

# Array Multidimensional

---

Até agora trabalhamos com arrays de uma dimensão. Porém, tanto o Java como o C# nos permite criar arrays com mais de uma dimensão, ou seja, *arrays multidimensionais*. Isso nos permite trabalhar com arrays para representar tabelas, matrizes ou até um tabuleiro de batalha naval. Voltando à analogia que fizemos com um armário cheio de gavetas, seria como se pudéssemos guardar dentro da gaveta de um armário um outro armário com gavetas. Veja a figura abaixo:

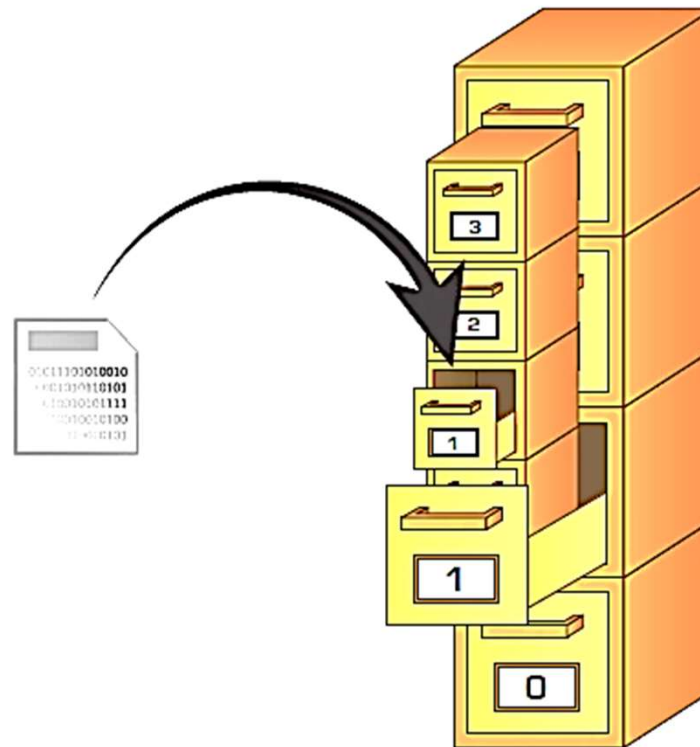
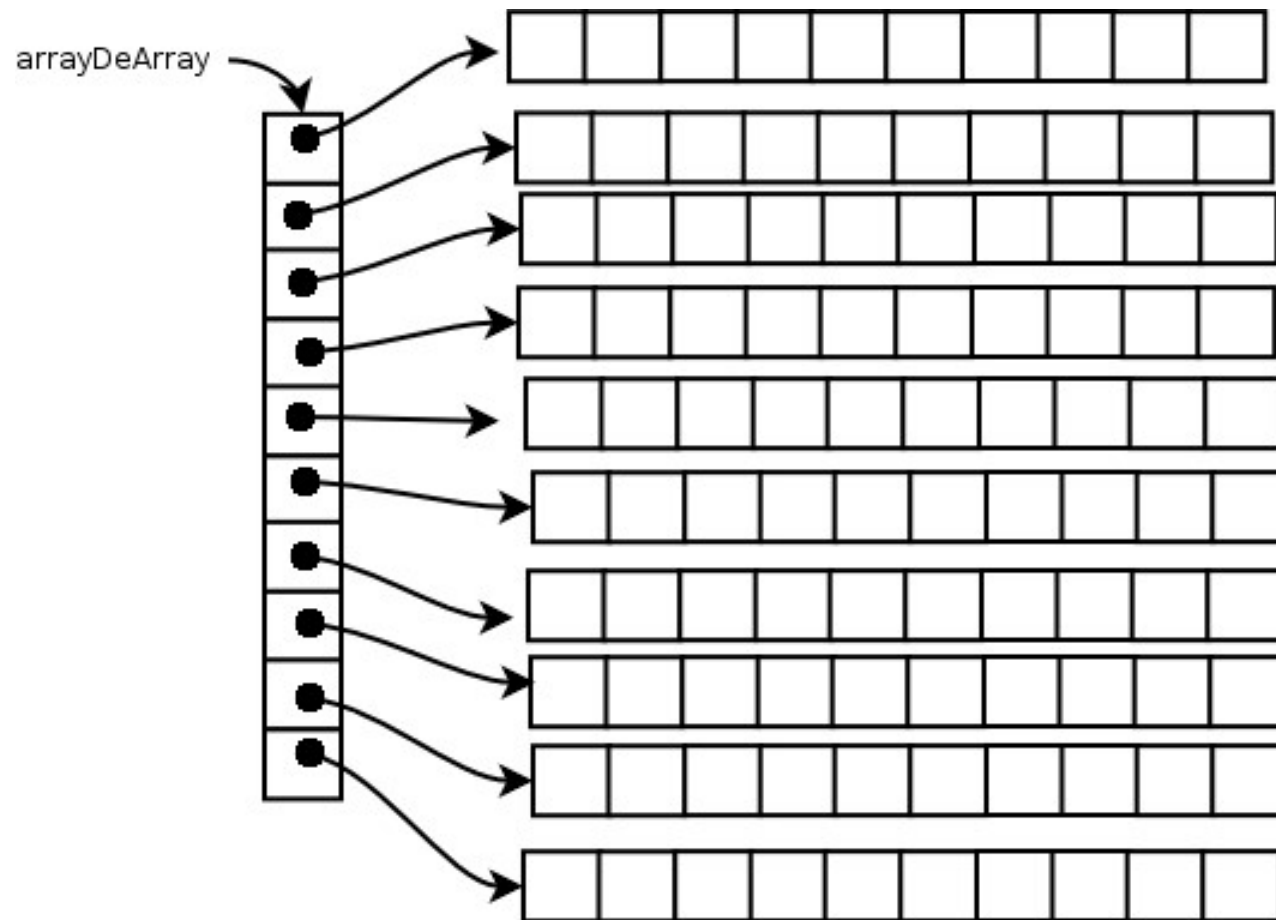


