

PROGRAMAÇÃO DE SOLUÇÕES COMPUTACIONAIS

Prof. Ricardo Ribeiro Assink

Prof. Edson Lessa

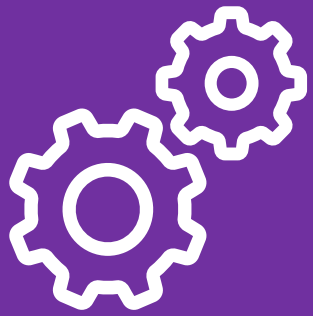




ESTRUTURAS DE REPETIÇÃO: Para.. Faça - FOR

Juntamente com as estruturas de seleção, as estruturas de repetição são de crucial importância para a programação do algoritmo.

As estruturas de repetição nos possibilitam executar o mesmo trecho de código várias vezes seguidas, enquanto um dado critério não é satisfeito.



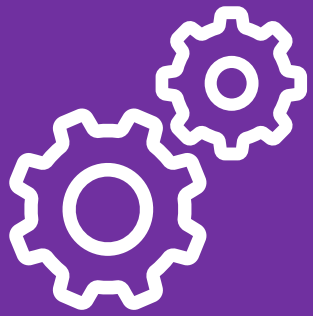
ESTRUTURAS DE REPETIÇÃO: Para.. Faça - FOR

```
for( < VI > ; < CP >; < RC > ){  
    <comando 1>;  
    <comando 2>;  
}
```

VI Valor inicial do contador

CP Condição de parada

RC Regra do contador



(JAVA) ESTRUTURAS DE REPETIÇÃO: Para.. Faça - FOR

```
import javax.swing.JOptionPane;  
public class Exemplofor {  
    public static void main(String[] args) {
```

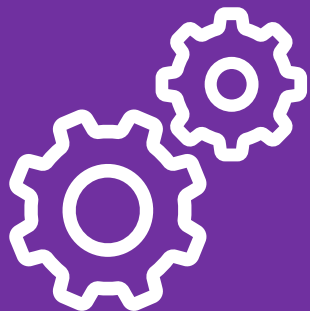
```
        for ( int i = 0; i < 5; i++ ){  
            JOptionPane.showMessageDialog(null,"Valor de i: " + i);  
        }
```

```
    }  
}
```

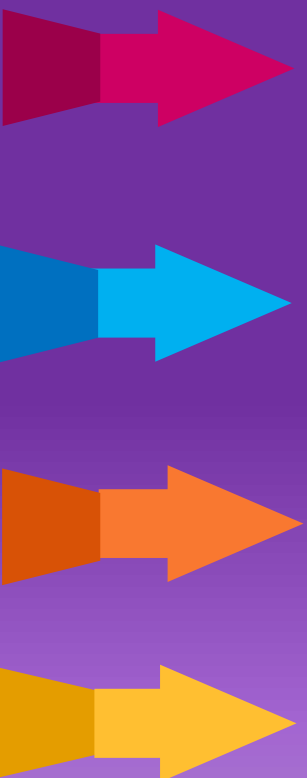


ESTRUTURAS DE REPETIÇÃO: Enquanto.. Faça: WHILE

Para repetir um conjunto de instruções enquanto uma certa condição for satisfeita



ESTRUTURAS DE REPETIÇÃO de Repetição: Enquanto.. Faça - WHILE

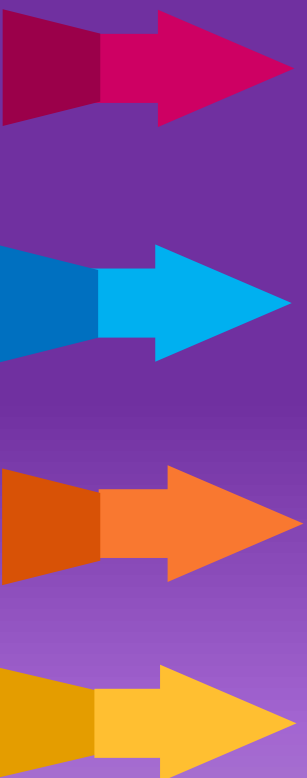


```
while(< CP > ){  
    <comando 1>;  
    <comando 2>;  
}
```

CP Condição de parada



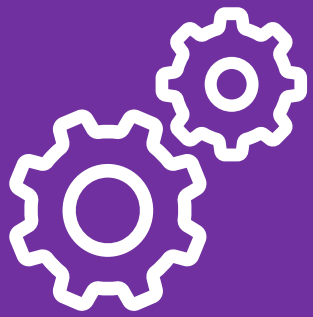
(JAVA) ESTRUTURAS DE REPETIÇÃO: Enquanto.. Faça - WHILE



