Méthodes numériques et éléments de programmation

Guy Munhoven

Institut d'Astrophysique et de Géophysique (Bât. B5c)
Bureau 0/13
eMail: Guy.Munhoven@ulg.ac.be
Tél.: 04-3669771

23 septembre 2014

Méthodes numériques et éléments de programmation
Guy Munhoven
Modules
Opérations entrée/sortie

Plan du cours 2014-2015

Cours théoriques

16-09-2014 *Méthodes numériques pour E.D.O.: introduction*

22-09-2014 *Méthodes de Runge-Kutta; Contrôle du pas; Équations raides*

22-09-2014 *Fortran 95: bases, boucles*

23-09-2014 *Fortran 95: branchements, sous-programmes*

23-09-2014 Fortran 95

- Modules
- Opérations d'entrée-sortie

29-09-2014 Fortran 95: tableaux

07-10-2014 Fortran 95 compléments

Modules

- Rassembler des collections
  - de constantes,
  - de définitions de précision,
  - de variables à partager,
  - de sous-routines,
  - d'interfaces.
- Encapsuler des données ⇒ concepts de base de la programmation orientée objet.

Modules : Exemple d'utilisation

Modules: Examples of Usage

Module avec définition de constantes

```
MODULE ConstantesFondamentales
IMPLICIT NONE

DOUBLE PRECISION, PARAMETER :: pi = 3.14159265358979323846D+00
DOUBLE PRECISION, PARAMETER :: un_demi = 1D0/2D0
DOUBLE PRECISION, PARAMETER :: acc_gravifique = 9.81D0
END MODULE ConstantesFondamentales
```

Programme utilisant le module ConstantesFondamentales

```
PROGRAM DisqueAire
USE ConstantesFondamentales

IMPLICIT NONE
DOUBLE PRECISION :: rayon

WRITE(*,'("Entrez le rayon du disque :")', ADVANCE='NO')
READ(*,*) rayon
WRITE(*,*) 'Aire du disque: ', pi*rayon**2
END PROGRAM DisqueAire
```

Modules : particularités de compilation
Modules: compilation pecularities

- Un module doit être compilé avant tout programme ou sous-programme qui l'utilise.
- Informations d'interfaçage éventuelles automatiquement générées et incluses dans le fichier `*.mod`.
- Certains compilateurs permettent de stocker les fichiers `*.mod` pour utilisation ultérieure à un emplacement centralisé (de manière semblable à des fichiers `*.h` en C ou C++).

Méthodes numériques et éléments de programmation
Guy Munhoven
Modules
Opérations entrée/sortie

Modules : opportunités, dangers et remèdes
Modules: opportunities, dangers, and remedies

- Un module peut utiliser un autre module
- Un module ne peut pas faire référence à soi-même, même pas de manière indirecte
- Conflits de nom possibles entre identificateurs (noms de variables ou de sous-programmes) définis
  - dans deux ou plusieurs modules utilisés dans une même unité de programme
  - dans un module et dans l'unité de programme qui l'utilise
- Résolution
  - restrictions de mise à disposition à la spécification `USE` via la clause `ONLY`
  - renomination (aliasing) lors de la spécification `USE`

Modules : Exemple de restriction au USE

Modules: Example of Restriction with USE

Module avec définition de constantes

```
MODULE ConstantesFondamentales
IMPLICIT NONE

DOUBLE PRECISION, PARAMETER :: pi = 3.14159265358979323846D+00
DOUBLE PRECISION, PARAMETER :: un_demi = 1D0/2D0
DOUBLE PRECISION, PARAMETER :: acc_gravifique = 9.81D0
END MODULE ConstantesFondamentales
```

Programme utilisant le module ConstantesFondamentales

```
PROGRAM DisqueAire
USE ConstantesFondamentales, ONLY: pi

IMPLICIT NONE
DOUBLE PRECISION :: rayon

WRITE(*,'("Entrez le rayon du disque :")', ADVANCE='NO')
READ(*,*) rayon
WRITE(*,*) 'Aire du disque: ', pi*rayon**2
END PROGRAM DisqueAire
```

Méthodes numériques et éléments de programmation
Guy Munhoven
Modules
Opérations entrée/sortie