Figura 6.4: Abstração de um array multidimensional.

# Array Multidimensional

---



## Array Multidimensional - Matriz

---

```
int[][] vetorb = new int[2][2];
```

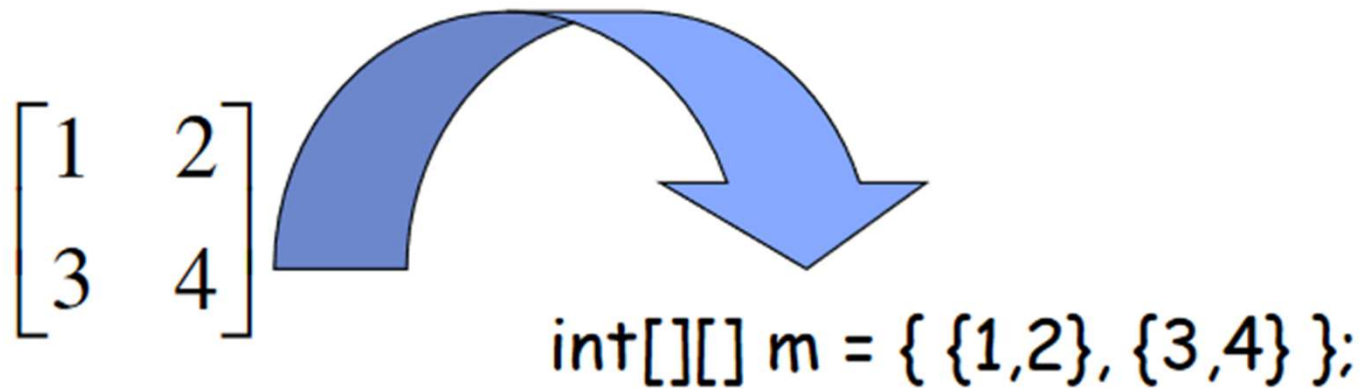
- `vetorb[0][0] = 10;`
- `vetorb[0][1] = 2;`
- `vetorb[1][0] = 34;`
- `vetorb[1][1] = 50;`

