

OBJETIVOS DO CAPÍTULO

- Conceitos de: sub-rotina, programa-principal, projeto com diversos programas-fonte, passagem de argumentos
- Comandos do FORTRAN: PROGRAM, EXTERNAL, SUBROUTINE, CALL, CONTAINS

12.1 programa12a

Para inicializar as atividades deste capítulo, deve-se executar:

- 1) Para acessar o programa Fortran, no Windows: **Start, Programs, Fortran PowerStation 4.0, Microsoft Developer Studio**
- 2) No Fortran, seguindo o [procedimento-padrão](#), **criar um projeto** com o nome **programa12a**
- 3) No Fortran, seguindo o [procedimento-padrão](#), **criar e inserir** no projeto o programa-fonte **principal1.f90**
- 4) Dentro do espaço de edição do Fortran, na subjanela maior, **copiar** exatamente o texto em vermelho mostrado na **Tabela 12.1**.
- 5) No Fortran, seguindo o [procedimento-padrão](#), **criar e inserir** no projeto o programa-fonte **rotina1.f90**
- 6) Dentro do espaço de edição do Fortran, na subjanela maior, **copiar** exatamente o texto em vermelho mostrado na **Tabela 12.2**.
- 7) Objetivos do programa:
 - a) Aplicar os novos comandos PROGRAM, EXTERNAL, SUBROUTINE e CALL do FORTRAN
 - b) Utilizar uma sub-rotina externa ao programa-principal
- 8) Comentários sobre o programa:
 - a) Neste programa são usados quatro novos comandos do FORTRAN: PROGRAM, EXTERNAL, SUBROUTINE e CALL. Eles são aplicados com o que se denomina em programação de sub-rotina.
 - b) Sub-rotinas são programas inseridos ou usados por um programa-principal.
 - c) O programa-principal é o programa-fonte que contém o algoritmo que se deseja executar e que usa as sub-rotinas.
 - d) As sub-rotinas podem estar contidas dentro do programa-principal ou podem estar dentro de outros programas-fonte. O primeiro caso será exemplificado na seção 12.4 deste capítulo. Já o segundo caso é exemplificado na presente seção e nas duas seguintes. Neste caso, isto é, quando

há mais de um programa-fonte que constitui um projeto, usa-se o comando PROGRAM para definir qual programa-fonte é o programa-principal. E o comando EXTERNAL é usado para informar ao programa-principal quais são os outros programas-fonte que fazem parte do projeto. Outra forma de usar sub-rotinas é utilizar o comando USE para incluir uma biblioteca dentro do programa-principal, para que este possa chamar as sub-rotinas desejadas da biblioteca.

Tabela 12.1 Programa-fonte principal1.f90.

```
PROGRAM CAPITULO_12A

EXTERNAL ROTINA1

REAL A, B, C

WRITE(*,*) "Entre com o valor de A"
READ(*,*) A

WRITE(*,*) "Entre com o valor de B"
READ(*,*) B

CALL SOMA ( A, B, C )

WRITE(*,*) "A + B = ", C

END PROGRAM CAPITULO_12A
```

- e) Quando há mais de um programa-fonte no projeto, primeiro deve-se compilar os programas-fonte que contêm sub-rotinas. O programa-principal deve ser compilado por último.
- f) Cada sub-rotina pode depender de variáveis do programa-principal ou pode ser um programa totalmente independente. Neste caso, com poucas adaptações, um programa já existente pode ser transformado em uma sub-rotina de um outro programa.
- g) A principal vantagem de se usar sub-rotinas é dividir um programa muito grande ou complexo em unidades menores, ou subprogramas, que são mais fáceis de implementar e que facilitam a detecção de erros.
- h) Cada sub-rotina deve ter um nome específico, que é definido com o comando SUBROUTINE. Este nome é usado para chamar ou usar cada sub-rotina no local desejado. As sub-rotinas são ativadas ou chamadas através do comando CALL.
- i) As sub-rotinas podem ser escritas pelo próprio programador ou por outros programadores, na forma de programas-fonte ou bibliotecas.

- j) Na linha **PROGRAM CAPITULO_12A**, do programa-fonte principal1.f90, define-se o início e o nome do programa-principal como sendo CAPITULO_12A. E na última linha, com o comando **END PROGRAM CAPITULO_12A**, define-se o fim do programa-principal. O nome do programa não tem utilidade prática nenhuma. Deve-se perceber que ele é diferente do nome do projeto e do nome do programa-fonte.

