

# Algoritmos e Programação: Fundamentos

Prof. Mateus Raeder



# Arrays

- Variável: armazena apenas um valor

```
int opcao;  
double nota;  
float tamanho;  
char letra;
```

tamanho

7.4

nota

2.5

letra

c

opcao

2

- Como criar em uma única variável, espaço para armazenar diversos valores diferentes?

# Arrays unidimensionais

- Arrays são objetos que permitem ao programador armazenar diversas variáveis do **mesmo tipo**



- Estas variáveis podem ser tanto tipos primitivos (int, char, ...) como outros objetos

Array de “int” →

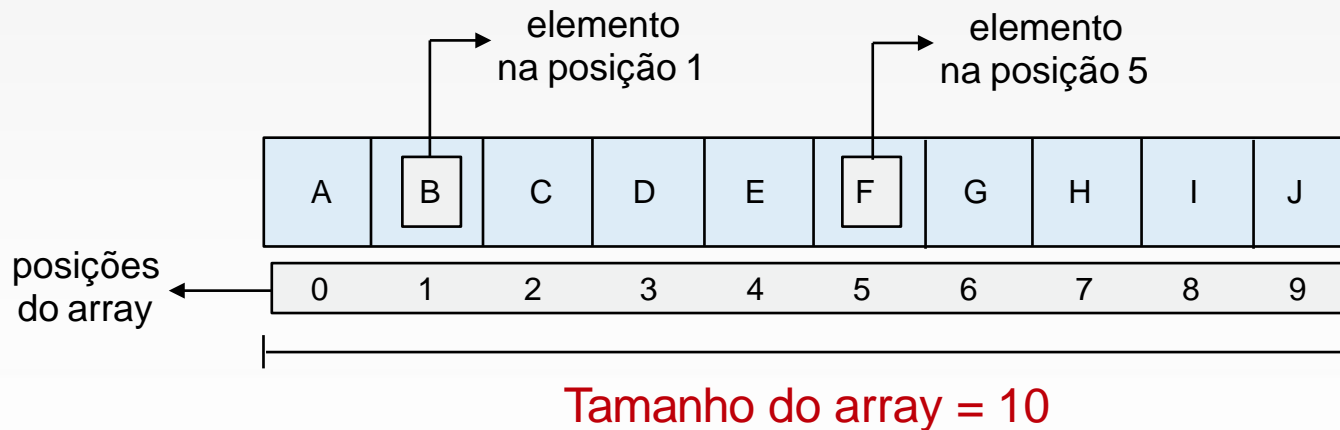
0	1	2	3	4	5	6	7	8	9
---	---	---	---	---	---	---	---	---	---

Array de “char” →

A	B	C	D	E	F	G	H	I	J
---	---	---	---	---	---	---	---	---	---

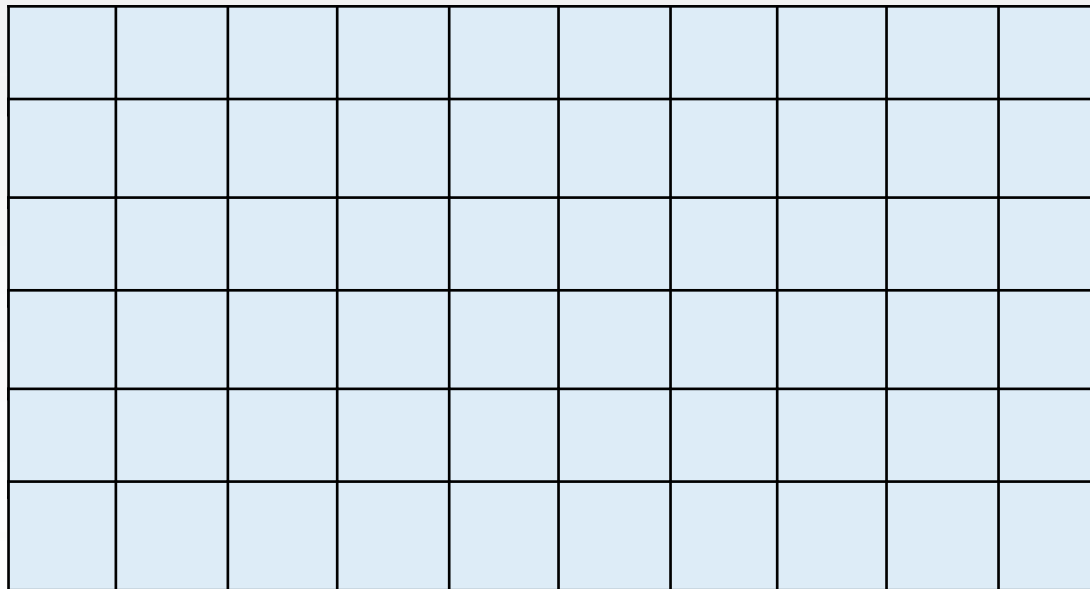
# Arrays unidimensionais

- Cada item do array é chamado de **elemento**
- Os elementos de um array são associados a uma **posição numérica**



# Arrays bidimensionais

- A estrutura de **arrays bidimensionais** também permite o armazenamento de diversas variáveis, porém, com uma estrutura similar a uma matriz:




- Estas variáveis podem ser tanto tipos primitivos (int, char, ...) como outros objetos

# Arrays bidimensionais

- Exemplo de **array bidimensional** de inteiros:

1	4	8	0	67	9	1	8
7	8	0	23	1	12	0	0
1	4	8	0	67	9	1	8
55	86	10	23	19	12	132	0
5	6	9	7	23	4	4	2

