

OBJETIVOS DO CAPÍTULO

- Conceitos de: conjunto multidimensional (matriz), ciclo implícito, ciclo duplo
- Comandos do FORTRAN: DIMENSION(:, :), MATMUL

10.1 programa10a.f90

- 1) No Fortran, **criar um projeto** com o nome **programa10**
- 2) No Fortran, **criar e inserir** no projeto o programa-fonte **programa10a.f90**
- 3) No Fortran, **copiar** exatamente o texto em vermelho mostrado na **Tabela 10.1**.
- 4) Objetivos do programa:
 - a) Aplicar os comandos DIMENSION, ALLOCATABLE e ALLOCATE a um conjunto bidimensional (matriz) do tipo inteiro
 - b) Ler os elementos da matriz e escrevê-los utilizando ciclos duplos (novo conceito)
- 5) Comentários sobre o programa:
 - a) Neste programa os comandos DIMENSION, ALLOCATABLE e ALLOCATE são aplicados a uma variável do tipo conjunto bidimensional, que também é chamada de matriz. A sintaxe deles, isto é, a forma de utilizá-los é mostrada na Tabela 10.2.
 - b) Em FORTRAN, conjunto bidimensional é um conjunto no qual são necessários dois índices para localizar cada elemento em sua estrutura. Portanto, os conjuntos empregados no capítulo 9 podem ser classificados como conjuntos unidimensionais pois basta um único índice para localizar cada elemento dentro deles.
 - c) Generalizando, existem os conjuntos multidimensionais, isto é, aqueles nos quais são necessários diversos índices (2, 3, etc) para localizar cada elemento dentro deles. Em FORTRAN, os conjuntos podem ter até sete dimensões ou índices. Um exemplo de conjunto tridimensional é aquele onde se armazenam as coordenadas X, Y e Z de um sólido tridimensional.
 - d) Todos os conceitos vistos no capítulo 9 para conjuntos unidimensionais também são válidos para conjuntos multidimensionais.
 - e) O comando DIMENSION(:, :) é usado para definir a dimensão de uma variável do tipo conjunto bidimensional. O par de dois pontos indica que a dimensão ainda será especificada no programa.
 - f) Uma das principais aplicações de conjuntos uni e bidimensionais é a resolução de sistemas de equações algébricas representados por

$$[A][X] = [B] \quad (10.1)$$

onde $[A]$ é a matriz (conjunto bidimensional) dos coeficientes do sistema de equações; $[X]$ é o vetor (conjunto unidimensional) incógnita, isto é, representa as variáveis a determinar com a resolução do sistema; e $[B]$ é o vetor (conjunto unidimensional) independente, ou seja, ele contém valores conhecidos.

Tabela 10.1 Programa10a.f90.

```
INTEGER LINHA, LINHAS, COLUNA, COLUNAS
INTEGER, ALLOCATABLE, DIMENSION(:, :) :: MATRIZ

WRITE(*,*) "Entre com o numero de linhas da matriz"
READ(*,*) LINHAS
WRITE(*,*) "Entre com o numero de colunas da matriz"
READ(*,*) COLUNAS

ALLOCATE ( MATRIZ ( LINHAS, COLUNAS ) )

DO LINHA = 1, LINHAS
    WRITE(*,*) "Entre com os valores dos elementos da matriz da linha =", LINHA
    DO COLUNA = 1, COLUNAS
        WRITE(*,*) "Entre com o valor do elemento da matriz da coluna =", COLUNA
        READ(*,*) MATRIZ(LINHA,COLUNA)
    END DO
END DO

WRITE(*,*) "Escrita da MATRIZ sem formato"
WRITE(*,*) MATRIZ

WRITE(*,*) "Escrita da MATRIZ na mesma sequencia dos dados"
DO LINHA = 1, LINHAS
    DO COLUNA = 1, COLUNAS
        WRITE(*,1) LINHA, COLUNA, MATRIZ(LINHA,COLUNA)
        1 FORMAT( 3X, "MATRIZ(", I1, ", ", I1, ") = ", I5)
    END DO
END DO

END
```

- g) Na linha `INTEGER, ALLOCATABLE, DIMENSION(:, :) :: MATRIZ` do programa, define-se a variável chamada MATRIZ com as seguintes características: do tipo conjunto bidimensional (matriz), devido ao comando DIMENSION(:,:); do tipo alocável, devido ao comando ALLOCATABLE; e do tipo inteiro, ou seja, cada elemento da variável MATRIZ poderá conter um número inteiro, devido ao comando INTEGER.