Modules : Exemple de renomination à USE

Modules: Example of Renaming at USE

Programme utilisant le module ConstantesFondamentales

```
PROGRAM DisqueAire
USE ConstantesFondamentales
! conflit entre le 'pi' du module et le 'pi' déclaré localement


IMPLICIT NONE
DOUBLE PRECISION :: pi = 3.1416
DOUBLE PRECISION :: rayon

WRITE(*,'("Entrez le rayon du disque :")', ADVANCE='NO')
READ(*,*) rayon
WRITE(*,*) 'Aire du disque: ', pi*rayon**2

END PROGRAM DisqueAire
```

Modules : Exemple de renomination à USE

Modules: Example of Renaming at USE

Programme utilisant le module ConstantesFondamentales

```
PROGRAM DisqueAire
USE ConstantesFondamentales, pi => pi_precis
! conflit entre le 'pi' du module et le 'pi' déclaré localement
! le 'pi' du MODULE est localement référencé par 'pi_precis'

IMPLICIT NONE
DOUBLE PRECISION :: pi = 3.1416
DOUBLE PRECISION :: rayon

WRITE(*,'("Entrez le rayon du disque :")', ADVANCE='NO')
READ(*,*) rayon
WRITE(*,*) 'Aire du disque: ', pi*rayon**2
WRITE(*,*) 'Aire du disque: ', pi_precis*rayon**2
END PROGRAM DisqueAire
```

Méthodes numériques et éléments de programmation
Guy Munhoven
Modules
Opérations entrée/sortie

Modules : Exemple de renomination à USE

Modules: Example of Renaming at USE

Programme utilisant le module ConstantesFondamentales

```
PROGRAM DisqueAire
USE ConstantesFondamentales, ONLY: pi => pi_precis
! conflit entre le 'pi' du module et le 'pi' déclaré localement
! le 'pi' du MODULE est localement référencé par 'pi_precis'

IMPLICIT NONE
DOUBLE PRECISION :: pi = 3.1416
DOUBLE PRECISION :: rayon

WRITE(*,'("Entrez le rayon du disque :")', ADVANCE='NO')
READ(*,*) rayon
WRITE(*,*) 'Aire du disque: ', pi*rayon**2
WRITE(*,*) 'Aire du disque: ', pi_precis*rayon**2
END PROGRAM DisqueAire
```

Opérations d'entrée-sortie
Input-Output Operations

- Communication par canaux.
- Référence est faite à un canal par un numéro d'unité logique (de type `INTEGER`).
- Unités logiques peuvent être connectées
  - au terminal (sortie),
  - au clavier (entrée),
  - à un fichier (entrée et sortie),
  - à une imprimante (rarement de nos jours),
  - un lecteur de bande (historique).
  - ...

Opérations d'entrée-sortie
Input-Output Operations

Méthodes numériques et éléments de programmation
Guy Munhoven
Modules
Opérations entrée/sortie

- Connection d'unités logiques et spécifications de communication par la commande `OPEN`

  `OPEN(UNIT=`*`i_unité`*`, FILE=`*`nom_fichier`*`,` *options*`)`
- *`i_unité`* est un `INTEGER`
  - sous forme de constante littérale (déconseillé),
  - sous forme de variable `INTEGER`.
- *`nom_fichier`* réflète le chemin d'accès vers le fichier désiré et est de type `CHARACTER`
  - sous forme de constante littérale (déconseillé),
  - sous forme de variable `CHARACTER`.
- Fermeture et déconnection d'unités logiques par la commande `CLOSE`

  `CLOSE(UNIT=`*`i_unité`*`)`.
- Nombreuses options

Opérations d'entrée-sortie
Input-Output Operations

- Lecture par la commande `READ`

  `READ(UNIT=`*`i_unité`*`, FMT=`*`desc_format`*`,` *`options`*`)`

  ou

  `READ(`*`i_unité`*`,` *`desc_format`*`,` *`options`*`)`.

- *`i_unité`* est l'unité logique de l'origine, présente sous forme de `*` pour l'unité par défaut (clavier), qui est généralement l'unité 5 (ouverte *a priori*).
- *`desc_format`* indique le format des données à lire et se présente sous forme
  - d'une `*` pour utiliser le format par défaut;
  - d'une étiquette, qui réfère à la ligne où est décrit le format à l'aide de l'instruction `FORMAT` (non exécutable);
  - d'une chaîne de caractères (littérale ou variable `CHARACTER`).

