53 45 52 45 49 20 46 49 45 4c 20
41 4f 53 20 50 52 45 43 45 49 54
4f 53 20 44 41 20 48 4f 4e 52 41
20 45 20 44 41 20 43 49 c3 8a 4e
43 49 41 2c 20 50 52 4f 4d 4f 56
45 4e 44 4f 20 4f 20 55 53 4f 20
45 20 4f 20 44 45 53 45 4e 56 4f
4c 56 49 4d 45 4e 54 4f 20 44 41
20 49 4e 46 4f 52 4d c3 81 54 49
43 41 20 45 4d 20 42 45 4e 45 46
c3 8d 43 49 4f 20 44 4f 20 43 49
44 41 44 c3 83 4f 20 45 20 44 41
20 53 4f 43 49 45 44 41 44 45 2e

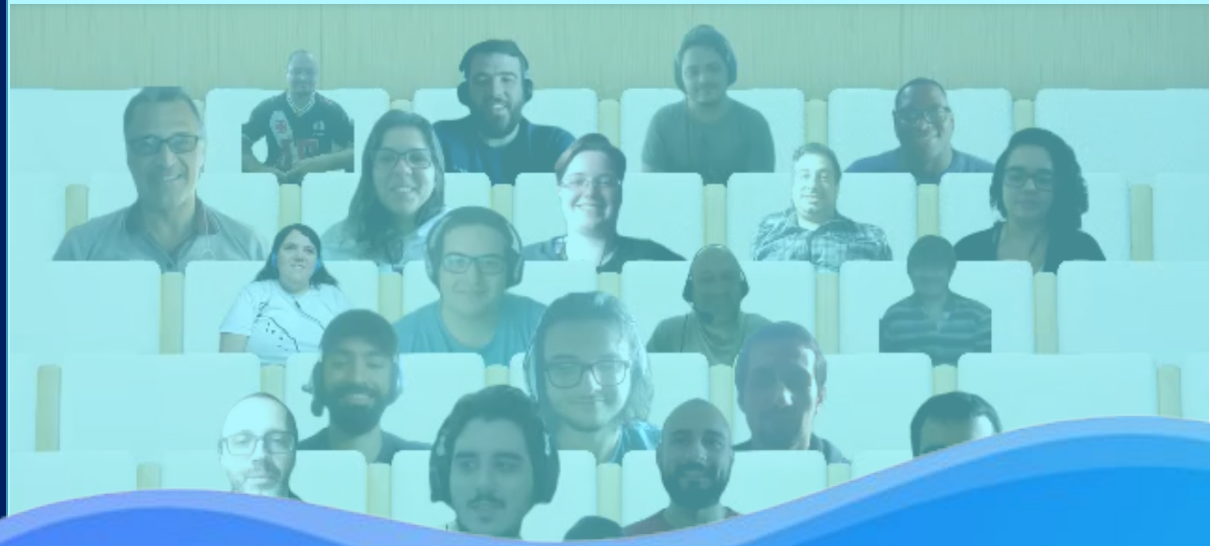
RESIDÊNCIA DE

SOFTWARE

CAPACITAR
TREINAR
EMPREGAR
TRANSFORMAR

Firjan SENAI

serratec
Parque Tecnológico Região Serrana



Entender a estrutura de dados (vetor e matriz)
Data:23/03/2022

Revisão Estruturas de Repetição

Enquanto - A condição é verificada no início do bloco. No exemplo abaixo precisamos fazer a leitura do nome antes e dentro bloco de repetição.

```
programa
{
    funcao inicio()
    {    cadeia nome

         escreva("Nome:")
         leia(nome)
         enquanto(nome != ""){
              escreva("Nome:")
              leia(nome)
         }

    }
}
```

Nesse exemplo a condição de parada é que a variável **i** seja igual a 5. Precisamos incrementar a variável **i** para que a condição seja satisfeita.

```
programa
{
    funcao inicio()
    {    cadeia nome
         inteiro i=1

         enquanto(i <=5 ){
              escreva("Nome:")
              leia(nome)
              i++
         }

    }
}
```

Revisão Estruturas de Repetição

Faca Enquanto - A condição é verificada no final do bloco.

```
programa
{
    funcao inicio()
    {    cadeia nome

        faca{
            escreva("Nome:")
            leia(nome)
        }enquanto(nome != "")
    }
}
```