- h) Na linha `ALLOCATE (MATRIZ (LINHAS, COLUNAS))` do programa, utilizando-se o comando `ALLOCATE` e as variáveis `LINHAS` e `COLUNAS`, que são dados do programa, define-se, respectivamente, quantas linhas e colunas a variável `MATRIZ` terá.

Tabela 10.2 Sintaxe de comandos para variáveis do tipo matriz (conjunto bidimensional).

PARA DEFINIR O TIPO DE VARIÁVEL:

```
REAL, ALLOCATABLE, DIMENSION(:, :) :: A, B
INTEGER, ALLOCATABLE, DIMENSION(:, :) :: C, D
CHARACTER(X), ALLOCATABLE, DIMENSION(:, :) :: E, F
```

PARA ALOCAR A MEMÓRIA DOS CONJUNTOS:

```
ALLOCATE ( A(L1,C1), B(L2,C2), C(L3,C3), D(L4,C4), E(L5,C5), F(L6,C6) )
```

onde `X` é um valor inteiro que define o número de caracteres; `A`, `B`, `C`, `D`, `E` e `F` são conjuntos bidimensionais (matrizes); e `L1 ... L6`, `C1 ... C6` são variáveis do tipo inteiro que definem a quantidade de elementos de cada conjunto, respectivamente, em suas linhas e colunas, com seus valores tendo sido definidos previamente.

Os nomes de todas estas variáveis podem ser quaisquer, bem como a quantidade de variáveis em cada tipo.

- i) Cada elemento de uma variável do tipo conjunto bidimensional é referenciado por dois números inteiros, chamados de índices, que correspondem à ordem deles dentro do conjunto. O primeiro índice corresponde à linha que o elemento ocupa dentro da matriz e o segundo, à coluna. Estes índices devem ficar dentro de parênteses, e separados por vírgula, após o nome da variável do tipo conjunto bidimensional. Os índices ou os números do elemento podem ser representados por variáveis, por exemplo, na linha

`READ(*,*) MATRIZ(LINHA,COLUNA)` do programa, que está dentro de um ciclo duplo.

- j) Ciclo duplo é quando se tem um ciclo dentro de outro. São usados dois ciclos duplos no programa 10a.f90.

Um deles é

```
DO LINHA = 1, LINHAS
  DO COLUNA = 1, COLUNAS
    WRITE(*,1) LINHA, COLUNA, MATRIZ(LINHA,COLUNA)
    1 FORMAT( 3X, "MATRIZ(", I1, ",", I1, ") = ", I5)
  END DO
END DO
```

- k) Recomenda-se indentar, isto é, deslocar para a direita todas as linhas do programa que estão entre os comandos `DO` e `END DO`. Assim, facilmente se percebe os comandos que compõem o ciclo. No caso de ciclos duplos, deve-se usar uma indentação diferente para cada ciclo, conforme o exemplo mostrado no item anterior.