Tabela 12.2 Programa-fonte rotina1.f90.

```
SUBROUTINE SOMA ( X, Y, Z )  
  
REAL X, Y, Z  
  
Z = X + Y  
  
END SUBROUTINE SOMA
```

- k) Na linha **EXTERNAL ROTINA1**, do programa-fonte principal1.f90, declara-se que o programa-fonte rotina1.f90 faz parte do programa-fonte principal1.f90. Deve-se notar que no comando **EXTERNAL** é declarado o nome do programa-fonte que contém a sub-rotina, e não o nome da sub-rotina.
- l) Na linha **CALL SOMA (A, B, C)**, do programa-fonte principal1.f90, chama-se a sub-rotina de nome SOMA e transfere-se a ela os valores das variáveis que estão entre parênteses, isto é, as variáveis A, B e C, que foram declaradas como variáveis reais no programa-principal.
- m) Na linha **SUBROUTINE SOMA (X, Y, Z)**, do programa-fonte rotina1.f90, define-se o início e o nome da sub-rotina como sendo SOMA e, ainda, quais as variáveis que são recebidas e devolvidas ao programa-principal, no caso as variáveis X, Y e Z. E na última linha, com o comando **END SUBROUTINE SOMA**, define-se o fim da sub-rotina SOMA.
- n) As variáveis de uma sub-rotina que são recebidas e devolvidas ao programa-principal são denominadas de argumentos da sub-rotina. Elas têm que ser do mesmo tipo das variáveis usadas no programa-principal que chama a sub-rotina, mas não precisam ter o mesmo nome. Não é obrigatório que as sub-rotinas tenham argumentos.
- 9) Algoritmo do programa:
- No programa-principal, declarar que o programa-fonte rotina1.f90 faz parte do programa-principal
 - No programa-principal, definir as variáveis A, B e C como sendo do tipo real
 - No programa-principal, ler o valor das variáveis A e B
 - No programa-principal, chamar a sub-rotina SOMA, transferindo a ela os valores atuais das variáveis A, B e C

- e) Na sub-rotina SOMA, receber os valores das variáveis X, Y e Z, transferidos do programa-principal
 - f) Na sub-rotina SOMA, realizar a soma das variáveis X e Y e atribuir o resultado à variável Z
 - g) Na sub-rotina SOMA, ao encontrar o fim da sub-rotina, voltar ao programa-principal no ponto onde a sub-rotina foi chamada, transferindo os valores atuais das variáveis X, Y e Z, da sub-rotina, para as variáveis A, B e C do programa-principal
 - h) No programa-principal, escrever o valor atual da variável C
 - i) No programa-principal, encerrar a execução do programa
- 10) Executar **Build, Compile** para compilar o programa-fonte rotinal.f90.
 - 11) Executar **Build, Compile** para compilar o programa-fonte principal.f90.
 - 12) Gerar o programa-executável fazendo **Build, Build**.
 - 13) Ao se executar o programa, através de **Build, Execute**, surge uma janela, mostrada na Figura 12.1, dentro da qual tem-se:

```

C:\marchi\graduacao\2004_1\TM_102\p...
Entre com o valor de A
1
Entre com o valor de B
2
A + B = 3.000000
Press any key to continue

```

Figura 12.1 Resultado do programa12a.

- a) Na primeira linha, o comentário Entre com o valor de A, resultado do comando **WRITE(*,*) "Entre com o valor de A"** do programa.
- b) Na segunda linha, o programa pára e fica aguardando que seja fornecido o valor da variável A, resultado do comando **READ(*,*) A** do programa. Para que o programa continue sua execução é necessário **digitar 1**, por exemplo, e, em seguida, **clicar na tecla Enter**.
- c) Na terceira linha, o comentário Entre com o valor de B, resultado do comando **WRITE(*,*) "Entre com o valor de B"** do programa.
- d) Na quarta linha, o programa pára e fica aguardando que seja fornecido o valor da variável B, resultado do comando **READ(*,*) B** do programa. Para que o programa continue sua execução é necessário **digitar 2**, por exemplo, e, em seguida, **clicar na tecla Enter**.
- e) Na quinta linha, é apresentado o resultado da soma das variáveis A e B, calculado dentro da sub-rotina SOMA.

