

# PROGRAMAÇÃO DE SOLUÇÕES COMPUTACIONAIS

Prof. Ricardo Ribeiro Assink





## MÉTODOS DE PESQUISA

Quando se trabalha com vetores, eles poderão gerar grandes tabelas, dificultando localizar um determinado elemento de forma rápida. Imagine um vetor possuindo 4000 elementos (4000 nomes de pessoas).

Será que você conseguirá encontrar rapidamente um elemento desejado de forma manual, mesmo estando a lista de nomes em ordem alfabética? Certamente que não. Para solucionar este tipo de problema, você pode fazer pesquisas com uso de programação.

## ● PESQUISA LINEAR

Pesquisa Sequencial ou Linear: consiste em efetuar a busca da informação desejada a partir do primeiro elemento seqüencialmente até o último.

Localizando a informação no caminho, ela é apresentada. Este método de pesquisa é lento, porém eficiente nos casos em que um vetor encontra-se com seus elementos desordenados.



# MÉTODOS DE PESQUISA

## ● PESQUISA LINEAR

```
import javax.swing.*;
public class Exemplo4 {
    public static void main(String args[])
    {
        int i, flag;
        int numElementos = Integer.parseInt(JOptionPane.showInputDialog("Digite o número de pessoas a ser cadastrado "));
        String vetorPesquisado[] = new String[numElementos];
        for(i=0; i<numElementos;i++){
            vetorPesquisado[i] = JOptionPane.showInputDialog("Digite o nome para cadastro");
        }
        String elementoProcurado = JOptionPane.showInputDialog("Digite o nome a ser procurado");
        flag = 0;
        for(i=0; i<numElementos;i++){
            if (vetorPesquisado[i].equalsIgnoreCase(elementoProcurado)){
                JOptionPane.showMessageDialog(null,"o valor procurado foi encontrado na posição "+i);
                flag = 1;
            }
        }
        if (flag == 0){
            JOptionPane.showMessageDialog(null, "o nome não foi encontrado.");
        }
    }
}
```



## MATRIZES

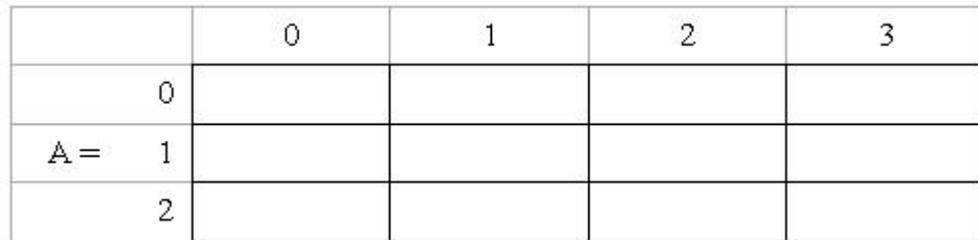
Uma variável indexada Bidimensional também conhecida como “Matriz”, como o próprio nome já indica, possui duas dimensões (linha e coluna), sendo possível definir variáveis com quaisquer tipo de dados.

# MATRIZES

## EXEMPLO:

Definir uma variável indexada bidimensional como sendo do tipo Real, sendo que a matriz deverá ter 3 linhas e 4 colunas. Esta matriz deverá corresponder a 12 posições de memória. .

```
public class Exemplo_Declaracao {  
    public static void main(String args[]) {  
        double A[ ][ ] = new double[3][4];  
        <comandos>  
    }  
}
```



	0	1	2	3
0				
A = 1				
2				

Após a definição da variável A, a memória estará como mostrada no esquema ao lado.

Os Valores numéricos apresentados acima(coluna) e a esquerda(linha) correspondem aos índices da variável indexada bidimensional. Portanto se tivermos uma matriz com 10 linhas, a primeira linha terá como índice o valor 0 e a última linha terá como índice o valor 9.

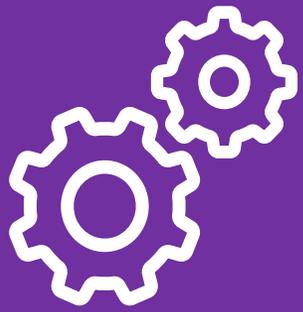


Descrição de imagem:

Mostra quadro com representando espaços na memória utilizados para uma matriz.

1 – LINHA: 0 1 2

2 – COLUNA: 0 1 2 3



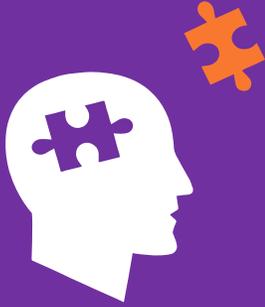
# VARIÁVEL INDEXADA BIDIMENSIONAL (MATRIZ):

## EXEMPLO

```
import javax.swing.*;
public class Exemplo {
    public static void main(String args[ ]) {
        double A[ ][ ] = new double[3][4];
        int i, j;

        // leitura dos valores
        for(i=0; i<3; i++) {
            for (j=0; j<4; j++){
                A[i][j] = Double.parseDouble(JOptionPane.showInputDialog("Digite o valor da linha "+i+" e coluna "+j+": "));
            }
        }

        // escrita dos valores
        for(i=0; i<3; i++){
            for (j=0; j<4; j++){
                JOptionPane.showMessageDialog(null, "O valor contido na linha " +i+" e coluna "+j+ " eh " + A[i][j]);
            }
        }
    }
}
```



## (JAVA) EXERCÍCIO 25



Implemente e execute o código de pesquisa linear dado em aula, incluindo códigos para INTERROMPER a varredura completa do vetor, quando o elemento procurado é encontrado



(Fazer como busca ativa!) Faça um algoritmo que leia uma matriz de ordem 3x3 de números inteiros. Após a leitura faça:

- A** Calcule e mostre a soma dos elementos da primeira coluna;
- B** Calcule e mostre a soma dos elementos de cada coluna;
- C** Calcule e mostre o produto dos elementos da primeira linha;
- D** Calcule e mostre a soma de todos os elementos da matriz;
- E** Calcule e mostre a média dos elementos da matriz;
- F** Calcule e mostre os elementos que são maiores que a média;
- G** Calcule e mostre o maior elemento da matriz e sua posição;
- H** Calcule e mostre o menor elemento da matriz e sua posição;

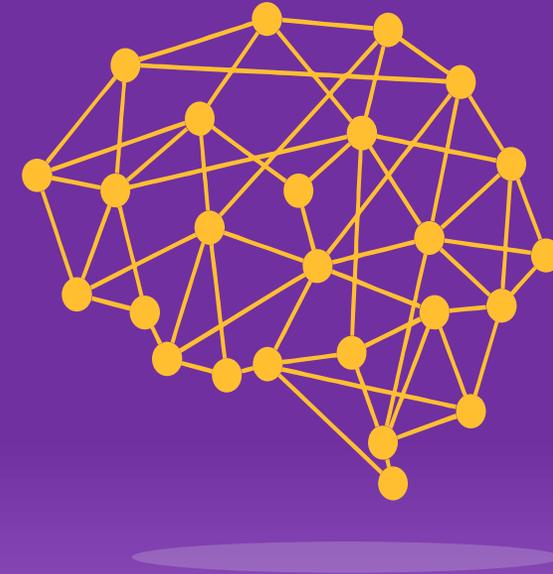


(JAVA) EXERCÍCIO 26



# Busca Ativa!

- 1 Procure e assista vídeos na internet que demonstram o funcionamento de matrizes.
- 2 Implemente e execute TODOS os exemplos da aula de hoje.
- 3 Crie um exemplo de uso de matrizes.



**FIM**