- l) São usadas duas formas para escrever os valores dos elementos de uma variável do tipo conjunto bidimensional. A primeira é agir da mesma forma que no caso de uma variável simples, por exemplo, a linha `WRITE(*,*) MATRIZ` do programa. Nesta forma a matriz é escrita no formato default do FORTRAN que é: primeiro, escrever os valores de todos os elementos da primeira coluna, indo da primeira à última linha; repetir o procedimento para a segunda coluna, e assim por diante até a última coluna. A segunda forma é escrever o valor de cada elemento através de um ciclo duplo, com ou sem formato de edição. Um exemplo disso é apresentado no final do programa10a.f90
- 6) Algoritmo do programa:
- Definir os tipos de todas as variáveis
 - Ler a quantidade de elementos em cada linha e coluna da variável MATRIZ, que é do tipo conjunto bidimensional
 - Alocar a memória para a variável MATRIZ
 - Ler os valores de todos os elementos da variável MATRIZ
 - Escrever os valores da variável MATRIZ sem utilizar formato de edição
 - Escrever os dois índices e o valor correspondente a cada elemento da variável MATRIZ utilizando formatos de edição
- 7) Executar **Build, Compile** para compilar o programa.
- 8) Gerar o programa-executável fazendo **Build, Build**.
- 9) Ao se executar o programa, através de **Build, Execute**, surge uma janela, mostrada na Figura 10.1, dentro da qual tem-se:
- Na primeira linha, o comentário “Entre com o numero de linhas da matriz”, resultado do comando `WRITE(*,*) "Entre com o numero de linhas da matriz"` do programa.
 - Na segunda linha, o programa pára e fica aguardando até que seja fornecido o valor da variável LINHAS, resultado do comando `READ(*,*) LINHAS` do programa. Para que o programa continue a sua execução é necessário **digitar 3**, por exemplo, e, em seguida, **clicar na tecla Enter**.
 - Na terceira linha, o comentário “Entre com o numero de colunas da matriz”, resultado do comando `WRITE(*,*) "Entre com o numero de colunas da matriz"` do programa.
 - Na quarta linha, o programa pára e fica aguardando até que seja fornecido o valor da variável COLUNAS, resultado do comando `READ(*,*) COLUNAS` do programa. Para que o programa continue a sua execução é necessário **digitar 2**, por exemplo, e, em seguida, **clicar na tecla Enter**.
 - Na quinta linha, o comentário “Entre com os valores dos elementos da matriz da linha = 1”, resultado do comando `WRITE(*,*) "Entre com os valores dos elementos da matriz da linha =", LINHA` do programa.
 - Na sexta linha, o comentário “Entre com o valor do elemento da matriz da coluna = 1”, resultado do comando `WRITE(*,*) "Entre com o valor do elemento da matriz da coluna =", COLUNA` do programa.

```
C:\MSDEV\Projects\programa10\Debug\programa10.exe
Entre com o numero de linhas da matriz
3
Entre com o numero de colunas da matriz
2
Entre com os valores dos elementos da matriz da linha =      1
Entre com o valor do elemento da matriz da coluna =      1
1
Entre com o valor do elemento da matriz da coluna =      2
4
Entre com os valores dos elementos da matriz da linha =      2
Entre com o valor do elemento da matriz da coluna =      1
2
Entre com o valor do elemento da matriz da coluna =      2
5
Entre com os valores dos elementos da matriz da linha =      3
Entre com o valor do elemento da matriz da coluna =      1
3
Entre com o valor do elemento da matriz da coluna =      2
6
Escrita da MATRIZ sem formato
      1      2      3      4      5      6
Escrita da MATRIZ na mesma sequencia dos dados
MATRIZ(1,1) =      1
MATRIZ(1,2) =      4
MATRIZ(2,1) =      2
MATRIZ(2,2) =      5
MATRIZ(3,1) =      3
MATRIZ(3,2) =      6
Press any key to continue
```

Figura 10.1 Resultado do programa10a.f90.

- g) Na sétima linha, o programa pára e fica aguardando até que seja fornecido o primeiro valor da primeira linha da matriz, resultado das instruções

```
DO COLUNA = 1, COLUNAS
  WRITE(*,*) "Entre com o valor do elemento da matriz da coluna =", COLUNA
  READ(*,*) MATRIZ(LINHA,COLUNA)
END DO
```

do programa, que contém um ciclo que começa no elemento da primeira coluna e vai até a última. Deve-se perceber que o comando READ é usado para ler o valor de apenas um elemento a cada vez; assim é necessário **digitar cada valor e**, em seguida, **clicar na tecla Enter** antes de se digitar um novo valor. Usar, por exemplo, os valores 1, 4, 2, 5, 3 e 6. Estes dados, nesta seqüência, correspondem à seguinte matriz:

$$\begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} \quad (10.2)$$

h) Em seguida são apresentados os resultados correspondentes à execução do programa.

