ALVARO F. M. AZEVEDO; ALÍPIO E. V. FERREIRA; RICARDO M. P. SANTOS

Normalização da Escrita de Programas em Fortran 77

Relatório Técnico
Faculdade de Engenharia da Universidade do Porto
Portugal
Abril de 1989

NORMALIZAÇÃO DA ESCRITA DE PROGRAMAS EM FORTRAN 77

Álvaro Azevedo , Alípio Ferreira e Ricardo Santos
Faculdade de Engenharia da Universidade do Porto

1989/04/10

O objectivo das presentes sugestões é o da uniformização do modo de escrever programas em Fortran 77, procurando aproximar esta linguagem dos princípios da programação estruturada.

Como em tudo, haverá vantagens e inconvenientes, sendo no entanto indiscutível que as vantagens valem o investimento de algumas horas na reestruturação dos programas, sendo estas recuperadas mais tarde, quando se tornar necessário modificar ou reutilizar alguns módulos.

As principais vantagens da adopção das presentes recomendações são as seguintes:

- construir uma biblioteca de ferramentas compatíveis com diversas aplicações.
- aumentar a modularização facilitando o desenvolvimento e o debug de grandes aplicações.
- facilitar a leitura de programas escritos por outros programadores.
- aumentar a portabilidade das aplicações para outros sistemas operativos e outras linguagens.
- adoptar ficheiros com uma estrutura bem definida, que permita que os resultados de uma aplicação sejam facilmente utilizados como dados de outra aplicação distinta.

O principal inconveniente da adopção da programação estruturada é o de uma ligeira perda de eficiência dos programas, que na maior parte dos casos não é significativa.

Segue-se a descrição detalhada das recomendações propostas:

1 - UTILIZAÇÃO DE MAIÚSCULAS OU MINÚSCULAS

Fortran 77 sob MSDOS,OS2,AOS OU VMS -> MAIÚSCULAS

Fortran 77 sob UNIX -> minúsculas

Notas: A utilização de maiúsculas aconselha-se na maior parte dos casos porque é tradicional em Fortran e porque embora possa haver dificuldade em distinguir um O de um 0, parece-nos ser mais problemática a confusão do 1 com o l. A excepção deve-se à difícil utilização dos editores do UNIX em maiúsculas.

2 - TIPOS DE VARIÁVEIS POR DEFEITO

Em Fortran, quando o tipo de uma variável não é declarado, ela é considerada inteira se o primeiro caracter pertencer ao intervalo (I-N) e real se pertencer aos intervalos (A-H,O-Z). Esta convenção está já muito enraizada nos hábitos dos programadores em Fortran, tendo a vantagem de permitir uma rápida identificação do tipo de uma variável sem consultar a lista das declarações. Esta convenção apresenta também uma boa concordância com a nomenclatura habitual da bibliografia técnica. Por estes motivos se aconselha o seguimento desta convenção em qualquer linguagem.

Sugere-se ainda que os inteiros sejam por defeito de 4 bytes e os reais de 8, devendo-se incluir à cabeça de qualquer módulo as seguintes linhas:

```
IMPLICIT INTEGER*4 (I-N)
IMPLICIT REAL*8    (A-H,O-Z)
```

3 - VARIÁVEIS, SUBROTINAS, FUNÇÕES, PROGRAMAS E FICHEIROS

Uma vez que alguns compiladores de Fortran ainda apresentam o limite de 6 caracteres no nome dos identificadores, e uma vez que o MSDOS tem como limite para o nome dos ficheiros 8.3 caracteres, sugerem-se as seguintes convenções:

Variáveis - 5 caracteres

Subrotinas - 6 caracteres

Funções - 6 caracteres

Programas - 8 ou menos caracteres

Ficheiros - 8.3 ou menos caracteres

Excepções:

No vector único, atribuir aos apontadores para arrays o nome do array precedido de um L (Ex: LCOORD) e designar o próprio vector único por V (Ex: V(LCOORD)).

Quando uma variável é referida muitas vezes na mesma linha e é local em relação a um bloco pequeno, é vantajoso atribuir-lhe um menor número de caracteres.

