```
CONTAINS
    FUNCTION f(x)
    ...
    END FUNCTION f
END MODULE mon_module
```

Utilisation d'un module

```
PROGRAM mon_prog
    USE mon_module
    IMPLICIT NONE
    REAL :: z
    PRINT*,'Resultat␣=␣',f(z)
END PROGRAM mon_prog
```

Entrées-sorties

Écriture à l'écran

```
PRINT *,"i␣=␣",i,z
```

Formats

Instruction de format sur une ligne

```
100 FORMAT (1X,I3,2X,F8.3)
    WRITE(10,100)i*2, TAN(2.0*x)
```

Instruction d'écriture avec format

```
PRINT '(1X,I4,3X,E12.4)',i,z
```

I3, I3.3	entiers 3 et 003
F8.3	réels virgule fixe
E12.3	réels virgule flottante
A16	chaine alpha numérique
4X	espaces
100(...)	répétition

Ouverture pour un fichier

```
OPEN(10,FILE='monfichier.txt')
```

Options d'ouverture

```
IOSTAT=erreur,STATUS='OLD',STATUS='NEW'
```

Écriture dans un fichier

```
WRITE(10,*)'Bonjour␣!'
```

Options d'écriture

```
IOSTAT=erreur,ADVANCE='NO'
```

Lecture à partir d'un fichier

```
READ(10,*)x
```

Fermeture

```
CLOSE(10)
```

Compilation avec gfortran

Compilation simple, exécutable a.out (ou a.exe)

```
gfortran truc.f95
```

ou

```
f95 truc.f95
```

Compilation de plusieurs fichiers

```
gfortran truc.f95 sprog.f95
```

Compilation sans création de l'exécutable donne des fichiers objets truc.o (ou truc.obj).

```
gfortran -c truc.f95
```

Vérification des tableaux

```
gfortran truc.f95 -fcheck=bounds
```

Change le nom de l'exécutable, ici truc

```
gfortran truc.f95 -o truc
```

Optimisation de l'exécutable

```
gfortran -O3 truc.f95
```

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN

Aide-mémoire Fortran

MEUNIER-GUTTIN-CLUZEL Siegfried

19 novembre 2022

Cet aide mémoire très succinct, rappelle certaines notions utilisées en travaux dirigés. Il est très incomplet et ne reprend que des points importants. La syntaxe est simplement illustrée par des exemples.

Variables et mathématiques

Types de variables, tableaux

```
REAL :: x, y
REAL (KIND=8) :: z
REAL, DIMENSION(10) :: t
REAL, DIMENSION(1:MX,1:NX) :: mat
INTEGER :: i, j, k
CHARACTER :: ch
CHARACTER (LEN=16) :: str
COMPLEX :: cz
```

Types dérivés : réels double précision

```
INTEGER, PARAMETER :: &
   DB=SELECTED_REAL_KIND(10,100)
REAL (KIND=DB) :: x, y, z
z = 3.5_DB
x = 1.2e+5_DB
```

Type dérivé : entiers simples

```
INTEGER, PARAMETER :: &
   SI=SELECTED_INT_KIND(3)
INTEGER (KIND=SI) :: i,j,k
```

Structures, définition

```
TYPE cercle
    REAL :: x,y,r
    INTEGER :: couleur
END TYPE cercle
```

Utilisation d'une structure

```
TYPE(cercle) :: c1, c2
c1 = cercle(0.0, 0.0, 2.0, 6)
c2%couleur = 5
```

Opérateurs : addition soustraction, multiplication, division, puissance

```
+ - * / **
```

Opérateurs de comparaison

```
< > >= <= ==
```

Opérateurs logiques

```
.OR. .AND. .NOT. .EQV. .NEQV.
```

Opérateur de concaténation de chaines

```
'Abc'//'def'
```

Modulo : reste de la division de n par m

```
MOD(n,m)
```

Constante littérale complexe

```
ci = (0,1)
cid = CMPLX(0.0, 1.0, KIND=DB)
```

Nombres aléatoires réels distribués uniformément sur $[0, 1]$

```
CALL RANDOM_NUMBER(x)
```

Sections de tableau

m(:)	tout le tableau m
m(:,:)	toute la matrice
m(0:n)	de m(0) à m(n)
m(1:n:2)	m(1), m(3), m(5) ...
m	tout le tableau (à éviter)

Fonctions mathématiques

ABS(x)	$\lvert x\rvert$	TAN(x)	$\tan(x)$
COS(x)	$\cos(x)$	COSH(x)	$\cosh(x)$
EXP(x)	$\exp(x)$	SIN(x)	$\sin(x)$
LOG(x)	$\ln(x)$	LOG10(x)	$\log_{10}(x)$
SQRT(x)	$\sqrt{x}$	AIMAG(z)	$\Im(z)$
CONJ(z)	z^*	INT(x)	$\lfloor x\rfloor$

Fonctions complexes

CMPLX(x,y)	$z = x + jy$	REAL(z)	$\Re(z)$
CONJ(z)	z^*	AIMAG(z)	$\Im(z)$

Traitement des chaines de caractères

ACHAR(32)	caractère de codes ASCII 32
IACHAR('A')	code ASCII du caractère 'A'
LEN(ch)	longueur de la chaine ch
LEN_TRIM(ch)	idem, sans les espaces de fin
TRIM(ch)	supprime les espaces de fin
ch(2:6)	les caractères de 2 à 6

Tests

Test sur une ligne

```
IF (i>ifin) EXIT
```

Test sur plusieurs lignes

```
IF (i<=ifin)THEN
  i = i+1
ENDIF
```

Test si ... sinon

```
IF (i<=ifin) THEN
   i = i+1
ELSE
   PRINT*,'C''est␣fini'
ENDIF
```

Test plus compliqué

```
IF (delta>0.0) THEN
    PRINT*,'Deux␣solutions'
ELSEIF (delta==0.0) THEN
    PRINT*,'Une␣solution␣double'
ELSE
    PRINT*,'Deux␣solutions␣complexes'
ENDIF
```

Boucles

Boucle générale

```
DO
...
   IF(i>ifin) EXIT
ENDDO
```

Boucle tant que ... faire

```
DO WHILE(i<ifin)
...
ENDDO
```

Boucle avec compteur

```
DO i=1,IMAX
...
ENDDO
```

Boucle avec compteur rétrograde

```
DO i=IMAX,1,-1
...
ENDDO
```

Saut à la fin de la boucle

```
CYCLE
```

Sort de la boucle

```
EXIT
```

Met fin au programme

```
STOP
```

Blocs

Programme

```
PROGRAM monprog
    IMPLICIT NONE
    INTEGER :: i ! déclarations
    i = 2        ! instructions
    PRINT*,'i␣=␣', i
END PROGRAM monprog
```

Sous-programmes

```
SUBROUTINE mon_sous_prog(i)
    IMPLICIT NONE
    INTEGER :: i,j
    i=j
END SUBROUTINE mon_sous_prog
```

Appel du sous-programme

```
CALL SUBROUTINE mon_sous_prog(4)
```

Fonctions

```
FUNCTION f(x)
    IMPLICIT NONE
    REAL :: f, x
    f = 2.0*x+5.8
END FUNCTION f
```

Appel de la fonction

```
z = 5.3*f(3.7*u+2.4)
PRINT*,'Resultat␣=␣',f(z)
```

Module

```
MODULE mon_module
    IMPLICIT NONE
    REAL :: t, x
```