Precisamos também incrementar a variável **i**

```
programa
{
    funcao inicio()
    {    cadeia nome
        inteiro i=1

        faca{
            escreva("Nome:")
            leia(nome)
            i++
        }enquanto(i <=5)
    }
}
```

Revisão Estruturas de Repetição

Para

Na estrutura do para temos criar as seguintes definições

- Criar e inicializar uma variável
- Atribuir a condição de parada
- Definir o incremento

```
programa
{
	funcao inicio()
	{	cadeia nome

		para(inteiro i=1; i <=5; i++){
			escreva("Nome:")
			leia(nome)
		}
	}
}
```

Revisão Vetor

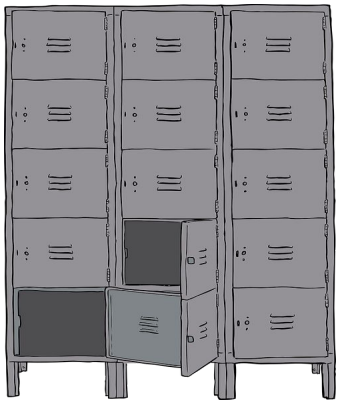
```
programa
{

    funcao inicio()
    {
        cadeia nome[5]
        para(inteiro i=0; i <=4; i++){
            escreva("Nome:")

            leia(nome[i])
        }

        //Imprimindo primeiro e último nome do vetor
        escreva("Primeiro Nome:" +nome[0]+ " Último Nome:", nome[4])

    }
}
```

variável nome[5]

0	1	2	3	4
João	Maria	Ana	Fabiano	Carla

Matrizes

É um array bidimensional onde teremos a linha e coluna para identificar uma matriz. Para acessar um valor individual é necessário dois índices, um para linha e outro para coluna. Durante a inicializadão deve-se respeitar quantidade de linhas e colunas informados.

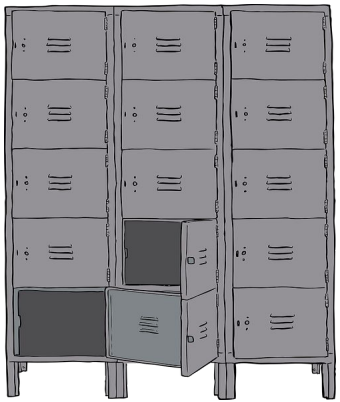


Tipo	Nome	Capacidade
mochila	meuArmario	[5][3]

mochila meuArmario[5][3]

Sintaxe:
tipo de dados nome da matriz[qtde linhas] [qtde colunas]

Matrizes

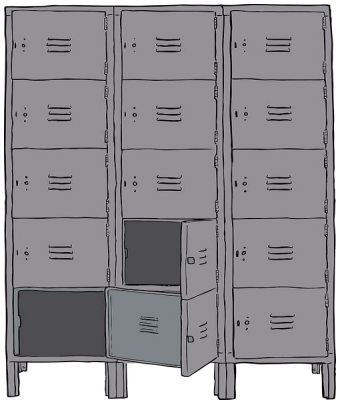


Tipo	Nome	Capacidade
mochila	meuArmario	[5][3]

mochila meuArmario[5][3];

	[0]	[1]	[2]
[0]	1.3	1.5	1.6
[1]	1.2	1.7	2.2
[2]	100.5	75.6	2.95

Matrizes



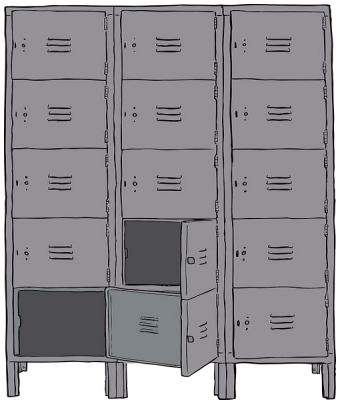
Tipo	Nome	Capacidade
mochila	meuArmario	[5][3]

mochila meuArmario[5][3];

	[0]	[1]	[2]
[0]	1.3	1.5	1.6
[1]	1.2	1.7	2.2
[2]	100.5	75.6	2.95

Tipo	Nome	Capacidade

Matrizes



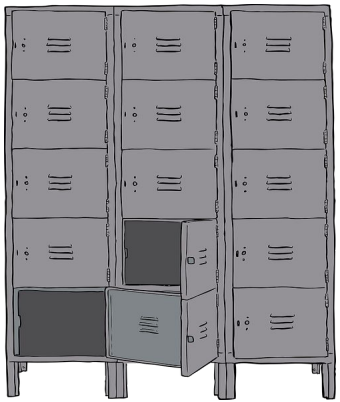
Tipo	Nome	Capacidade
mochila	meuArmario	[5][3]

mochila meuArmario[5][3];