(Ex: F(X,Y)=X*SIN (X**2+Y**3)+SQRT (LOG(X/(X+1))))

Convém resistir à tentação de escrever DO 10 I=1,N. Escrever antes DO 10 ISTEP=1,NSTEP. Este procedimento torna possível uma eventual substituiçáo global do nome de uma variável com o editor, o que não é possível se se utilizarem variáveis de uma letra.

Notas:

Nas variáveis e funções, o primeiro caracter deve obedecer às características dos tipos por defeito (inteiro,real) já referidos.

Todos as atribuições de nomes devem ter como base palavras em inglês e/ou as respectivas abreviaturas.

Deve-se procurar sempre uma lógica na atribuição de nomes a variáveis que representem entidades análogas.

Ex: NNVAB - Number of Nodal VAriaBles

NEVAB - Number of Element VAriaBles

NGVAB - Number of Global VAriaBles

Evitar a coincidência com palavras reservadas, mesmo de outras linguagens ou sistemas operativos.

Ex: Evitar FOPEN (linguagem C), MKDIR (UNIX), UNTIL (PASCAL), LLIST (BASIC), etc.

Na atribuição de nomes a ficheiros sugerem-se as duas seguintes construções:

8.3 - 8 ou menos caracteres no nome base (job name)

- 3 ou menos caracteres no sufixo

Ex: PORTICO1.DAT ; ASNA.RES ; PROG.C

5_2.3 - 5 ou menos caracteres no nome base (job name)

- 2 ou menos caracteres no primeiro sufixo que indica qual o conteudo do ficheiro

- 3 ou menos caracteres no segundo sufixo que indica o tipo de organização do ficheiro

Ex: PORT1_DA.S3D ; ASNA_DI.LPT

Exemplos do primeiro sufixo:

DA - DAta

DI - DIsplacements

ST - STresses

RE - REactions

ER - ERrors

PS - Principal Stresses

MX - Moments mX

ME - MEsh

LO - LOad

Exemplos do segundo sufixo:

BIN - BINário ou não formatado

LPT - preparado para o Line PrinTer (não editável)

EDI - ficheiro EDItável com qualquer editor de texto

S3D - formato relativo aos programas TRIDIM, HIDLINE, etc.

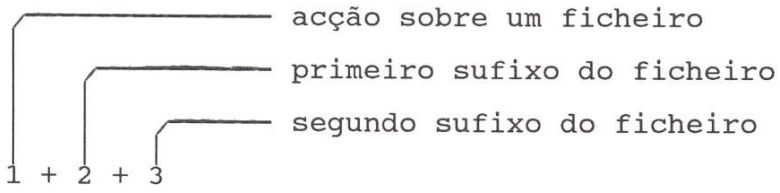
G2D - formato relativo ao programa GRAFICO

F77 - source code em Fortran 77

FOR - source code em Fortran 77

C - source code em C

Na atribuição de nomes a subrotinas sugerem-se dois tipos de construção:



Exemplos de acções:

A - Assemble

C - Calculate/Compute

D - Draw

E - End

I - Initialize

R - Read

S - Solve

V - Verify/Validate

W - Write

Z - Zero

Ex: RLNODS - Read List of NODal connectionS

VPARAM - Verify PARAMeters

WDABIN - Write file jobname_DA.BIN

WFIS3D - Write FIle jobname.S3D

Na atribuição de nomes a funções, o primeiro caracter deve respeitar as características dos tipos por defeito (inteiro e real), constituindo os restantes um nome sugestivo.

4 - CONSIDERAÇÕES DIVERSAS

- O diálogo com o utilizador deve ser em inglês.

- Os comentários intercalados no source code podem ser em português embora fosse preferível também em inglês.

- Colocar comentários a preceder cada módulo, cada bloco de instruções e cada chamada de subrotina.

- Sempre que tal se justificar, explicar a lógica da atribuição de um nome do seguinte modo:

RLOPAR - Read LOad PARameters

NEVAB - Number of Element VAriaBles

- As constantes reais devem ser explicitadas como nos seguintes exemplos: RVALU=5.0D+15 ou RVALU=0.0D+0 ou RVALU=1.0D-2

- Todos os ciclos DO, estruturas IF-THEN-ELSE-ENDIF, etc. devem ser indentadas de 3 em 3 espaços (ver programas exemplo).

