

# Programação I

## Aula 5

Professor: Diogo Passos Ranghetti

## Sumário

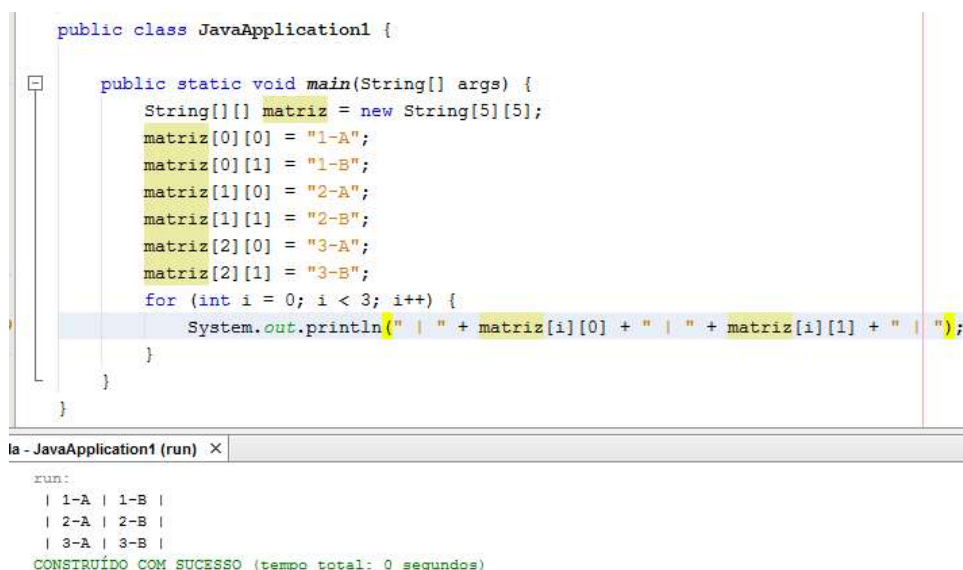
Matriz.....	3
Ponteiros.....	4
Alocação dinâmica de memória.....	5
Referências bibliográficas.....	6

# Matriz

De maneira simples, uma matriz pode ser considerado um vetor bidimensional sendo assim uma estrutura de dados homogeneia, ou seja, todos os elementos são do mesmo tipo. É possível também trabalhar com vetor multidimensional contendo mais de duas camadas.

A estrutura básica de um vetor mais conhecido com *Array*, é representada por seu nome e um índice, que será utilizado toda a vez em que precisar acessar um determinado elemento dentro do vetor. Ao trabalhar com matriz devesse passar dois ou mais índices, assim como o vetor matrizes também precisam ter o seu tamanho definido na sua inicialização possuindo um tamanho fixo, ou seja, não é possível redimensionar uma matriz ou adicionar a ela mais elementos do que este pode suportar.

Em Java ao instanciar uma matriz, determinamos seu tamanho e tipo, e ao informar o tamanho é alocada a quantidade de memória necessária para esse tamanho, assim como na criação de um vetor, cada uma das partes que compõe o vetor ou matriz possui o índice que determina a posição em que a informação é alocada, onde a matriz possui um índice que indica a linha e outro que indica a coluna levando em conta que seja um *Array* bidimensional, sendo a posição inicial definida pelo valor zero, e as demais seguem a sequência numérica, tanto vertical como horizontalmente.



```
public class JavaApplication1 {  
  
    public static void main(String[] args) {  
        String[][] matriz = new String[5][5];  
        matriz[0][0] = "1-A";  
        matriz[0][1] = "1-B";  
        matriz[1][0] = "2-A";  
        matriz[1][1] = "2-B";  
        matriz[2][0] = "3-A";  
        matriz[2][1] = "3-B";  
        for (int i = 0; i < 3; i++) {  
            System.out.println(" | " + matriz[i][0] + " | " + matriz[i][1] + " |");  
        }  
    }  
}
```

la - JavaApplication1 (run) X

run:

```
| 1-A | 1-B |  
| 2-A | 2-B |  
| 3-A | 3-B |  
CONSTRUIDO COM SUCESSO (tempo total: 0 segundos)
```

# Ponteiros

Ponteiro é uma variável capaz de armazenar um endereço de memória ou o endereço de outra variável.

No Java todos os tipos primitivos (ex.: *char*, *int*) são representados por valores, enquanto os tipos compostos, ou seja, as classes (ex.: *String*, *Integer*) utilizam referências.

Quando atribui um valor a uma variável, caso seu conteúdo seja passado para outra variável, a segunda variável vai receber a cópia do valor, de forma que se a primeira for alterada não influenciará a segunda.

Quando é atribuído o endereço ou referência de uma variável a outra, caso o valor da primeira seja alterado isso terá efeito na segunda, isso ocorre em decorrência do fato de que, o que realmente está passando é a referência do valor alocado na memória e não um valor propriamente dito.

No Java a alocação de memória se dá através do comando *new* de forma que a memória só é requisitada no momento em que o objeto é necessário evitando desperdício.

## Alocação dinâmica de memória

Em Java o gerenciamento de memória é realizado de forma automática, ou seja, o desenvolvedor não precisa realizar a gerencia da alocação ou deslocação de memória.

A forma de alocação dinâmica de memória funciona através da criação de objetos que são deslocados automaticamente por um mecanismo de *Garbage Collection*, sempre que não são mais necessários ao programa.

Um dos maiores responsáveis pela robustez, simplicidade e facilidade de desenvolvimento do Java é o seu suporte ao gerenciamento de alocação dinâmica automática de memória.

Sendo o Java responsável pela tarefa de alocação e deslocação de memória o Java não sofre com os erros de memória causados pela manipulação incorreta de ponteiros, situação muito comum para desenvolvedores que utilizam linguagens como o C++.

## **Referências bibliográficas**

DEITEL, H. M.; DEITEL, P. J. Java como Programar. 8ª.ed. São Paulo: Pearson Brasil, 2010.