- 10) Até entender, **analisar** os resultados do programa10a.f90, mostrados na Figura 10.1, considerando cada linha do programa-fonte e as explicações descritas no item 5, acima.
- 11) **Executar** novamente o programa com outros dados. Até entender, **analisar** os novos resultados do programa10a.f90 considerando cada linha do programa-fonte e as explicações descritas no item 5, acima.

10.2 programa10b.f90

- 1) No Fortran, **criar e inserir** no projeto o programa-fonte **programa10b.f90**
- 2) No Fortran, **copiar** exatamente o texto em vermelho mostrado na **Tabela 10.3**.

Tabela 10.3 Programa10b.f90.

```
USE PORTLIB

INTEGER LINHA, LINHAS, COLUNA, COLUNAS, CONSTANTE, VER
REAL,ALLOCATABLE,DIMENSION(:, :) :: MATRIZ_A, MATRIZ_B, MATRIZ_C

WRITE(*,*) "Entre com o valor da CONSTANTE (inteiro)"
READ(*,*) CONSTANTE
WRITE(*,*) "Entre com o numero de linhas das matrizes (inteiro)"
READ(*,*) LINHAS
WRITE(*,*) "Entre com o numero de colunas das matrizes (inteiro)"
READ(*,*) COLUNAS

ALLOCATE ( MATRIZ_A(LINHAS,COLUNAS), MATRIZ_B(LINHAS,COLUNAS), &
           MATRIZ_C(LINHAS,COLUNAS) )

DO LINHA = 1, LINHAS
  WRITE(*,*) "Entre com os valores dos elementos da MATRIZ_A da linha =", LINHA
  READ(*,*) ( MATRIZ_A(LINHA,COLUNA), COLUNA = 1, COLUNAS )
END DO

DO LINHA = 1, LINHAS
  DO COLUNA = 1, COLUNAS
    MATRIZ_B(LINHA,COLUNA) = CONSTANTE * MATRIZ_A(LINHA,COLUNA)
  END DO
END DO

DO LINHA = 1, LINHAS
```

```

DO COLUNA = 1, COLUNAS
    MATRIZ_C(LINHA,COLUNA) = MATRIZ_A(LINHA,COLUNA) + MATRIZ_B(LINHA,COLUNA)
END DO
END DO

OPEN(10, FILE="SAIDA10B.TXT")

WRITE(10,31) CONSTANTE
31 FORMAT("CONSTANTE = ", I5, /)

WRITE(10,32)
32 FORMAT(3X, "**** MATRIZ_A (dados) ****")
DO LINHA = 1, LINHAS
    WRITE(10,33) ( MATRIZ_A(LINHA,COLUNA), COLUNA = 1, COLUNAS )
    33 FORMAT( 10 (3X, 1PE10.3 ) )
END DO

WRITE(10,41)
41 FORMAT(1/, 3X, "**** MATRIZ_B (CONSTANTE * MATRIZ_A) ****")
DO LINHA = 1, LINHAS
    WRITE(10,33) ( MATRIZ_B(LINHA,COLUNA), COLUNA = 1, COLUNAS )
END DO

WRITE(10,51)
51 FORMAT(1/, 3X, "**** MATRIZ_C (MATRIZ_A + MATRIZ_B) ****")
DO LINHA = 1, LINHAS
    WRITE(10,33) ( MATRIZ_C(LINHA,COLUNA), COLUNA = 1, COLUNAS )
END DO

CLOSE(10)

VER = SYSTEM("Notepad SAIDA10B.TXT")

END

```

3) Objetivos do programa:

- Aplicar os comandos DIMENSION, ALLOCATABLE e ALLOCATE a conjuntos bidimensionais (matrizes) do tipo real
- Ler os valores de elementos de matrizes, e escrevê-los em arquivo, utilizando ciclos simples e implícitos
- Realizar operações matemáticas com matrizes

4) Comentários sobre o programa:

- a) Com os ciclos usados no programa da seção anterior, programa10a.f90, consegue-se ler apenas um elemento da matriz por vez, isto é, a cada comando Enter. Já neste programa10b.f90, com o seguinte bloco de linhas do programa

```
DO LINHA = 1, LINHAS
  WRITE(*,*) "Entre com os valores dos elementos da MATRIZ_A da linha =", LINHA
  READ(*,*) ( MATRIZ_A(LINHA,COLUNA), COLUNA = 1, COLUNAS )
END DO
```

consegue-se ler os valores de todos os elementos de cada linha da MATRIZ_A. Isso é possível porque: existe um ciclo externo, definido pela linha `DO LINHA = 1, LINHAS` do programa, que a cada execução percorre uma linha da matriz; e também existe um ciclo chamado de implícito na linha

`READ(*,*) (MATRIZ_A(LINHA,COLUNA), COLUNA = 1, COLUNAS)` do programa, que a cada execução percorre todas as colunas da matriz.

- b) O chamado ciclo implícito ou aninhado é utilizado sem os comandos DO e END DO, conforme exemplificado na leitura e escrita da MATRIZ_A e na escrita da MATRIZ_B e da MATRIZ_C no programa10b.f90.

- c) Com o bloco de linhas do programa

```
DO LINHA = 1, LINHAS
  DO COLUNA = 1, COLUNAS
    MATRIZ_B(LINHA,COLUNA) = CONSTANTE * MATRIZ_A(LINHA,COLUNA)
  END DO
END DO
```

o valor de cada elemento da MATRIZ_A é multiplicado pela variável CONSTANTE e o resultado é atribuído a cada elemento correspondente da MATRIZ_B.

- d) Com o bloco de linhas do programa

```
DO LINHA = 1, LINHAS
  DO COLUNA = 1, COLUNAS
    MATRIZ_C(LINHA,COLUNA) = MATRIZ_A(LINHA,COLUNA) + MATRIZ_B(LINHA,COLUNA)
  END DO
END DO
```

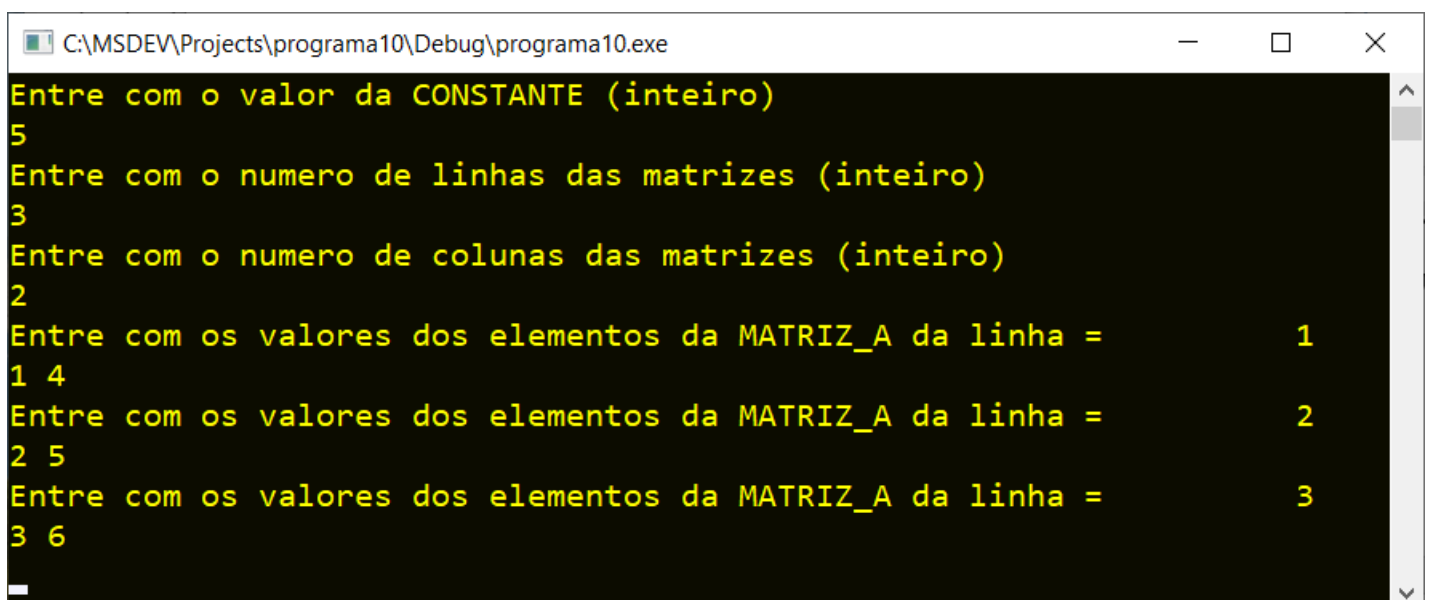
o valor de cada elemento da MATRIZ_A é adicionado ao valor de cada elemento correspondente da MATRIZ_B e o resultado é atribuído a cada elemento correspondente da MATRIZ_C.