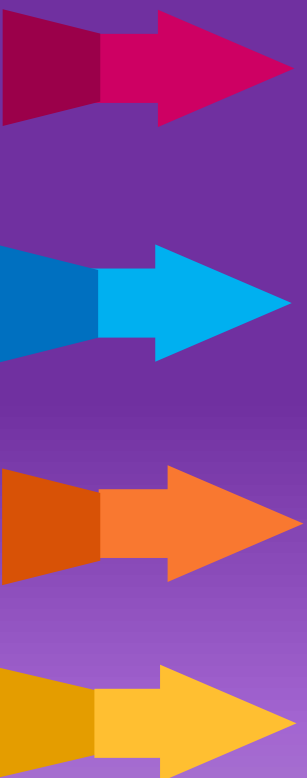
```
import javax.swing.JOptionPane;
public class Exemplowhile {
    public static void main(String[] args) {
        int i = 0;
        while( i < 5){
            JOptionPane.showMessageDialog(null,"Valor de i: " + i);
            i++;
        }
    }
}
```

ESTRUTURAS DE REPETIÇÃO: Repita.. Até: DO - WHILE

Para repetir um conjunto de instruções enquanto uma certa condição for satisfeita, este conjunto é executado pelo menos 1 vez e esta é a diferença para a estrutura Enquanto.. Faça(while) que pode não executar nenhuma vez caso a condição não seja satisfeita.



ESTRUTURAS DE REPETIÇÃO: Repita.. Até: DO - WHILE

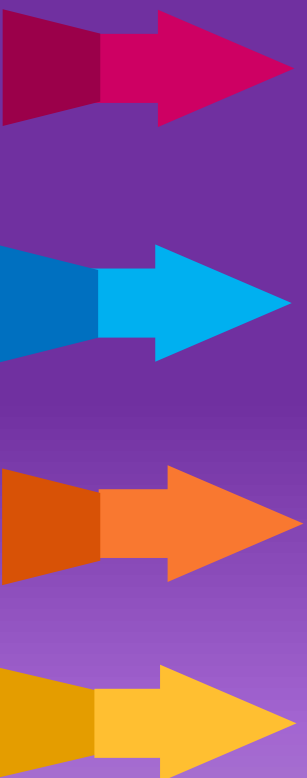


```
do{  
    <comando 1>;  
    <comando 2>;  
} while(CP);
```

CP Condição de parada



(JAVA) ESTRUTURAS DE REPETIÇÃO: Repita.. Até: DO - WHILE



```
import javax.swing.JOptionPane;
public class Exemplowhile {
    public static void main(String[] args) {
        int i = 0;
        do{
            JOptionPane.showMessageDialog(null,"Valor de i: " + i);
            i++;
        }while(i < 5);
    }
}
```

VARIÁVEL INDEXADA:

Uma variável indexada corresponde a uma sequência de posições de memória, a qual daremos único Nome, sendo que cada uma destas pode ser acessada através do que conhecemos por índice.

O índice corresponde a um valor numérico (exceto Real).

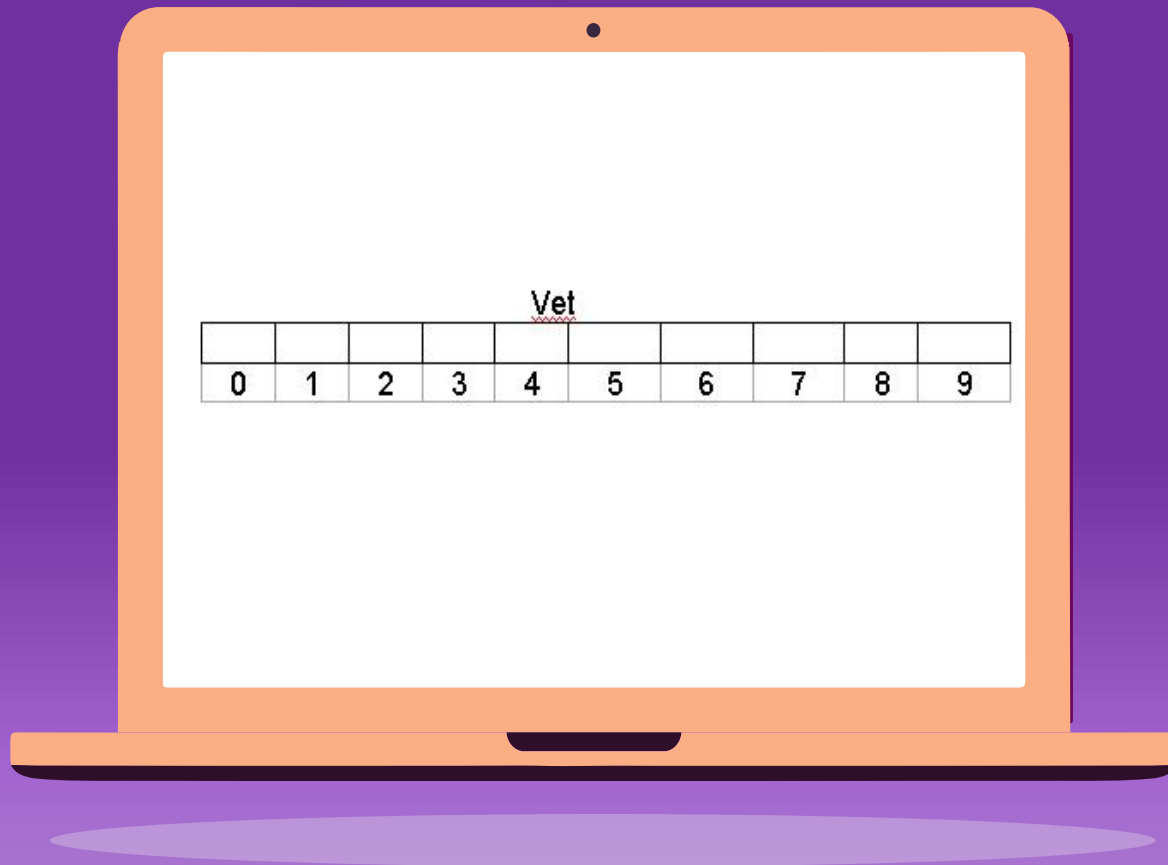
Cada uma das posições de memória de uma variável indexada pode receber valores no decorrer do algoritmo como se fosse uma variável comum, a única diferença reside na Sintaxe de utilização desta variável.