	[0]	[1]	[2]
[0]	1.3	1.5	1.6
[1]	1.2	1.7	2.2
[2]	100.5	75.6	2.95

Tipo	Nome	Capacidade
real		

Matrizes



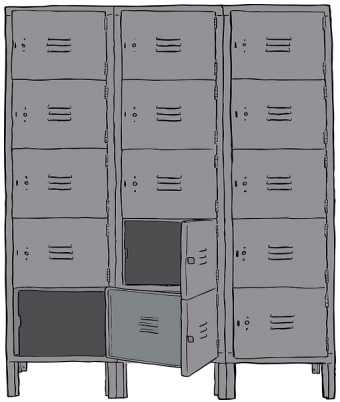
Tipo	Nome	Capacidade
mochila	meuArmario	[5][3]

mochila meuArmario[5][3];

	[0]	[1]	[2]
[0]	1.3	1.5	1.6
[1]	1.2	1.7	2.2
[2]	100.5	75.6	2.95

Tipo	Nome	Capacidade
real	minhaMatriz	

Matrizes



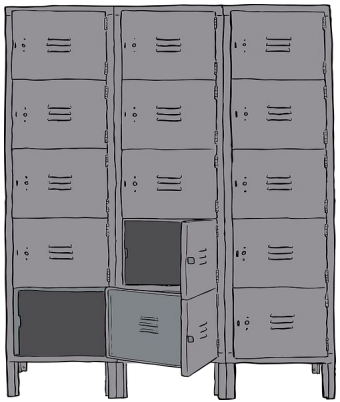
Tipo	Nome	Capacidade
mochila	meuArmario	[5][3]

mochila meuArmario[5][3];

	[0]	[1]	[2]
[0]	1.3	1.5	1.6
[1]	1.2	1.7	2.2
[2]	100.5	75.6	2.95

Tipo	Nome	Capacidade
real	minhaMatriz	[3][3]

Matrizes



Tipo	Nome	Capacidade
mochila	meuArmario	[5][3]

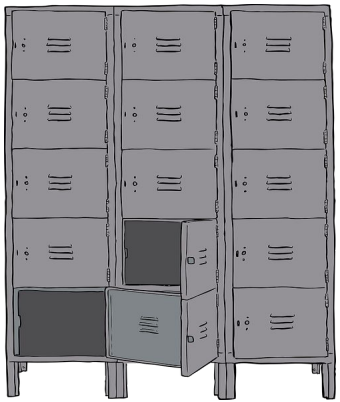
mochila meuArmario[5][3]

	[0]	[1]	[2]
[0]	1.3	1.5	1.6
[1]	1.2	1.7	2.2
[2]	100.5	75.6	2.95

Tipo	Nome	Capacidade
real	minhaMatriz	[3][3]

real minhaMatriz[3][3]

Matrizes



Tipo	Nome	Capacidade
mochila	meuArmario	[5][3]

mochila meuArmario[5][3]

minhaMatriz[1][2] = 5.0

	[0]	[1]	[2]
[0]	1.3	1.5	1.6
[1]	1.2	1.7	5.0
[2]	100.5	75.6	2.95

Tipo	Nome	Capacidade
real	minhaMatriz	[3][3]

real minhaMatriz[3][3]

Atribuindo valores

```
programa
{
    inteiro matriz[1][1]

    funcao inicio()
    {

        matriz[0][0] = 10
        //matriz[1][1] = 20
        escreva("Valor: " + matriz[0][0])
    }
}
```

Como percorrer uma matriz

Para "varrer" uma matriz é parecido com o vetor sendo que na matriz possuímos duas dimensões então precisaremos de duas repetições, uma para os índices das linhas e outro para as colunas.

```
programa
{
     inteiro matriz[3][2] = {{22,10},
                             {40,10},
                             {19,30}}

     funcao inicio()
     {
          para(inteiro i=0; i <= 2; i++){
            para(inteiro j=0; j <=1; j++){
               //escreva("["+i+","+j +"] ",matriz[i][j],"\t")
               escreva(matriz[i][j])
               se (j==0){
                    escreva(",")
               }
             }
          escreva("\n")
          }
     }
}
```