- e) Com o bloco de linhas do programa

```
DO LINHA = 1, LINHAS
  WRITE(10,33) ( MATRIZ_A(LINHA,COLUNA), COLUNA = 1, COLUNAS )
  33 FORMAT( 10 (3X, 1PE10.3) )
END DO
```

escreve-se o conteúdo da variável MATRIZ_A em formato de matriz através de um ciclo simples (externo) e um ciclo implícito (interno). O formato de edição empregado `10 (3X, 1PE10.3)` permite que sejam escritos até 10 elementos a cada linha com três colunas vazias entre cada um (3X), utilizando o formato 1PE10.3 para números reais. Esse número (10) é um exemplo. Se o “colunas” <10, não há problema; mas, se > 10, o excedente será escrito na linha seguinte.

- f) Com conjuntos multidimensionais, também se pode fazer simultaneamente operações com todos os elementos, conforme já visto no programa9c.f90 do capítulo 9 para conjuntos unidimensionais. Por exemplo, as cinco linhas de programa mostradas no item c, acima, podem ser substituídas de forma equivalente a apenas **MATRIZ_B = CONSTANTE * MATRIZ_A**
- E as cinco linhas de programa mostradas no item d, acima, podem ser substituídas de forma equivalente a apenas **MATRIZ_C = MATRIZ_A + MATRIZ_B**
- g) Com conjuntos multidimensionais, também podem ser utilizadas as funções intrínsecas das Tabelas 5.5 e 5.6, conforme já visto no programa9c.f90 do capítulo 9 para conjuntos unidimensionais.
- 5) Algoritmo do programa:
- Definir os tipos de todas as variáveis
 - Ler o valor do tipo inteiro de uma constante e a quantidade de elementos em cada linha e coluna das variáveis MATRIZ_A, MATRIZ_B e MATRIZ_C, que são do tipo conjunto bidimensional
 - Alocar memória para as três variáveis do tipo matriz
 - Ler os valores de todos os elementos da variável MATRIZ_A no formato de matriz
 - Calcular o valor de cada elemento da MATRIZ_B através da multiplicação do valor de cada elemento da MATRIZ_A pela variável CONSTANTE
 - Calcular o valor de cada elemento da MATRIZ_C através da adição do valor de cada elemento da MATRIZ_A ao valor de cada elemento da MATRIZ_B
 - Escrever em arquivo o valor da CONSTANTE e o conteúdo das variáveis MATRIZ_A, MATRIZ_B e MATRIZ_C em formato de matriz
- 6) Executar **Build, Compile** para compilar o programa.
- 7) Gerar o programa-executável fazendo **Build, Build**.
- 8) Ao se executar o programa, através de **Build, Execute**, surge uma janela, mostrada na Figura 10.2, dentro da qual tem-se:



```
C:\MSDEV\Projects\programa10\Debug\programa10.exe
Entre com o valor da CONSTANTE (inteiro)
5
Entre com o numero de linhas das matrizes (inteiro)
3
Entre com o numero de colunas das matrizes (inteiro)
2
Entre com os valores dos elementos da MATRIZ_A da linha = 1
1 4
Entre com os valores dos elementos da MATRIZ_A da linha = 2
2 5
Entre com os valores dos elementos da MATRIZ_A da linha = 3
3 6
```

Figura 10.2 Janela DOS do programa10b.f90.