- Utilizar a instrução GOTO apenas na simulação de estruturas que não existem em Fortran ("while" e "repeat-until") e no corte da execução de uma estrutura com passagem da execução para a instrução imediatamente a seguir a esse bloco de instruções ("break"). É também tolerada a utilização do GOTO na saída de um módulo quando ocorre uma situação de erro ("exit").

- Uma vez que em Fortran o GOTO só pode ser dirigido a um "label" numérico, este deve estar colocado numa instrução "CONTINUE".

- Todos os "label" das instruções CONTINUE devem estar por ordem numérica crescente.

- Evitar a instrução FORMAT recorrendo à colocação dos formatos na própria instrução WRITE.

- Evitar a instrução INCLUDE que torna a listagem de mais difícil leitura e obriga a transportar para outras aplicações definições não necessárias quando se pretende utilizar o mesmo módulo em diferentes programas.

- Evitar a instrução COMMON porque não permite o dimensionamento variável, torna os ficheiros executáveis gigantescos, e não permite utilizar a totalidade da memória no MSDOS.

- Colocar entre espaços as constantes lógicas (.TRUE. e .FALSE.) e os operadores lógicos (.OR. , .GE. , .NOT. , etc.).

- Quando houver uma linha que seja a continuação da anterior, deixar o separador (".AND." , "OR" , "," , "+" , "-" , "*" , "/") na linha anterior e usar o "." na 6. coluna da 2. linha.

- As instruções READ devem ser efectuadas em formato livre como no exemplo: READ (201,*) RVALU . Exceptua-se o caso da leitura de variáveis tipo CHARACTER que deve ser feita em formato (A) como no exemplo: READ (201,'(A)') TLINE .

- As instruções WRITE devem ser sempre formatadas por excesso, sendo de evitar o formato livre (*).

- Evitar a utilização do canal (*), que pode ter um funcionamento diferente de compilador para compilador. Abrir sempre canais, mesmo para leitura e escrita na consola, com números entre 20 e 255.

- Não utilizar caracteres com código ASCII acima de 127 porque não estão standardizados e comprometem a portabilidade do software. (Ex: á, ç, caixilhos, etc.)

- Nunca tocar na tecla "TAB" porque é interpretada de um modo diferente de sistema para sistema.

- Colocar uma subrotina ou função em cada ficheiro.

- Colocar os ficheiros relacionados com cada aplicação em directórios distintos para evitar colisões nos nomes dos módulos e para não ter directórios muito cheios.

- Colocar os módulos que forem comuns a varias aplicações num directório à parte.

- Manter uma lista actualizada dos módulos que é necessário linkar para obter a versão executável de uma determinada aplicação (ficheiro progname.FIL).

- Colocar validações e mensagens de erro sugestivas sempre que uma situação de erro possa ocorrer, quer em fase de desenvolvimento, quer em fase de utilização do programa.

5 - EXECUÇÃO INTERACTIVA OU EM BACKGROUND (BATCH)

Se se pretender que o mesmo programa executável possa correr interactivamente ou em background (batch), deve-se proceder do seguinte modo:

- prever a existência de um ficheiro com nome fixo (_STDIN) que contém na primeira linha o jobname e na segunda Y ou N conforme se pretende ou não o diálogo interactivo.

- Se o diálogo não for interactivo (N na segunda linha), todas as leituras interactivas são efectuadas no ficheiro jobname.STI, que tem de ser previamente preparado, e todas as mensagens são enviadas para o ficheiro jobname.STO. Deste modo o processo pode correr em background (batch) desde que no ficheiro jobname.STI já se encontrem todas as respostas às perguntas interactivas. O eco das leituras deve ser enviado para o ficheiro jobname.STO.

- Se o ficheiro _STDIN não existir ou se existir e tiver um Y na segunda linha, o diálogo é interactivo, i e, todas as leituras

são efectuadas no teclado e as mensagens são enviadas para o monitor .

6 - LISTAGENS

Apresentam-se em seguida um conjunto de listagens de cabeçalhos tipo e de programas de demonstração daquilo que foi proposto que devem ser pormenorizadamente estudadas.

Estes ficheiros encontram-se na directoria :UTIL:DEMO:STD .