Como ler valores do teclado para uma matriz

```
programa
{

    funcao inicio()
    {
        cadeia matriz[2][2]

        para(inteiro linha=0; linha <2; linha++){
            para(inteiro coluna =0; coluna < 2; coluna++){
                escreva("Digite o nome:")
                leia(matriz[linha][coluna])
            }
        }

        para(inteiro linha=0; linha <2; linha++){
            para(inteiro coluna =0; coluna < 2; coluna++){
                escreva(matriz[linha][coluna], "   ")
            }
            escreva("\n")
        }
    }
}
```

Criar uma algoritmo com uma matriz 2x3, leia nome, telefone e email e imprima no console

```
programa
{

    funcao inicio()
    {
        cadeia matriz [2][3]

        para(inteiro i=0; i <=1; i++){
            para(inteiro j=0; j <=2; j++){
                escreva("Leia os dados ["+i+"]" + "[" +j+ "]:" )
                leia(matriz[i][j])
            }
        }
        limpa()
        para(inteiro i=0; i <=1; i++){
            para(inteiro j=0; j <=2; j++){
                escreva(matriz[i][j],"\t")
            }
        escreva("\n")
        }

    }
}
```

Criar um algoritmo que leia uma matrizes 3x2. Em seguida, exiba a soma de todos elementos da matriz

```
programa
{

    funcao inicio()
    {
        inteiro matriz[3][2], soma=0

        para(inteiro  i=0; i < 3 ; i++){
          para(inteiro j=0; j < 2; j++){
             escreva("Digite os valores para linha e coluna:")
             leia(matriz[i][j])
             soma = soma + matriz[i][j]
          }
        }
        escreva("A soma da matriz é:", soma)
    }
}
```

Criar um algoritmo que leia uma matrizes 3x2. Em seguida, exiba a soma dos elementos de **cada uma das linhas** e no final exiba o total da matriz.

total por linha

```
programa
{

    funcao inicio()
    {
        inteiro matriz[3][2], soma=0

        para(inteiro  i=0; i < 3 ; i++){
            soma =0
          para(inteiro j=0; j < 2; j++){
            escreva("Digite os valores para linha e coluna:")
            leia(matriz[i][j])
            soma = soma + matriz[i][j]
          }
        escreva("A soma da linha "+i+ " é:", soma,"\n")
        }
    }
}
```

total por linha e total geral

```
programa
{

    funcao inicio()
    {
        inteiro matriz[3][2], soma=0, totalMatriz=0

        para(inteiro  i=0; i < 3 ; i++){
            soma =0
          para(inteiro j=0; j < 2; j++){
            escreva("Digite os valores para linha e colu
            leia(matriz[i][j])
            soma = soma + matriz[i][j]
          }
        totalMatriz += soma
        escreva("A soma da linha "+i+ " é:", soma,"\n")
        }
        escreva("O total da matriz é:", totalMatriz)
    }
}
```

Faça um programa leia uma matriz 3x2 e imprima ela de forma transposta

Uma matriz transposta é a transformação do que é linha para coluna como resultado.

```
programa
{
    funcao inicio()
    {
        inteiro matriz[3][2]

        para(inteiro i=0; i <=2; i++){
            para(inteiro j=0; j<=1; j++){
                escreva("Leia os valores da matriz:")
                leia(matriz[i][j])
            }
        }
        limpa()
        para(inteiro i=0; i <=2; i++){
            para(inteiro j=0; j<=1; j++){
                escreva(matriz[i][j], " ")
            }
        escreva("\n")
        }

        escreva("----Matriz Transposta----\n")
        para(inteiro j=0; j <=1; j++){
            para(inteiro i=0; i<=2; i++){
                escreva(matriz[i][j], " ")
            }
        escreva("\n")
        }
    }
}
```

Exercícios

1) Escreva um programa que leia 10 números inteiros do teclado e armazena no vetor. Após isso, imprima os 10 inteiros em ordem inversa ao que foi digitado.

2) Preencher uma matriz[4][2] com valores iniciais e fazer uma rotina para o usuário ler um número e exibir uma mensagem se este número existe na matriz, se existir pedir ao usuário para preencher com outro valor.

3) Fazer um algoritmo com um matriz 3x2 de inteiros. Preencher a matriz fazendo a leitura no console. Fazer o somatório dos valores das linhas das colunas da matriz, exibindo na tela os resultados.

4) Crie um algoritmo informe qual o maior e qual o menor elemento existente em uma matriz 4x3.