- a) Na primeira linha, o comentário “Entre com o valor da CONSTANTE (inteiro)”, resultado do comando `WRITE(*,*) "Entre com o valor da CONSTANTE (inteiro)"` do programa.
- b) Na segunda linha, o programa pára e fica aguardando até que seja fornecido o valor da variável CONSTANTE, resultado do comando `READ(*,*) CONSTANTE` do programa. Para que o programa continue a sua execução é necessário **digitar 5**, por exemplo, e, em seguida, **clicar na tecla Enter**.
- c) Na terceira linha, o comentário “Entre com o numero de linhas das matrizes (inteiro)”, resultado do comando `WRITE(*,*) "Entre com o numero de linhas das matrizes (inteiro)"` do programa.
- d) Na quarta linha, o programa pára e fica aguardando até que seja fornecido o valor da variável LINHAS, resultado do comando `READ(*,*) LINHAS` do programa. Para que o programa continue a sua execução é necessário **digitar 3**, por exemplo, e, em seguida, **clicar na tecla Enter**.
- e) Na quinta linha, o comentário “Entre com o numero de colunas das matrizes (inteiro)”, resultado do comando `WRITE(*,*) "Entre com o numero de colunas das matrizes (inteiro)"` do programa.
- f) Na sexta linha, o programa pára e fica aguardando até que seja fornecido o valor da variável COLUNAS, resultado do comando `READ(*,*) COLUNAS` do programa. Para que o programa continue a sua execução é necessário **digitar 2**, por exemplo, e, em seguida, **clicar na tecla Enter**.
- g) Na sétima linha, o comentário “Entre com os valores dos elementos da MATRIZ_A da linha = 1”, resultado do comando `WRITE(*,*) "Entre com os valores dos elementos da MATRIZ_A da linha =", LINHA` do programa.
- h) Na oitava linha, o programa pára e fica aguardando até que sejam fornecidos os valores de todos os elementos da primeira linha da matriz; **digitar 1 4**, por exemplo, e, em seguida, **clicar na tecla Enter**. Para a segunda linha da matriz, **digitar 2 5**, por exemplo, e, em seguida, **clicar na tecla Enter**. Para a terceira linha da matriz, **digitar 3 6**, por exemplo, e, em seguida, **clicar na tecla Enter**. Estes dados, nesta sequência, correspondem à seguinte matriz:

$$\begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} \quad (10.3)$$

- i) Em seguida, o aplicativo Notepad apresenta os resultados correspondentes à execução do programa no arquivo SAIDA10B.TXT, mostrados na Figura 10.3.
- 9) Até entender, **analisar** os resultados do programa10b.f90, mostrados na Figura 10.3, considerando cada linha do programa-fonte e as explicações descritas no item 4, acima.
- 10) **Executar** novamente o programa com outros dados. Até entender, **analisar** os novos resultados do programa10b.f90 considerando cada linha do programa-fonte e as explicações descritas no item 4, acima.

```
SAIDA10B - Bloco de Notas
Arquivo  Editar  Formatar  Exibir  Ajuda
CONSTANTE = 5

*** MATRIZ_A (dados) ***
1.000E+00  4.000E+00
2.000E+00  5.000E+00
3.000E+00  6.000E+00

*** MATRIZ_B (CONSTANTE * MATRIZ_A) ***
5.000E+00  2.000E+01
1.000E+01  2.500E+01
1.500E+01  3.000E+01

*** MATRIZ_C (MATRIZ_A + MATRIZ_B) ***
6.000E+00  2.400E+01
1.200E+01  3.000E+01
1.800E+01  3.600E+01

Ln 1, Col 1  100%  Windows (CRLF)  UTF-8
```

Figura 10.3 Arquivo SAIDA10B.TXT do programa10b.f90.