- 14) Até entender, **analisar** os resultados do programa12a, mostrados na Figura 12.1, considerando cada linha dos dois programas-fonte envolvidos e as explicações descritas nos itens 8 e 9 acima.
- 15) **Executar** novamente o programa com outros dados.
- 16) No Fortran, para fechar o projeto atual, executar **File, Close Workspace**

12.2 programa12b

- 1) No Fortran, seguindo o [procedimento-padrão](#), **criar um projeto** com o nome **programa12b**
- 2) No Fortran, seguindo o [procedimento-padrão](#), **criar e inserir** no projeto o programa-fonte **principal2.f90**
- 3) Dentro do espaço de edição do Fortran, na subjanela maior, **copiar** exatamente o texto em vermelho mostrado na **Tabela 12.3**.

Tabela 12.3 Programa-fonte principal2.f90.

```
PROGRAM CAPITULO_12B

EXTERNAL ROTINAS2

INTEGER INTEIRO
REAL A, B, C

WRITE(*,*) "Entre com o valor de A"
READ(*,*) A

WRITE(*,*) "Entre com o valor de B"
READ(*,*) B

CALL SOMA ( A, B, C )

WRITE(*,*) "A + B = ", C

WRITE(*,*) "Entre com um valor inteiro para calcular seu fatorial"
READ(*,*) INTEIRO

CALL FATORIAL ( INTEIRO )

END PROGRAM CAPITULO_12B
```

- 4) No Fortran, seguindo o [procedimento-padrão](#), **criar e inserir** no projeto o programa-fonte **rotinas2.f90**
- 5) Dentro do espaço de edição do Fortran, na subjanela maior, **copiar** exatamente o texto em vermelho mostrado na **Tabela 12.4**.

Tabela 12.4 Programa-fonte rotinas2.f90.

```
SUBROUTINE SOMA ( X, Y, Z )

REAL X, Y, Z

Z = X + Y

END SUBROUTINE SOMA

SUBROUTINE FATORIAL ( N )

INTEGER I, N, FAT

FAT = 1

IF ( N < 0 ) THEN
    WRITE(*,*) "Nao existe fatorial de ", N
ELSE
    DO I = 2, N
        FAT = FAT * I
    END DO
END IF

WRITE(*,*) "O fatorial de", N, " eh = ", FAT

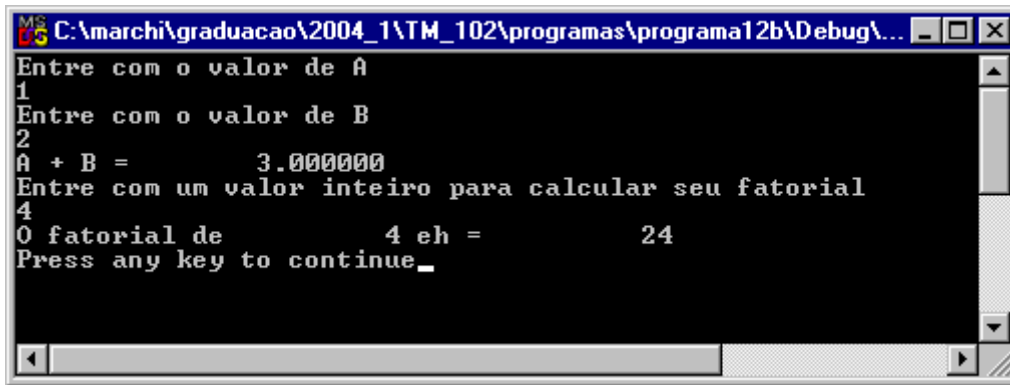
END SUBROUTINE FATORIAL
```

- 6) Objetivos do programa:
 - a) Implementar um programa-fonte com duas sub-rotinas
 - b) Utilizar duas sub-rotinas externas ao programa-principal
- 7) Comentários sobre o programa:
 - a) Um programa-fonte pode ser constituído por uma ou várias sub-rotinas. Um exemplo é o programa-fonte rotinas2.f90 que contém duas sub-rotinas.