Na directoria :UTIL:DEMO:TOOLS encontram-se outros ficheiros que devem ser consultados por quem tenciona seguir as presentes recomendações.

7 - NOTA FINAL

As presentes recomendações devem ser encaradas como uma tentativa de disciplinar o software existente e em fase de desenvolvimento e não devem em caso algum constituir um entrave ao aparecimento e utilização de novas ideias.

São bem vindas todas as sugestões e comentários de programadores e utilizadores de software, quer sigam ou não as presentes recomendações.

Ficheiro -> :UTIL:DEMO:STD:PROGDEMO.F77

```
      PROGRAM PROGDEMO
C
C*****************************************************************
C
C *** PROGram DEMOstration
C *** DEMONSTRACAO COM VARIAS LINHAS DE PROGRAMACAO STANDARD.
C *** FICHEIROS UTILIZADOS:
C        jobname.DAT - FICHEIRO DE DADOS
C        .......... - ...
C *** SIGNIFICADO DAS VARIAVEIS:
C        NELEM - NUMERO DE ELEMENTOS
C        ..... - ...
C
C     VERSION: 1.0
C     DATE   : 1989-04-05
C     AUTHOR : ALVARO AZEVEDO, ALIPIO FERREIRA E RICARDO SANTOS
C
C*****************************************************************
C
      IMPLICIT INTEGER*4 (I-N)
      IMPLICIT REAL*8    (A-H,O-Z)
C
      CHARACTER*1 CCAUX
C
      CHARACTER*80 FNAME
      CHARACTER*80 JNAME
C ***
      PARAMETER (MELEM=1000)
      PARAMETER (MNODE=1000)
      PARAMETER (MTOTV=MELEM*MNODE)
C ***
      INTEGER*4 LNODS(MELEM,MNODE)
C
      REAL*8 COORD(MELEM,MNODE)
      REAL*8 POINT(MTOTV)
C
      LOGICAL*1 LOKAY(MELEM)
C
C *** INICIALIZACAO DE ALGUMAS VARIAVEIS
C
      TOLER=1.0D-15
C
C *** LER O NOME BASE DOS FICHEIROS
C
      WRITE (*,'(A)') ' JOB NAME ? '
      READ  (*,'(A)') JNAME
      LENGJ=LENGTH (JNAME)
C
C *** ABRIR O FICHEIRO DE DADOS
C
      FNAME=JNAME(1:LENGJ)//'.DAT'
      OPEN (50,FILE=FNAME,STATUS='OLD',PAD='YES')
C
C *** CICLO "DO"
C
      DO 120 IELEM=1,NELEM
         LNODS(IELEM,INODE)=23
         DO 110 INODE=1,8
            LNODS(IELEM,INODE)=LNODS(IELEM,INODE)+
```

```
                                    INODE
  110    CONTINUE
  120 CONTINUE
C
C *** FECHAR O FICHEIRO DE DADOS
C
      CLOSE (50)
C
C *** BLOCO "IF-THEN-ELSEIF-ELSE-ENDIF"
C
      IF ((ITOTV .GT. 10 .AND. IELEM .EQ. 5) .OR.
     .           (IELEM .LT. 100 .AND. .NOT. LOKAY(1))) THEN
         WRITE (*,'(A)')
         WRITE (*,'(A)') ' OLA'
      ELSEIF (ITOTV .GT. NTOTV) THEN
         WRITE (*,'(1X,F30.15)') COORD(IPOIN,IDIME)
      ELSE
         WRITE (*,'(1X,I5,5X,I5)') IELEM,LNODS(IELEM,INODE)
      ENDIF
C
C *** BLOCO "REPEAT-UNTIL"
C
      IELEM=0
  200 CONTINUE
         IELEM=IELEM+1
         WRITE (*,'(I5)') LNODS(IELEM,INODE)
      IF (IELEM .LT. NELEM) GOTO 200
C
C *** BLOCO "REPEAT-UNTIL" COM "BREAK"
C
      IELEM=0
  300 CONTINUE
         IELEM=IELEM+1
         IF (LOKAY(10)) GOTO 400
         WRITE (*,'(I5)') LNODS(IELEM,INODE)
      IF (IELEM .LT. NELEM) GOTO 300
  400 CONTINUE
C
C *** CICLO INFINITO COM "BREAK(S)"
C
      IELEM=0
  500 CONTINUE
         IF (IELEM .EQ. NELEM) GOTO 600
         IELEM=IELEM+1
C
C ***    TESTAR A CONDICAO DE SAIDA
C
         IF (LOKAY(IELEM)) GOTO 600
         WRITE (*,'(I5)') LNODS(IELEM,INODE)
      GOTO 500
  600 CONTINUE
C
C *** BLOCO "SWITCH: CASE"
C
      IF (CCAUX .EQ. 'A') THEN
         WRITE (*,'(A)') ' SELECCAO = A'
      ELSEIF (CCAUX .EQ. 'B') THEN
         WRITE (*,'(A)') ' SELECCAO = B'
      ELSEIF (CCAUX .EQ. 'C') THEN
         WRITE (*,'(A)') ' SELECCAO = C'
      ELSE
```

```
         WRITE (*,'(A)') ' ERRO NA SELECCAO'
      ENDIF
C
C *** CICLOS "NESTED"
C
      FOUND= .FALSE.
      IELEM=0
  700 CONTINUE
         IELEM=IELEM+1
         INODE=0
  800    CONTINUE
            INODE=INODE+1
            IF (LNODS(IELEM,INODE) .EQ. LNODE) FOUND= .TRUE.
         IF (INODE .LT. NNODE  .AND. .NOT. FOUND) GOTO 800
      IF (IELEM .LT. NELEM  .AND. .NOT. FOUND) GOTO 700