10.3 EXERCÍCIOS

Exercício 10.1

Editar um programa-fonte em FORTRAN para executar o seguinte algoritmo (passos):

- 1) Ler o valor de uma constante
- 2) Ler os valores reais da Matriz_A
- 3) Dividir pela constante o valor de cada elemento da Matriz_A e atribuir o resultado à Matriz_B
- 4) Escrever em arquivo a Matriz_B identificando os índices de cada elemento

Exercício 10.2

Editar um programa-fonte em FORTRAN para executar o seguinte algoritmo (passos):

- 1) Ler os valores reais da Matriz_A = $\begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$
- 2) Calcular a exponencial do valor de cada elemento da Matriz_A e atribuir o resultado à Matriz_B
- 3) Escrever em arquivo a Matriz_B identificando os índices de cada elemento

Exercício 10.3

Editar um programa-fonte em FORTRAN para calcular o produto de duas matrizes $[A] \times [B]$, executando o seguinte algoritmo (passos):

- 1) Ler os seguintes dados (suficientes para resolver o problema):
 - a) número de linhas da Matriz_A
 - b) número de colunas da Matriz_A
 - c) número de colunas da Matriz_B
 - d) Valores dos elementos da Matriz_A
 - e) Valores dos elementos da Matriz_B
- 2) Calcular o valor de cada elemento da Matriz_C que resulta do produto da Matriz_A pela Matriz_B; para fazer isso, sugere-se analisar o produto literal entre uma matriz 2×2 com outra 2×1 ; depois, uma 3×2 com outra 2×2 ; deduzir o algoritmo e implementá-lo
- 3) Escrever em arquivo todos os dados lidos e a Matriz_C identificando todas as variáveis e elementos
- 4) Calcular a Matriz_D através da seguinte linha inserida dentro do programa-fonte:

$\text{Matriz_D} = \text{MATMUL} (\text{Matriz_A}, \text{Matriz_B})$

onde o comando MATMUL é uma função intrínseca do FORTRAN que calcula o produto de duas matrizes

- 5) Escrever em arquivo a Matriz_D identificando seus elementos; ela deve ser idêntica à Matriz_C; se a Matriz_C for diferente da Matriz_D, há erro no seu produto de matriz

Observações:

- a) A Matriz_C terá a dimensão do número de linhas da Matriz_A pelo número de colunas da Matriz_B.
- b) O número de linhas da Matriz_B tem que ser igual ao número de colunas da Matriz_A.
- c) O produto envolve o uso de 3 ciclos, um dentro do outro.

Exercício 10.4

- 1) Criar um projeto com o nome **Cap10** e inserir nele o programa-fonte **Cap10.f90**
- 2) Criar o arquivo de saída **Cap10.TXT** e escrever nele o nome completo do aluno.
- 3) Ler o número de linhas e de colunas da Matriz_A
- 4) Ler os valores da Matriz_A
- 5) Calcular o quadrado do valor de cada elemento da Matriz_A e atribuir o resultado à Matriz_B
- 6) Escrever no arquivo de saída a Matriz_B em formato de matriz
- 7) Abrir o arquivo de saída com o aplicativo Bloco de Notas (Notepad).

8) Executar o programa com $\text{Matriz_A} = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$ Resultados esperados: $\text{Matriz_B} = \begin{bmatrix} 1 & 16 \\ 4 & 25 \\ 9 & 36 \end{bmatrix}$

Exercício 10.5

- 1) Criar um projeto com o nome **Cap10b** e inserir nele o programa-fonte **Cap10b.f90**
- 2) Criar o arquivo de saída **Cap10b.TXT** e escrever nele o nome completo do aluno.
- 3) Ler o número de linhas e de colunas da Matriz_A
- 4) Ler os valores reais da Matriz_A
- 5) Calcular o logaritmo decimal do valor de cada elemento da Matriz_A e atribuir o resultado à Matriz_B
- 6) Escrever no arquivo de saída a Matriz_B em formato de matriz
- 7) Abrir o arquivo de saída com o aplicativo Bloco de Notas (Notepad).

8) Executar o programa com $\text{Matriz_A} = \begin{bmatrix} 1 & 10 \\ 5 & 50 \\ 10 & 100 \end{bmatrix}$ Resultados esperados: $\text{Matriz_B} = \begin{bmatrix} 0 & 1 \\ 0.70 & 1.70 \\ 1 & 2 \end{bmatrix}$