- b) Na linha **CALL FATORIAL (INTEIRO)**, do programa-fonte principal2.f90, chama-se a sub-rotina de nome FATORIAL e transfere-se a ela o valor da variável que está entre parênteses, isto é, a variável INTEIRO, que foi declarada como variável do tipo inteiro no programa-principal.
- 8) Algoritmo do programa:
- a) No programa-principal, declarar que o programa-fonte rotinas2.f90 faz parte do programa-principal
 - b) No programa-principal, definir as variáveis A, B e C como sendo do tipo real e a variável INTEIRO como do tipo inteiro
 - c) No programa-principal, ler o valor das variáveis A e B
 - d) No programa-principal, chamar a sub-rotina SOMA, transferindo a ela os valores atuais das variáveis A, B e C
 - e) Na sub-rotina SOMA, receber os valores das variáveis X, Y e Z, transferidos do programa-principal
 - f) Na sub-rotina SOMA, realizar a soma das variáveis X e Y e atribuir o resultado à variável Z
 - g) Na sub-rotina SOMA, ao encontrar o fim da sub-rotina, voltar ao programa-principal no ponto onde a sub-rotina foi chamada, transferindo os valores atuais das variáveis X, Y e Z, da sub-rotina, para as variáveis A, B e C do programa-principal
 - h) No programa-principal, escrever o valor atual da variável C
 - i) No programa-principal, ler o valor da variável INTEIRO
 - j) No programa-principal, chamar a sub-rotina FATORIAL, transferindo a ela o valor atual da variável INTEIRO
 - k) Na sub-rotina FATORIAL, receber o valor da variável N, transferido do programa-principal
 - l) Na sub-rotina FATORIAL, realizar o cálculo do fatorial da variável N e atribuir o resultado à variável FAT
 - m) Na sub-rotina FATORIAL, escrever o valor da variável FAT
 - n) Na sub-rotina FATORIAL, ao encontrar o fim da sub-rotina, voltar ao programa-principal no ponto onde a sub-rotina foi chamada, transferindo o valor atual da variável N, da sub-rotina, para a variável INTEIRO do programa-principal
 - o) No programa-principal, encerrar a execução do programa
- 9) Executar **Build, Compile** para compilar o programa-fonte rotinas2.f90.
- 10) Executar **Build, Compile** para compilar o programa-fonte principal2.f90.
- 11) Gerar o programa-executável fazendo **Build, Build**.
- 12) Ao se executar o programa, através de **Build, Execute**, obtém-se o resultado mostrado na Figura 12.2.
- 13) Até entender, **analisar** os resultados do programa12b, mostrados na Figura 12.2, considerando cada linha dos dois programas-fonte envolvidos e as explicações descritas nos itens 7 e 8 acima.

14) **Executar** novamente o programa com outros dados.

15) No Fortran, para fechar o projeto atual, executar **File, Close Workspace**



```
MS-DOS C:\marchi\graduacao\2004_1\TM_102\programas\programa12b\Debug\...
Entre com o valor de A
1
Entre com o valor de B
2
A + B =          3.000000
Entre com um valor inteiro para calcular seu fatorial
4
O fatorial de          4 eh =          24
Press any key to continue_
```

Figura 12.2 Resultado do programa12b.

12.3 programa12c

- 1) No Fortran, seguindo o [procedimento-padrão](#), **criar um projeto** com o nome **programa12c**
- 2) No Fortran, seguindo o [procedimento-padrão](#), **criar e inserir** no projeto o programa-fonte **principal3.f90**
- 3) Dentro do espaço de edição do Fortran, na subjanela maior, **copiar** exatamente o texto em vermelho mostrado na **Tabela 12.5**.

Tabela 12.5 Programa-fonte principal3.f90.

```
PROGRAM CAPITULO_12C

EXTERNAL ROTINA3

CALL OUTRAS

END PROGRAM CAPITULO_12C
```

- 4) No Fortran, seguindo o [procedimento-padrão](#), **criar e inserir** no projeto o programa-fonte **rotinas2.f90**
- 5) Dentro do espaço de edição do Fortran, na subjanela maior, **copiar** exatamente o texto em vermelho mostrado na **Tabela 12.4**. O conteúdo do arquivo rotinas2.f90 também pode ser copiado diretamente do projeto anterior, programa12b.
- 6) No Fortran, seguindo o [procedimento-padrão](#), **criar e inserir** no projeto o programa-fonte **rotina3.f90**

- 7) Dentro do espaço de edição do Fortran, na subjanela maior, **copiar** exatamente o texto em vermelho mostrado na **Tabela 12.6**.
- 8) Objetivos do programa:
- Implementar um programa com três sub-rotinas divididas em dois programas-fonte
 - Fazer uma sub-rotina chamar outras sub-rotinas
 - Utilizar uma sub-rotina sem argumentos