TIPOS

**Variáveis
Indexadas
Unidimensionais
(Vetor)**

**Variáveis
Indexadas
Bidimensionais
(Matriz)**

VARIÁVEL INDEXADA UNIDIMENSIONAL (VETOR) :



EXEMPLO:

Definir uma variável indexada como sendo do tipo REAL, sendo que a mesma deverá corresponder a 10 posições de memória.

```
public class Exemplo {  
    public static void main(String args[ ]) {  
        double vet[ ] = new double[10];  
        <comandos>;  
    }  
}
```



Descrição de imagem:

Mostra quadro com os números de 0 a 9 e lacunas representando espaços na memória.
1 – 0 1 2 3 4 5 6 7 8 9

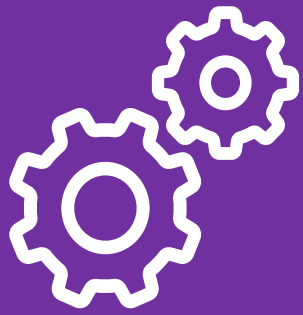
VARIÁVEL INDEXADA UNIDIMENSIONAL (VETOR)

EXEMPLO 01

```
public class Atribui {  
    public static void main(String args[]) {  
        String nomes[ ] = new String[20];  
        nomes[0] = "João da Silva";  
        nomes[1] = "Ana";  
    }  
}
```

EXEMPLO 02

```
public class VetorAtribuir {  
    public static void main ( String args[]) {  
        int X[ ] = new int[20]; // declarando e instanciando o vetor  
        X [0] = 100;  
        X [1] = X [0] + 3;  
    }  
}
```



VARIÁVEL INDEXADA UNIDIMENSIONAL (VETOR):

Leitura - Exemplo

```
import javax.swing.*;
public class Exemplo1 {
    public static void main(String args[]) {
        int i;
        String nomes[] = new String[20];
        for(i = 0; i < 20; i++) {
            nomes[i] = JOptionPane.showInputDialog("Digite o nome: ");
        }
    }
}
```



VARIÁVEL INDEXADA UNIDIMENSIONAL (VETOR):

Escrita - Exemplo

```
import javax.swing.*;
public class Exemplo3 {
    public static void main(String args[]) {
        int i;
        String nomes[ ] = new String[20];
        nomes[0] = "Unisul";
        nomes[1] = "Aluno";
        nomes[2] = "Sistema";
        for(i = 0; i < 3; i++) {
            JOptionPane.showMessageDialog(null, "o nome na posição "+i+ " é "+ nomes[i] );
        }
    }
}
```



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.

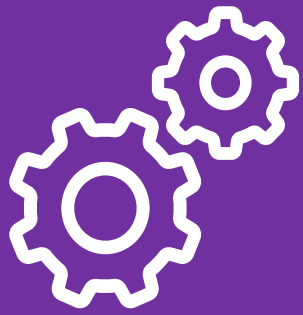


Descrição de imagem:

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

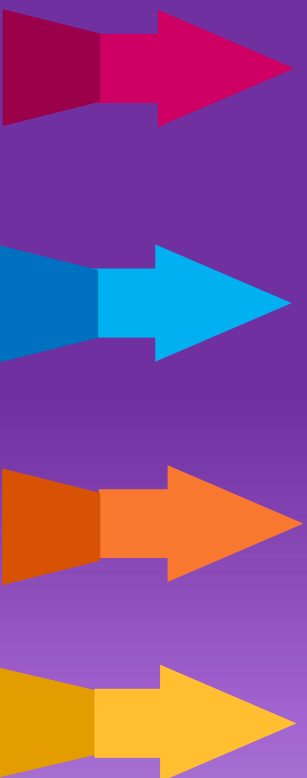
1 – LINHA: 0 1 2

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]);
            }
        }
    }
}
```

Busca Ativa!

- 1 Procure e assista vídeos na internet que demonstram o funcionamento de vetores e matrizes.
- 2 Implemente e execute TODOS os exemplos da aula de hoje.
- 3 Crie um exemplo de uso de matrizes.
- 4 Cole o exercício que você criou, no forms que será passado em aula pelo professor.