C
C *** ABRIR O FICHEIRO DE RESULTADOS
C
      FNAME=JNAME(1:LENGJ)//'.RES'
      OPEN (60,FILE=FNAME,STATUS='FRESH')
C
C *** ESCREVER OS RESULTADOS
C
      CALL WRESUL (A2345,B2345)
C
C *** FECHAR O FICHEIRO DE RESULTADOS
C
      CLOSE (60)
C
C *** ABRIR, LER E FECHAR UM FICHEIRO NAO FORMATADO
C
      FNAME=JNAME(1:LENGJ)//'.BIN'
      OPEN (51,FILE=FNAME,STATUS='FRESH',FORM='UNFORMATTED',
     .         MODE='BINARY',RECFM='DYNAMIC')
      READ (51) NELEM
      CLOSE (51,STATUS='DELETE')
C
      END
```

```
Ficheiro -> :UTIL:DEMO:STD:PROG.H

      PROGRAM A2345678
C
C*********************************************************************
C
C *** TITULO.
C *** DESCRICAO SUMARIA.
C *** DESCRICAO PORMENORIZADA.
C *** FICHEIROS UTILIZADOS:
C        _STDIN          - ...
C        jobname_DA.S3D - ...
C *** SIGNIFICADO DAS VARIAVEIS:
C        X2345 - ...
C        Y2345 - ...
C
C     VERSION: 1.0
C     DATE   : 1989-02-16
C     AUTHOR : ALVARO AZEVEDO, ALIPIO FERREIRA E RICARDO SANTOS
C
C*********************************************************************
C
      IMPLICIT INTEGER*4 (I-N)
      IMPLICIT REAL*8    (A-H,O-Z)
C
      INTEGER*2 I1234
      INTEGER*2 J1234
      INTEGER*2 K1234
C
      REAL*4 R2345
      REAL*4 S2345
C
      CHARACTER*1 C2345
      CHARACTER*1 D2345
C
      CHARACTER*80 B2345
C
      CHARACTER*255 A2345
C
      LOGICAL*1 L2345
C ***
      PARAMETER (M2345=1000)
      PARAMETER (N2345=1000)
C ***
      INTEGER*2 I2345(M2345,N2345)
C
      INTEGER*4 I2345(M2345,N2345)
      INTEGER*4 J2345(M2345)
      INTEGER*4 K2345(M2345)
C
      REAL*4 R2345(M2345)
C
      REAL*8 R2345(M2345)
      REAL*8 S2345(M2345)
C
      CHARACTER*1 C2345(M2345)
      CHARACTER*1 D2345(M2345)
C
      CHARACTER*80 B2345(M2345)
C
      CHARACTER*255 A2345(M2345)
```

```
C
      LOGICAL*1 L2345(M2345)
C
%INCLUDE 'FINCLUDE'
C
```

```
Ficheiro -> :UTIL:DEMO:STD:SUBR.H

      SUBROUTINE A23456 (A2345,B2345,C2345,D2345,
     .                   E2345)
------  - 4 POR LINHA (MAXIMO).
------  - EM 1.LUGAR AS VARIAVEIS NAO INDEXADAS DE TIPO POR DEFEITO.
------  - SEGUINDO-SE AS RESTANTES RESPEITANDO A ORDEM DAS DECLARACOE
------  - MUDAR DE LINHA QUANDO MUDAR O TIPO DE VARIAVEL.
------  - SE O NUMERO DE PARAMETROS DE PASSAGEM NAO EXCEDER 4 OU 5,
------    E' PREFERIVEL DAR-LHES A ORDENACAO QUE PARECER MAIS LOGICA.
C
C*********************************************************************
C
C *** TITULO.
C *** DESCRICAO SUMARIA.
C *** DESCRICAO PORMENORIZADA.
C *** FICHEIROS UTILIZADOS:
C        _STDIN          - ...
C        jobname_DA.S3D - ...
C *** SIGNIFICADO DOS PARAMETROS DE PASSAGEM:
C        A2345 (IO) - ...                    (IO) significa Input Output
C        B2345 (I ) - ...
C        C2345 ( O) - ...
C *** SIGNIFICADO DAS VARIAVEIS:
C        X2345 - ...
C        Y2345 - ...
C
C     VERSION: 1.0
C     DATE   : 1989-02-16
C     AUTHOR : ALVARO AZEVEDO, ALIPIO FERREIRA E RICARDO SANTOS
C
C*********************************************************************
C
      IMPLICIT INTEGER*4 (I-N)
      IMPLICIT REAL*8    (A-H,O-Z)
C
      INTEGER*2 I1234
      INTEGER*2 J1234
      INTEGER*2 K1234
C
      REAL*4 R2345
      REAL*4 S2345
C
      CHARACTER*1 C2345
      CHARACTER*1 D2345
C
      CHARACTER*80 B2345
C
      CHARACTER*255 A2345
C
      LOGICAL*1 L2345
C ***
      PARAMETER (M2345=1000)
      PARAMETER (N2345=1000)
C ***
      INTEGER*2 I2345(M2345,N2345)
C
      INTEGER*4 I2345(M2345,N2345)
      INTEGER*4 J2345(M2345)
      INTEGER*4 K2345(M2345)
C
```

```
      REAL*4 R2345(M2345)
C
      REAL*8 R2345(M2345)
      REAL*8 S2345(M2345)
C
      CHARACTER*1 C2345(M2345)
      CHARACTER*1 D2345(M2345)
C
      CHARACTER*80 B2345(M2345)
C
      CHARACTER*255 A2345(M2345)
C
      LOGICAL*1 L2345(M2345)
C
%INCLUDE 'FINCLUDE'
C
```