Tabela 12.6 Programa-fonte rotina3.f90.

```
SUBROUTINE OUTRAS

EXTERNAL ROTINAS2

INTEGER INTEIRO
REAL A, B, C

WRITE(*,*) "Entre com o valor de A"
READ(*,*) A

WRITE(*,*) "Entre com o valor de B"
READ(*,*) B

CALL SOMA ( A, B, C )

WRITE(*,*) "A + B = ", C

WRITE(*,*) "Entre com um valor inteiro para calcular seu fatorial"
READ(*,*) INTEIRO

CALL FATORIAL ( INTEIRO )

END SUBROUTINE OUTRAS
```

- 9) Comentários sobre o programa:
- O programa12c faz exatamente o mesmo que o programa12b, da seção anterior. A diferença é que tudo o que antes era feito no programa-principal agora é feito pela sub-rotina OUTRAS, que é chamada pelo programa-principal cuja única função dele é essa chamada.
 - Uma sub-rotina pode chamar uma ou diversas sub-rotinas. Por exemplo, a sub-rotina OUTRAS chama as sub-rotinas SOMA e FATORIAL.
- 10) Algoritmo do programa: é o mesmo da seção 12.2, item 8.

- 11) Executar **Build, Compile** para compilar o programa-fonte rotinas2.f90.
- 12) Executar **Build, Compile** para compilar o programa-fonte rotina3.f90.
- 13) Executar **Build, Compile** para compilar o programa-fonte principal3.f90.
- 14) Gerar o programa-executável fazendo **Build, Build**.
- 15) Ao se executar o programa, através de **Build, Execute**, obtém-se o resultado já mostrado na Figura 12.2.
- 16) Até entender, **analisar** os resultados do programa12c, mostrados na Figura 12.2, considerando cada linha dos três programas-fonte envolvidos e as explicações pertinentes.
- 17) No Fortran, para fechar o projeto atual, executar **File, Close Workspace**

12.4 programa12d

- 1) No Fortran, seguindo o [procedimento-padrão](#), **criar um projeto** com o nome **programa12d**
- 2) No Fortran, seguindo o [procedimento-padrão](#), **criar e inserir** no projeto o programa-fonte **principal4.f90**
- 3) Dentro do espaço de edição do Fortran, na subjanela maior, **copiar** exatamente o texto em vermelho mostrado na **Tabela 12.7**. Deve-se perceber que quase todo o conteúdo do arquivo principal4.f90 já foi digitado no projeto programa12c e, portanto, ele pode ser copiado dos arquivos rotinas2.f90 e rotina3.f90.
- 4) Objetivos do programa:
 - a) Implementar um programa com sub-rotinas inseridas dentro do programa-principal
 - b) Utilizar sub-rotina de biblioteca
- 5) Comentários sobre o programa:
 - a) O programa12d é praticamente idêntico ao programa12c, da seção anterior. Há apenas duas diferenças. A primeira é que as três sub-rotinas (OUTRAS, SOMA e FATORIAL) que antes estavam dentro de dois programas-fonte (rotinas2.f90 e rotina3.f90) agora estão inseridas dentro do próprio programa-principal.
 - b) O comando CONTAINS do FORTRAN é usado para separar o fim do programa-principal do início das sub-rotinas contidas dentro do programa-principal, conforme pode-se ver na Tabela 12.7.
 - c) A segunda diferença é que, após a chamada da sub-rotina OUTRAS, foi inserido uma chamada da sub-rotina TDATE, que é uma sub-rotina pertencente à biblioteca MSIMSLMS. E, para utilizar esta biblioteca, no início do programa-principal foi empregado o comando USE junto com o nome da biblioteca.

- d) A biblioteca MSIMSLMS contém muitas sub-rotinas com diversas finalidades. A lista completa das sub-rotinas desta biblioteca, informações detalhadas e exemplos sobre cada uma delas podem ser vistos no manual online do Fortran em: ? InfoView, IMSL Libraries Reference.
- 6) Algoritmo do programa: é o mesmo da seção 12.2, item 8, acrescido ao final da chamada da sub-rotina TDATE e da escrita da data corrente.

