Actualizacion y nivelacion de conocimientos de computación

orientado para futuros participantes del curso de HPC

Julio 2009

Agenda

- DIA 2 (15 de julio): Lenguaje FORTRAN
  - FORTRAN 77, FORTRAN 90, FORTRAN 95
  - Estructuras de control (if, do)
  - Tipos básicos de datos (integer, , etc)
  - Estructuras de datos (arrays, estructuras, etc)
  - Memoria dinámica (allocate y deallocate)

Lenguaje FORTRAN

Fortran 77

Restricciones posicionales:
- Líneas empiezan en columna 7.
- Líneas terminan en columna 72.
- Se continua una línea en la columna 6.

No posee allocamiento dinámico.
Se comenta con la letra c (C) en las columnas 1 a 5.
Emplean los bloques COMMON para el manejo de memoria.

Fortran 90

No posee restricciones posicionales.
Posee allocamiento dinámico (punteros).
Se comenta con la letra !.
Se introducen los modulos para encapsular datos.
Incluye KIND para definir la precisión.
Permite sobrecargar operadores.
Funciones recursiva.

Fortran 95

Paralelismo (HPF)

FORALL

FORTRAN 90

Estructura básica

```
PROGRAM nombre programa
  declaracion de variables
  sentencias Fortran
END PROGRAM nombre programa
```


Hello World !!!

```
program hello
 print *,"Hello World!"
end program hello
```

- Puede ir todo en mayuscula/minuscula.
- Luego del end puede no ir nada.

Estructuras de control

```
if (condición1) sentencia

if (condición1) then
        :
endif

if (condición1) then
:
Else
:
Endif
```

Operadores lógicos

```
>    >=   <    <=    ==   / =
.GT. .GE. .LT. .LE.  .EQ.  .NE.
```


Estructuras de control

```
do j = 1, n
  …
Enddo

do while ( continue )
  ..
Enddo

do
  ..
Enddo
```


Estructuras de control

```
SELECT CASE (numero)
 CASE (:3)
  . . .
 CASE (5)
  . . .
 CASE (4,6:9)
  . . . .
 CASE (10:15)

 CASE DEFAULT
  ……
 END SELECT
```


Tipos de datos

- **Intrínsecos**
  - **INTEGER, REAL, CHARACTER, LOGICAL, COMPLEX**

  - INTEGER:: numentero1, numentero2, numentero3
  - REAL:: numreal1, numreal2
  - CHARACTER (LEN=longitud):: nombre1, nombre2
  - CHARACTER (longitud):: nombre1, nombre2
  - CHARACTER*longitud:: nombre1, nombre2
  - LOGICAL:: var1, var2
  - Valores: .TRUE. .FALSE.
  - COMPLEX:: nombre1, nombre2 !Almacena par de números reales
  - COMPLEX:: numcplx
  - numcplx=(5.6,1.4E-2)
  - La función AIMAG(x) devuelve la parte imaginaria de x y CONJG(x) su conjugado.

Tipos de datos

Ejemplo de manejo de substrings:

- Operador de concatenación: //

- CHARACTER(LEN=8):: ch
- ch="abcdefgh"
- ch(4:5)="pq" → ch="abcpqfgh"
- ch(3:) ="xyz" → ch="abxyz• • •"
- ch(:3) ="xyz" → ch="xyzdefgh"

Tipos de datos

Constantes

No se permite cambiar su valor a lo largo del programa

- REAL, PARAMETER:: e=2.718282, ediv2=e/2.0

Tipos de datos

Parámetro KIND

```
INTEGER:: i
REAL(KIND=4):: x, y, z
! Tambien se permite REAL(4):: x, y, z
i=KIND(x) ! Devuelve i=4
```

La función KIND(x) devuelve el valor del parámetro KIND de la variable x.

KIND es dependiente del procesador, ver
SELECTED REAL KIND(P, R)

En CHARACTER KIND se utiliza para el juego de caracteres.

Tipos de datos

- **Derividados**

```
TYPE mitipo
  CHARACTER (LEN=12):: nombre, apellido
  INTEGER:: edad
END TYPE mitipo
```

- Declaración de variables:
  TYPE(mitipo):: persona1, persona2
- Asignaci´on de un valor constante (constructor deestructura):
  persona1=mitipo("juan","rodriguez",31)
- Referencia a componentes:
  persona1%nombre="juan"
  persona1%apellido=persona2%apellido

Tipos de datos

- **ARRAYS**

```
REAL, DIMENSION(20):: arr1, arr2, arr3
REAL:: arr1(20), arr2(20), arr3(20)

Por defecto, los índices comienzan en 1, podemos especificar rangos:
REAL, DIMENSION(21:40):: arr1

Referencia a componentes: nombre array(exp entera)

Inicialización de arrays (constructor de array):
arr4 = (/ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10/)
Uso de un DO implícito:
arr4 = (/ (i,i=1,10) /)

Se puede inicializar un array en su propia sentencia de declaración

Se puede tratar un array como si se tratara de un objeto único:
REAL, DIMENSION(20):: arr1, arr2
arr1 = 0.0
arr1 = SIN(arr2)
```


Tipos de datos

- **MATRICES**

```
REAL, DIMENSION(4,3):: matriz
REAL, DIMENSION(5:9,-1:0):: matriz

Algunas funciones intrínsecas F90: SUM, PRODUCT, DOT
  PRODUCT, MAXVAL, MINVAL (funciones de reducción)

Otras funciones: MATMUL, TRANSPOSE

REAL:: matriz1(2,3), matriz2(3,2), matriz12(2,2)
matriz2 = TRANSPOSE(matriz1)
matriz12 = MATMUL(matriz1,matriz2)

En Fortran90 (como en FORTRAN77) los arrays se
  almacenan por columnas !!!!!
```


Tipos de datos

- **ARRAYS DINÁMICOS (allocatable)**

```
Declaración:
REAL, ALLOCATABLE, DIMENSION(:,:):: dinmat

Reserva de espacio para el array. Sentencia ejecutable:
ALLOCATE(lista arrays [, STAT=variable estado])

INTEGER:: error, n
REAL, ALLOCATABLE, DIMENSION(:,:):: dinmat
INTEGER, ALLOCATABLE, DIMENSION(:):: dinvec
...
ALLOCATE(dinmat(4:9,n+2), dinvec(-n:n), STAT=error)

Liberación del espacio de memoria de un array dinámico:

DEALLOCATE(lista arrays [, STAT=variable estado])
DEALLOCATE(dinmat, dinvec, STAT=error)
```


Tipos de datos

- **PUNTROS**

```
POINTER: REAL, POINTER:: p, q

También puede apuntar a variables de un tipo de dato derivado:
TYPE(mitipo), POINTER:: p1

Las variables que pueden ser apuntadas por un puntero deben tener el
  atributo TARGET:
REAL, TARGET:: dato

Asociación de un puntero con una variable o con otro puntero:
p => dato q => p

Ejemplos:
REAL, POINTER:: p, q
REAL, TARGET:: da, db
p => da !p apunta a da
q => da !q tambien apunta a da
p => db !Ahora p apunta a db
r => q !r apunta a da
q => p !Ahora q apunta a db
```

Pendiente

Un montón de cosas, entre otras:

- Funciones, procedimientos
- Parametros de funciones/procedimientos
- WRITE / FORMAT
- Funciones de bits
- Modulos, intefaces, sobrecarga de operadores
- Entrada / Salida