Ficheiro -> :UTIL:DEMO:STD:FINCLUDE

```
C     FINCLUDE
C
C*********************************************************************
C
C *** TITULO.
C *** DESCRICAO SUMARIA.
C *** DESCRICAO PORMENORIZADA.
C *** FICHEIROS UTILIZADOS:
C        _STDIN          - ...
C        jobname_DA.S3D - ...
C *** SIGNIFICADO DOS PARAMETROS DE PASSAGEM:
C        A2345 (IO) - ...                    (IO) significa Input Output
C        B2345 (I ) - ...
C        C2345 ( O) - ...
C *** SIGNIFICADO DAS VARIAVEIS:
C        X2345 - ...
C        Y2345 - ...
C
C     VERSION: 1.0
C     DATE   : 1989-02-16
C     AUTHOR : ALVARO AZEVEDO, ALIPIO FERREIRA E RICARDO SANTOS
C
C*********************************************************************
C
      INTEGER*2 I1234
      INTEGER*2 J1234
      INTEGER*2 K1234
C
      REAL*4 R2345
      REAL*4 S2345
C
      CHARACTER*1 C2345
      CHARACTER*1 D2345
C
      CHARACTER*80 B2345
C
      CHARACTER*255 A2345
C
      LOGICAL*1 L2345
C ***
      PARAMETER (M2345=1000)
      PARAMETER (N2345=1000)
C ***
      INTEGER*2 I2345(M2345,N2345)
C
      INTEGER*4 I2345(M2345,N2345)
      INTEGER*4 J2345(M2345)
      INTEGER*4 K2345(M2345)
C
      REAL*4 R2345(M2345)
C
      REAL*8 R2345(M2345)
      REAL*8 S2345(M2345)
C
      CHARACTER*1 C2345(M2345)
      CHARACTER*1 D2345(M2345)
C
      CHARACTER*80 B2345(M2345)
C
```

```
      CHARACTER*255 A2345(M2345)
C
      LOGICAL*1 L2345(M2345)
C
```