Tabela 12.7 Programa-fonte principal4.f90.

```
PROGRAM CAPITULO_12D

USE MSIMSLMS

INTEGER DIA, MES, ANO

CALL OUTRAS

CALL TDATE ( DIA, MES, ANO )

WRITE(*,1) DIA, MES, ANO
1 FORMAT (/, 5X, "Data de hoje eh ", I2, "/", I2, "/", I4)

CONTAINS

! -----

SUBROUTINE OUTRAS

INTEGER INTEIRO
REAL A, B, C

WRITE(*,*) "Entre com o valor de A"
READ(*,*) A

WRITE(*,*) "Entre com o valor de B"
READ(*,*) B

CALL SOMA ( A, B, C )

WRITE(*,*) "A + B = ", C
```

```

WRITE(*,*) "Entre com um valor inteiro para calcular seu fatorial"
READ(*,*) INTEIRO

CALL FATORIAL ( INTEIRO )

END SUBROUTINE OUTRAS

! -----

SUBROUTINE SOMA ( X, Y, Z )

REAL X, Y, Z

Z = X + Y

END SUBROUTINE SOMA

! -----

SUBROUTINE FATORIAL ( N )

INTEGER I, N, FAT

FAT = 1

IF ( N < 0 ) THEN
    WRITE(*,*) "Nao existe fatorial de ", N
ELSE
    DO I = 2, N
        FAT = FAT * I
    END DO
END IF

WRITE(*,*) "O fatorial de", N, " eh = ", FAT

END SUBROUTINE FATORIAL

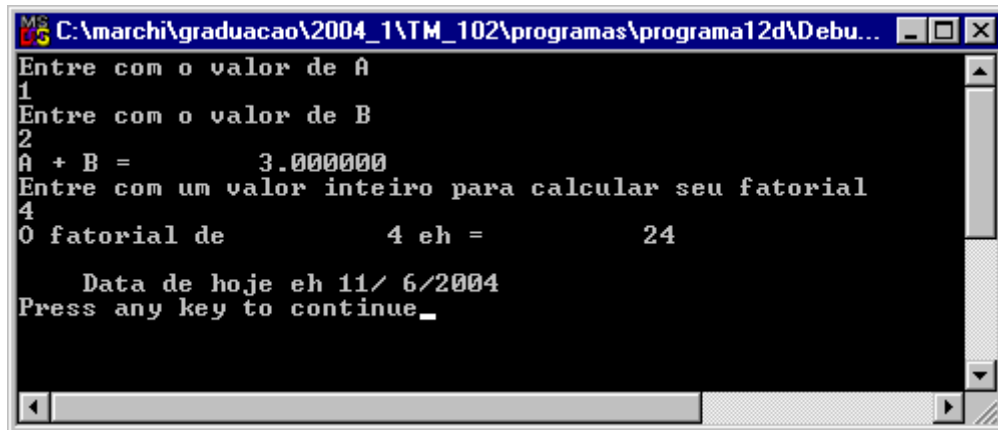
! -----

END PROGRAM CAPITULO_12D

```

7) Executar **Build, Compile** para compilar o programa-fonte principal4.f90.

- 8) Gerar o programa-executável fazendo **Build, Build**.
- 9) Ao se executar o programa, através de **Build, Execute**, obtém-se o resultado mostrado na Figura 12.3.
- 10) Até entender, **analisar** os resultados do programa12d, mostrados na Figura 12.3, considerando cada linha do programa-fonte e as explicações pertinentes.
- 11) Encerrar a sessão seguindo o [procedimento-padrão](#).



```
MS-DOS Batch File: C:\marchi\graduacao\2004_1\TM_102\programas\programa12d\Debu...
Entre com o valor de A
1
Entre com o valor de B
2
A + B = 3.000000
Entre com um valor inteiro para calcular seu fatorial
4
O fatorial de 4 eh = 24

Data de hoje eh 11/ 6/2004
Press any key to continue_
```

Figura 12.3 Resultado do programa12d.

12.5 EXERCÍCIO

Exercício 12.1

Alterar o programa12d, da seção 12.4, da seguinte forma:

- 1) Adaptar a sub-rotina FATORIAL e sua chamada na sub-rotina OUTRAS para que o valor do fatorial seja passado a sub-rotina OUTRAS
- 2) Implementar a sub-rotina ESCREVE para escrever num arquivo os resultados das variáveis A, B, C, INTEIRO e fatorial. Esta nova sub-rotina deverá ser chamada dentro da sub-rotina OUTRAS.
- 3) Implementar a sub-rotina ARQUIVO para mostrar, com o aplicativo Notepad, o conteúdo do arquivo criado pela sub-rotina ESCREVE. A sub-rotina ARQUIVO deverá ser chamada pela sub-rotina OUTRAS, após a chamada da sub-rotina ESCREVE.