# Arrays bidimensionais

- Cada item do array é chamado de **elemento**
- Os elementos de um array são associados a uma **posição**, representadas por um par de valores inteiros

linhas do array

0	2.5	7.6	-8.5	0.0	1.1	3.56
1	11.3	34.12	-1.1	10.3	0.05	45.3
2	13.25	0.002	0.45	11.09	0.4	1.32
3	9.45	1.3	9.76	-56.4	22.5	20.45
	0	1	2	3	4	5

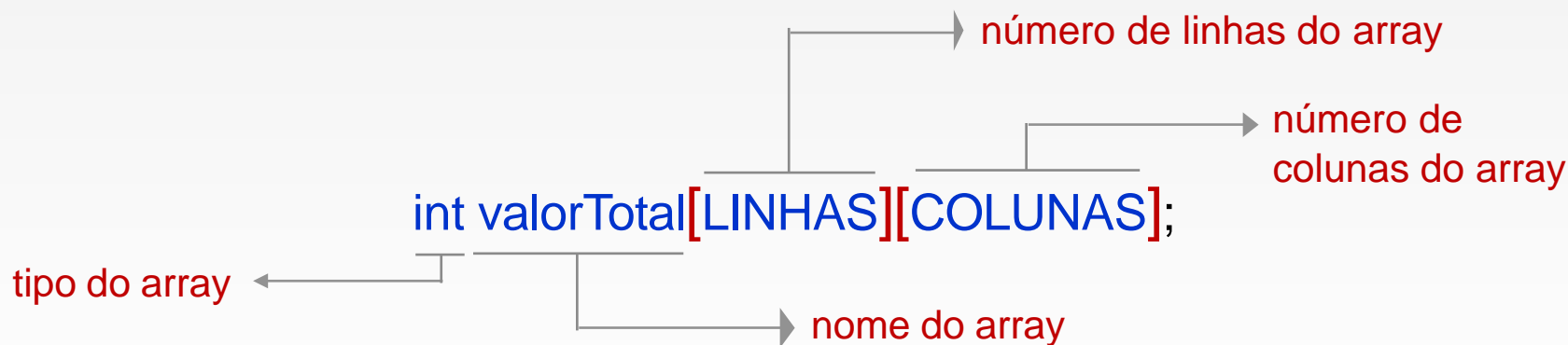
Elemento da posição (1, 3) do array

Elemento da posição (3, 1) do array

colunas do array

# Declarando arrays bidimensionais

- Arrays bidimensionais armazenam variáveis de um determinado tipo, e sua declaração ocorre da mesma forma de uma variável “comum”, porém utilizando um par de “[ ]” (colchetes) para a sua identificação.



criamos um array bidimensional de **inteiros**,  
chamado **valorTotal**, que possui  
**LINHAS** linhas e **COLUNAS** colunas  
(com um total de LINHAS x COLUNAS elementos)



# Declarando arrays bidimensionais

- Arrays bidimensionais armazenam variáveis de um determinado tipo, e sua declaração ocorre da mesma forma de uma variável “comum”, porém utilizando um par de “[ ]” (colchetes) para a sua identificação.

```
int valorTotal[4][6];
```

criamos um array bidimensional de **inteiros**,  
chamado **valorTotal**, que possui  
**4** linhas e **6** colunas  
(com um total de 24 elementos)

# Declarando arrays

## Exemplos:

```
//cria um array de inteiros com 5 linhas e 10 colunas  
int a[5][10];
```

```
//cria um array de double com 19 linhas e 30 colunas  
double b[19][30];
```

```
//cria um array de char com 100 linhas e 2 colunas  
char c[100][2];
```

```
//cria um array de float com 2000 linhas e 100 colunas  
float d[2000][100];
```

# Declarando e inicializando arrays

- Quando um array é criado, suas posições são automaticamente inicializadas com valores pré-determinados pela linguagem
- A **declaração** indica ao compilador que existe um array de determinado tipo e com determinado nome
- Devemos colocar valores no array para utilizá-lo

# Inicializando arrays

- Pode-se inicializar os valores de cada elemento do array de acordo com a necessidade/vontade, atribuindo valores para todos os seus elementos
- Supondo a declaração do array: `int valorTotal[4][6];`

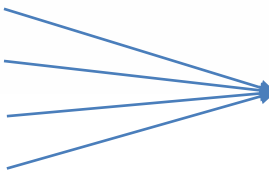
```
valorTotal[0][0] = 26;  
valorTotal[1][2] = 296;  
valorTotal[2][3] = 11;  
...  
valorTotal[3][2] = 1997;
```

ou

```
for(int i=0; i<4; i++)  
    for(int j=0; j<6; j++)  
        valorTotal[i][j] = 0;
```

- Pode-se também declarar, criar e inicializar arrays diretamente, da seguinte forma:

```
int valorTotal[4][6] = { {26, 296, 11, 34, 6, 76},  
                          {58, 98, 114, 754, 4, 54},  
                          {236, 1997, 0, -5, 7, 34},  
                          {6, 56, -132, 76, 8, 90} };
```



Array de arrays!

# Acessando elementos em arrays

- Os elementos do array são, então, acessados através de seus índices, que são as posições numéricas do array

```
printf("Valor da primeira posição é %d\n", valorTotal[0][0]);
```

```
printf("Valor da posição 0,1 é %d\n", valorTotal[0][1]);
```

```
int aux = valorTotal[2][1];
```

```
int temp = valorTotal[0][0] + valorTotal[1][0];
```

```
for(int i=0; i<4; i++)  
    for(int j=0; j<6; j++)  
        printf("Posição [%d,%d]: %d\n",i,j, valorTotal[i][j]);
```