	0	1
0	10	2
1	34	50

vetorb

## Array Multidimensional - Matriz

---



## Array Multidimensional – Matriz - Prática

---

Crie a matriz abaixo e atribua os valores correspondentes:

$$M = \begin{bmatrix} 3 & 8 & 5 \\ 9 & 7 & 3 \\ 4 & 2 & 9 \end{bmatrix}$$

## Array Multidimensional – Matriz - Prática

---

```
int[][] m = new int[3][3];  
m[0][0] = 3;  
m[0][1] = 8;  
m[0][2] = 5;  
m[1][0] = 9;  
m[1][1] = 7;  
m[1][2] = 3;  
m[2][0] = 4;  
m[2][1] = 2;  
m[2][2] = 9;  
  
int[][] m2 = {{3,8,5},{9,7,3},{4,2,9}};
```

$$M = \begin{bmatrix} 3 & 8 & 5 \\ 9 & 7 & 3 \\ 4 & 2 & 9 \end{bmatrix}$$

## Array Multidimensional – Matriz – Exibindo

---

```
for(int x =0; x < a.length; x++){  
    for(int y = 0; y < a[x].length; y++){  
        System.out.print(a[x][y]);  
    }  
}
```

```
for(int[] w: a){  
    for(int y: w){  
        System.out.print(y);  
    }  
    System.out.println();  
}
```

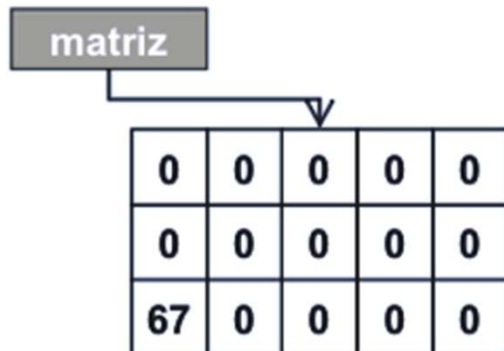
# Array Multidimensional - Matriz

---

Vetores multidimensionais podem ter dimensões homogêneas ou heterogêneas:

## Homogêneas:

```
int[][] matriz = new int[5][3];  
matriz[0][2] = 67;
```




0	0	0	0	0
0	0	0	0	0
67	0	0	0	0
0	0	0	0	0
0	0	0	0	0

## Heterogêneas:

```
int[][] amostras = new int[3][];  
amostras[0] = new int[3];  
amostras[1] = new int[4];  
amostras[2] = new int[2];  
amostras[2][1] = 15;  
amostras[0][1] = 56;  
amostras[1][2] = 31;
```

amostras



0	0	0	
56	0	15	0
0	31		
0			



## Exercícios

---

1. Ler uma matriz A e criar matriz B, invertendo linhas e colunas

Matriz A

6	3	15
1	41	9
55	31	11

Matriz B

6	1	55
3	41	31
15	9	11

2. Utilizando o exercício acima, criar uma matriz C com a soma dos elementos de A e B.

Matriz C

12	4	70
4	82	41
70	40	22

## Exercícios

---

3. Crie um programa que calcule a soma dos valores da diagonal principal de uma matriz 5x5. Veja a **diagonal principal** da matriz destacada no exemplo abaixo:

1	2	5	1	4
3	2	4	2	3
4	1	2	3	7
5	5	2	4	9
1	2	4	5	1

SOMA = 10

# Jogo da Velha

---

Utilizando vetores e matrizes, crie em Java um Jogo da Velha. O programa deve permitir que dois jogadores façam uma partida do jogo da velha, usando o computador para ver o tabuleiro.

- Cada jogador vai alternadamente informando a posição onde deseja colocar a sua peça.
- O programa deve impedir jogadas inválidas e determinar automaticamente quando o jogo terminou e quem foi o vencedor (Jogador 1, Jogador 2 ou Empate).
- A cada nova jogada o programa deve atualizar a situação do tabuleiro na tela.