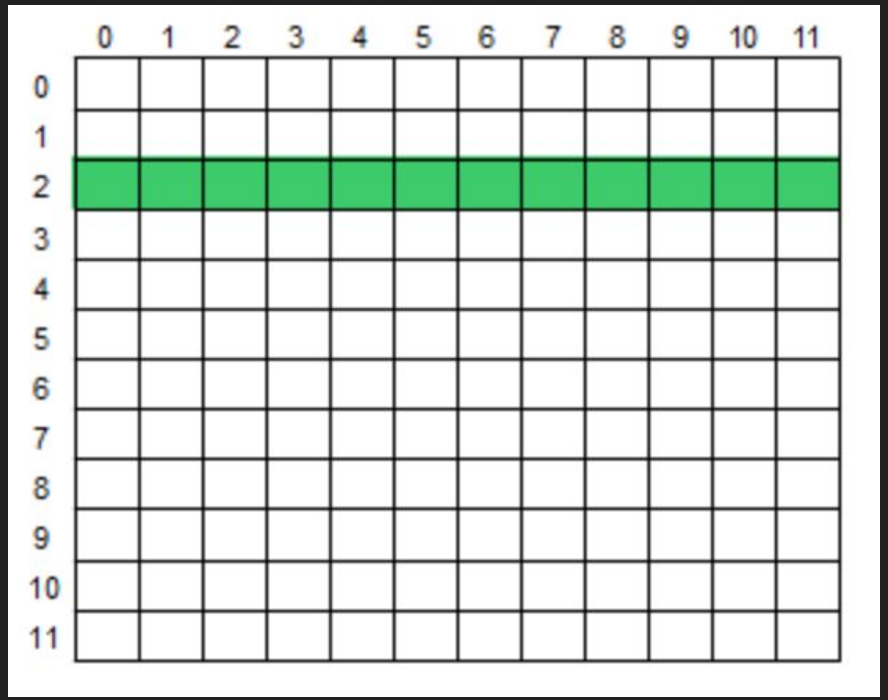
Estrutura com Duas Dimensões
(linhas e colunas

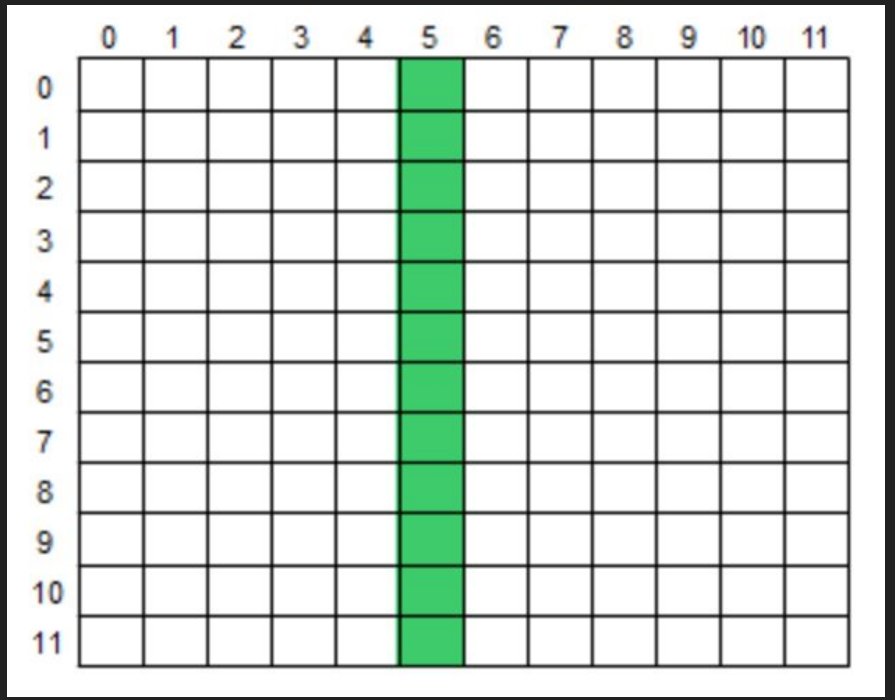


Item da Matriz

		j							
									colunas
	i	*0*	*1*	*2*	*3*	*4*	*5*	*6*	
i	*0*								
	1								
	2					177			numeros[2][4]
	3								
linhas									

Linhas e Colunas





Diagonais: Principal e Secundária

	0	1	2	3	j
0	55	3	2	-9	
1	-47	45	99	0	
2	-1	6	1	11	
3	4	9	33	6	
i					

i + j = ordem - 1 (Diagonal Secundária)

i = j (Diagonal Principal)

Matrix como "Banco de Dados"

	0	1	2
0	123	MARIETA JUSTOS	700
1	44	KARINA FERREIRA	8000
2	12	JOAQUIM JOSÉ	1200
3	320	ROGÉRIO SILVA	500
4	1	RICARDO RIOS	1200
5	14	KELSU TOP	1800

Datasets → https://www.kaggle.com/