Méthodes numériques et éléments de programmation
Guy Munhoven
Modules
Opérations entrée/sortie

Opérations d'entrée-sortie
Input-Output Operations

- Ecriture par la commande `WRITE`

  `WRITE(UNIT=`*`i_unité`*`, FMT=`*`desc_format`*`,` *`options`*`)`

  ou

  `WRITE(`*`i_unité`*`,` *`desc_format`*`,` *`options`*`)`

- *`i_unité`* est l'unité logique de destination, présente sous forme de `*` pour l'unité par défaut (écran, terminal), qui est généralement l'unité 6 (ouverte *a priori*).
- *`desc_format`* indique le format des données à écrire

Opérations d'entrée-sortie : descripteurs de format
Input-Output Operations: Format Descriptors

- Forme générique d'un descripteur de format

  $$(\texttt{[}n\texttt{]}C\texttt{c[.}d\texttt{[E}e\texttt{]], ...})$$

  précédé par `FORMAT` si spécification par ligne étiquettée, ou mis entre simple ou doubles guillemets si spécification par chaîne de caractères, et où
  - C est remplacé par `A` pour une chaîne de caractères, `D`, `E`, `EN` ou `ES` pour des réels en format scientifique, `F` ou `G` pour des réels en représentation virgule fixe, `I` pour des entiers;
  - c donne le nombre de colonnes que chaque objet de genre C va occuper,
  - d le nombre de décimales à inclure, le cas échéant
  - n le nombre d'objects de type C à écrire
  - e le nombre de chiffres que l'exposant comportera (sans le signe), lorsque la lettre `E` est en affixe.

Opérations d'entrée-sortie : descripteurs de format
Input-Output Operations: Format Descriptors

- Descripteurs `E`c`.`d et `E`c`.`d`E`e : $0 \leq$ significand < 1;
- Descripteurs `EN`c`.`d et `EN`c`.`d`E`e : exposant multiple de 3 et $1 \leq$ significand < 1000;
- Descripteurs `ES`c`.`d et `ES`c`.`d`E`e : $1 \leq$ significand < 10;
- Plusieurs descripteurs, séparés par des virgules, peuvent être rassemblés dans un descripteur global;
- Plusieurs sous-listes peuvent être regroupées entre parenthèses;
- Peuvent aussi faire partie des descripteurs:
  - des chaînes de caractères, qui doivent être constantes avec l'instruction `FORMAT(...)`
  - des descripteurs n`X` pour sauter (i.e., ignorer) n caractères (en entrée) ou imprimer n blancs (en sortie)

Opérations d'entrée-sortie : descripteurs de format

Input-Output Operations: Format Descriptors

Spécification de format: Manières possibles

```
...
WRITE(*,*) x, y, z            ! Format par défaut


WRITE(*,100) x, y, z          ! Format spécifié à l'étiquette 100
100 FORMAT(3F10.4)


fmt_string = '(3F10.4)'
WRITE(*,fmt_string) x, y, z   ! Format spécifié dans la chaîne
                              ! de caractères fmt_string
```

Descripteurs de format: exemples

Format Descriptors: Examples

X =	'(E13.5)'	'(ES13.5)'	'(EN13.5)'
123456789D0	␣␣0.12346E+09	␣␣1.23457E+08	123.45679E+06
123456789D-9	␣␣0.12346E+00	␣␣1.23457E-01	123.45679E-03
123456789D-200	␣␣0.12346-191	␣␣1.23457-192	␣␣1.23457-192
-123456789D0	␣-0.12346E+09	␣-1.23457E+08	**********
-123456789D-9	␣-0.12346E+00	␣-1.23457E-01	**********
-123456789D-200	␣-0.12346-191	␣-1.23457-192	␣-1.23457-192

- Symboles ␣ indiquent des espaces imprimés;
- Affichage d'un champ rempli de $*$ si une sortie ne peut se faire sous le format désiré;
- Omission de la lettre `E` (ou `D`) si nécessaire.