Ficheiro -> :UTIL:DEMO:STD:VECTDEMO.F77

```
      PROGRAM VECTDEMO
C
C*********************************************************************
C
C *** VECTor DEMOnstration
C *** PROGRAMA DE DEMONSTRACAO DA UTILIZACAO DO VECTOR UNICO.
C *** FICHEIROS UTILIZADOS:
C        _STDIN - FICHEIRO COM O JOB NAME E O KEYBI
C *** SIGNIFICADO DAS VARIAVEIS:
C        V     - VECTOR UNICO
C        LASTP - POSICAO CORRENTE DA ULTIMA POSICAO NO VECTOR
C        MAXVP - MAXIMA POSICAO POSSIVEL NO VECTOR
C        MLAST - MAXIMA POSICAO UTILIZADA NO VECTOR
C        TOBIG - INDICA QUE O PROBLEMA NAO CABE NO VECTOR
C        NBYTV - N. DE BYTES DE CADA POSICAO NO VECTOR (MIN. POSSIVEL
C        KINT2 - N. DE BYTES DE CADA INTEGER*2 / NBYTV
C        KINT4 - N. DE BYTES DE CADA INTEGER*4 / NBYTV
C        KREA4 - N. DE BYTES DE CADA REAL*4    / NBYTV
C        KREA8 - N. DE BYTES DE CADA REAL*8    / NBYTV
C        LFREE - POSICAO A QUE LASTP VAI REGRESSAR QUANDO DA LIBERTAC
C        JNAME - Job NAME
C        KEYBI - KEYBoard Input (Y/N)
C       LIFPRE - APONTADOR PARA O INICIO DO ARRAY IFPRE NO VECTOR UNI
C        NVFIX - NUMERO DE VARIAVEIS COM O SEU VALOR FIXADO 'A PARTID
C
C     VERSION: 1.0
C     DATE   : 1989-04-05
C     AUTHOR : ALVARO AZEVEDO, ALIPIO FERREIRA E RICARDO SANTOS
C
C*********************************************************************
C
      IMPLICIT INTEGER*4 (I-N)
      IMPLICIT REAL*8    (A-H,O-Z)
C
      CHARACTER*1 KEYBI
C
      CHARACTER*80 JNAME
C
      LOGICAL*1 TOBIG
C ***
      PARAMETER (MAXVP=1000000)
      PARAMETER (NBYTV=2)
      PARAMETER (KINT2=2/NBYTV)
      PARAMETER (KINT4=4/NBYTV)
      PARAMETER (KREA4=4/NBYTV)
      PARAMETER (KREA8=8/NBYTV)
C ***
      INTEGER*2 V(MAXVP)
C
C *** LER O JOB NAME DO KEYBOARD OU DO FICHEIRO _STDIN
C
      CALL RJNAME (JNAME,KEYBI)
C
C *** INICIALIZAR LASTP E MLAST
C
      LASTP=0
      MLAST=0
C
C *** DEFINIR OS APONTADORES DOS ARRAYS GLOBAIS
```

```
C
      LIFPRE=LASTP +1
      LNOFIX=LIFPRE+NVFIX*NDOFN    *KINT4
      LPRESC=LNOFIX+NVFIX          *KINT4
      LASTP =LPRESC+NVFIX*NDOFN    *KREA8 -1
C
C *** FIXAR A POSICAO A QUE LASTP VAI REGRESSAR
C
      LFREE=LASTP
C
C *** DEFINIR OS APONTADORES DOS ARRAYS A LIBERTAR
C
      LLNODS=LASTP +1
      LMATNO=LLNODS+NELEM*NNODE    *KINT4
      LCOORD=LMATNO+NELEM          *KINT4
      LPROPS=LCOORD+NPOIN*NDIME    *KREA8
      LASTP =LPROPS+NMATS*NPROP    *KREA8 -1
