**Ordenando Matrizes**

**Mário Leite**

...

Como eu sempre afirmei aqui, e nunca é demais relembrar: o que faz um programador se tornar um ótimo profissional é a sua capacidade de *pensar/planejar/implementar* um programa seguindo um raciocínio lógico abstrato, porém objetivo, e não a sua habilidade de codificação em uma linguagem de programação. Como já publiquei, em tom de brincadeira: “*A codificação é apenas um avatar da programação*”. O que eu quis dizer com esta frase é que, embora a codificação seja a etapa final do processo, a solução do problema é dada na etapa da programação, pela lógica do algoritmo que a implementa.

Pois bem: no meu mais recente livro sobre programação, mostro muitos exemplos baseados na Teoria dos Números e em particular, nas matrizes, pois, é um assunto fundamentais; principalmente em segurança de dados; e um exemplo é sobre ordenação de matrizes. A ordenação de vetores é bem mais tranquila, com o programador sempre podendo optar pelo Método da Bolha (*Bubble Sort*), por ser o mais fácil de implementar. Mas, para matrizes a ordenação o processo é mais complicado pois, na verdade, uma matriz bidimensional (como tratado na maioria dos casos) é um arranjo retangular de vetores na horizontal que determinam as linhas da matriz.

Para resolver esse problema criei o programa **“MatrizOrdenada”** em pseudocódigo, para atender a todos os programadores; cada um pode codificá-lo na linguagem de sua preferência.

A **figura 1** mostra uma saída do programa.

-----------------------------------------------------------------------------------------------------------------------------------

**Programa** "**MatrizOrdenada**"

//Lê uma matriz mxn de inteiros e a ordena.

//Em Pseudocódigo

//Autor: Mário Leite

//-------------------------------------------------------------------------------

**Const** MAXLIN=10 //limita as dimensões da matriz

MAXCOL=10

**Declare** MatNum: **array**[MAXLIN,MAXCOL] **de** **inteiro**

VetNum: **array**[100] **de** **inteiro**

i, j, k, Aux, NLin, NCol, TamMat: **inteiro**

**Início**

{Lê os elementos da matriz}

**Repita**

**Escreva**("Digite o número de linhas da matriz [min 2 - max",MAXLIN,"]: ")

**Leia**(NLin)

NLin ← **Int**(NLin)

**Escreva**("Digite o número de colunas da matriz [min 2 - max",MAXCOL,"]: ")

**Leia**(NCol)

NCol ← **Int**(NCol)

**EscrevaLn**("")

**AtéQue**((NLin<=MAXLIN) **e** (NCol<=MAXCOL))

TamMat ← NLin\*NCol //tamanho da matriz

**EscrevaLn**("")

**Para** i **De** 1 **Até** NLin **Faça**

**Para** j **De** 1 **Até** NCol **Faça**

**Repita** //valida o elemento da matriz

**Escreva**("Digite o elemento[",i,j,"] da matriz [0 a 99]: ")

**Leia**(MatNum[i,j])

MatNum[i,j] ← **Abs**(**Int**(MatNum[i,j])) //apenas inteiro positivo

**AtéQue**(MatNum[i,j]<=99)

**FimPara**

**FimPara**

{Cria um vetor auxiliar com todos os elementos da matriz}

k ← 0 //contador de elementos do vetor

**Para** i **De** 1 **Até** NLin **Faça**

**Para** j **De** 1 **Até** NCol **Faça**

k ← k + 1

VetNum[k] ← MatNum[i,j]

**FimPara**

**FimPara**

{Ordena o vetor auxiliar}

**Para** i **De** 1 **Até** (TamMat-1) **Faça**

**Para** j **De** (i+1) **Até** TamMat **Faça**

**Se**(VetNum[i] > VetNum[j]) **Então**

Aux ← VetNum[i]

VetNum[i] ← VetNum[j]

VetNum[j] ← Aux

**FimSe**

**FimPara**

**FimPara**

**LimPaTela**

{Exibe a matriz originalmente lida}

**EscrevaLn(**"Matriz lida:")

**Para** i **De** 1 **Até** NLin **Faça**

**Para** j **De** 1 **Até** NCol **Faça**

**Escreva**(MatNum[i,j], " ")

**FimPara**

**EscrevaLn**("")

**FimPara**

**EscrevaLn**("")

**EscrevaLn**("")

{Exibe a matriz ordenada}

**EscrevaLn(**"Matriz ordenada:")

k ← 0

**Para** i **De** 1 **Até** NLin **Faça**

**Para** j **De** 1 **Até** NCol **Faça**

k ← k + 1

MatNum[i,j] ← VetNum[k]

**Escreva**(MatNum[i,j], " ")

**FimPara**

**EscrevaLn**("")

**FimPara**

**FimPrograma //fim do programa "MatrizOrdenada"**