C
C *** VERIFICAR SE O LIMITE DO VECTOR E' EXCEDIDO E ACTUALIZAR MLAST
C
      CALL CHECKV (LASTP,MAXVP,MLAST,TOBIG)
      IF (TOBIG) GOTO 9999
C
C *** LER ALGUNS ARRAYS
C
      CALL RARRAY (V(LIFPRE),V(LLNODS),V(LMATNO),V(LNOFIX),
     .             V(LCOORD),V(LPRESC),V(LPROPS))
C
C *** LIBERTACAO DE ALGUNS ARRAYS
C
      LASTP=LFREE
C
C *** DEFINIR OS APONTADORES DOS ARRAYS QUE OCUPAM POSICOES LIBERTADA
C
      LIFFIX=LASTP +1
      LFIXED=LIFFIX+NTOTV          *KINT4
      LASTP =LFIXED+NTOTV          *KREA8 -1
C
C *** VERIFICAR SE O LIMITE DO VECTOR E' EXCEDIDO E ACTUALIZAR MLAST
C
      CALL CHECKV (LASTP,MAXVP,MLAST,TOBIG)
      IF (TOBIG) GOTO 9999
C
C *** ASSEMBLAR AS CARACTERISTICAS DOS APOIOS
C
      CALL AFIXED (V(LIFFIX),V(LIFPRE),V(LNOFIX),
     .             V(LFIXED),V(LPRESC))
C
C *** ESCREVER O NUMERO DE BYTES UTILIZADOS NO VECTOR UNICO
C
      WRITE (202,'(A)')
      WRITE (202,'(A,I10)') ' N. of bytes used in the vector = ',
     .                      NBYTV*MLAST
      WRITE (202,'(A,I10)') ' Vector dimension (bytes)       = ',
     .                      NBYTV*MAXVP
C
 9999 CONTINUE
C
      END
```

```
Ficheiro -> :UTIL:DEMO:STD:CHECKV.F77

      SUBROUTINE CHECKV (LASTP,MAXVP,MLAST,
     .                   TOBIG)
C
C*****************************************************************
C
C *** CHECK Vector
C *** VERIFICAR SE O LIMITE DO VECTOR E' EXCEDIDO E ACTUALIZAR MLAST.
C *** SIGNIFICADO DOS PARAMETROS DE PASSAGEM:
C        LASTP (I ) - POSICAO CORRENTE DA ULTIMA POSICAO NO VECTOR
C        MAXVP (I ) - MAXIMA POSICAO POSSIVEL NO VECTOR
C        MLAST (IO) - MAXIMA POSICAO UTILIZADA NO VECTOR
C        TOBIG ( O) - INDICA QUE O PROBLEMA NAO CABE NO VECTOR
C
C     VERSION: 1.0
C     DATE   : 1989-04-05
C     AUTHOR : ALVARO AZEVEDO, ALIPIO FERREIRA E RICARDO SANTOS
C
C*****************************************************************
C
      IMPLICIT INTEGER*4 (I-N)
      IMPLICIT REAL*8    (A-H,O-Z)
C
      LOGICAL*1 TOBIG
C
      TOBIG= .FALSE.
      IF (LASTP .GT. MAXVP) THEN
         TOBIG= .TRUE.
         WRITE (202,'(A)')
         WRITE (202,'(A)')      ' Problem too big to fit in the vector
         WRITE (202,'(A,I10)') '    Required MAXVP = ',LASTP
         WRITE (202,'(A,I10)') '    Allowed  MAXVP = ',MAXVP
         WRITE (202,'(A)')
      ELSE
         IF (LASTP .GT. MLAST) MLAST=LASTP
      ENDIF
C
      RETURN
      END
```