Enumerados, subrangos y tuplas

Introducción

Hasta el momento hemos visto los siguientes tipos de datos: entero (`integer`), real (`real`), lógico (`logical`) y carácter (`character`); aunque estos tipos resultan imprescindibles y muy útiles existen ocasiones en los que resulta necesario trabajar con datos que no se ajustan a ninguno de estos tipos y que precisarían la creación de un tipo de dato "a medida". Por ejemplo, los días de la semana, los meses del año, el sexo de una persona o una referencia bibliográfica no pertenecen a ninguno de los tipos predefinidos pero sí podrían describirse a partir de ellos.

Todos los lenguajes de programación modernos permiten al usuario la definición de tipos de datos propios, los más sencillos son los enumerados, subrangos y tuplas (o registros); en esta lección veremos la forma en que se definen y utilizan dichos tipos tanto en la notación algorítmica como en FORTRAN 90.

Tipo enumerado

Se trata del más sencillo de los tipos definidos por el usuario puesto que para su definición basta con enumerar los valores de sus elementos; algunos ejemplos de tipos enumerados son los siguientes:

- los días de la semana,
- las estaciones del año,
- los palos de una baraja,
- el sexo de una persona,
- ...

Para definir un nuevo tipo de datos en la notación algorítmica se debe indicar una nueva sección, de forma análoga a como se hace con las constantes y variables; para ello se utiliza la palabra reservada `tipos`. Si el tipo es un enumerado se emplearía la sintaxis siguiente:

```
tipos
  nombre_enumerado = {valor1, valor2, ..., valorN}
```

Así, los ejemplos anteriores se definirían de la forma siguiente:

```
tipos
  dias_semana = {lunes, martes, miercoles, jueves, viernes, sabado, domingo}
  estaciones_anno = {primavera, verano, otono, invierno}
  palos_baraja {oros, copas, espadas, bastos}
  sexo = {mujer, hombre}
```

Para declarar variables de dichos tipos se procede como con los tipos predefinidos:

```
variables
  dia ∈ semana
  estacion ∈ estaciones_anno
  palo ∈ palos_baraja
  sexo_contribuyente ∈ sexo
```

Una característica del tipo enumerado es que existe una relación de orden establecida entre sus valores de forma implícita al definir el tipo; de esta manera es posible utilizar los operadores relacionales entre variables de tipo `ord`, `pred` y `succ` que retornan, respectivamente, el valor ordinal, el valor anterior y el valor siguiente de un elemento de tipo enumerado. A continuación se muestran algunos ejemplos que ilustran la relación de orden en los tipos enumerados.

Expresión	Valor / Resultado
`lunes < martes`	`verdadero`
`sabado < domingo`	`verdadero`
`miercoles = sabado`	`falso`
`lunes ≠ viernes`	`verdadero`
`ord (lunes)`	`1`
`ord (miercoles)`	`3`
`ord (viernes)`	`5`
`ord (domingo)`	`7`

Expresión	Valor / Resultado
`succ (lunes)`	`martes`
`succ (miercoles)`	`jueves`
`succ (viernes)`	`sabado`
`succ (domingo)`	`NO DEFINIDO`
`pred (domingo)`	`sabado`
`pred (viernes)`	`jueves`
`pred (miercoles)`	`martes`
`pred (lunes)`	`NO DEFINIDO`

En cuanto a la definición de tipos enumerados en FORTRAN se puede resumir en dos palabras: **no existe**; hasta la versión de FORTRAN 2000 no se han introducido los tipos enumerados y estos admiten tan sólo valores enteros.

Tipo subrango

Un subrango es un subconjunto de un tipo enumerable (`entero`, `carácter`, `lógico` y cualquier tipo enumerado) que contiene todos los elementos entre dos valores datos; para definir un subrango se procede como sigue:

```
tipos
  nombre_subrango = valor_inicial .. valor_final
```

A continuación se muestran algunos ejemplos de subrangos:

```
tipos
  minusculas = 'a' .. 'z'
  mayusculas = 'a' .. 'z'
  numeros = '0' .. '9'
  dias_laborables = lunes .. viernes
  fin_semana = sabado .. domingo
```

Al igual que sucede con los tipos enumerados, no es posible definir subrangos en FORTRAN 90.

Tuplas o registros

Una tupla o registro es un conjunto de datos con un número fijo de componentes no necesariamente del mismo tipo; existen múltiples ejemplos de datos cuya representación es posible mediante una tupla:

- Información sobre un alumno (nombre, apellidos, DNI, dirección, etc.)
- Referencias bibliográficas (título, autor, editorial, año de edición, etc.)
- Información sobre una película (título, director, año, etc.)

La sintaxis para la definición de una tupla en la notación algorítmica es la siguiente:

```
tipos
  nombre_tupla = tupla
    campo1 ∈ tipo1
    campo2 ∈ tipo2
    ...
    campoN ∈ tipoN
  fin tupla
```

Una característica de este tipo de datos es su naturaleza "recursiva" puesto que cada componente puede ser de cualquier tipo es posible que dicho tipo sea, a su vez, una tupla; el ejemplo siguiente ilustra muy bien esta característica:

```
tipos
  direccion = tupla
    calle ∈ carácter
    portal, piso ∈ entero
    letra, CP, localidad, municipio, provincia ∈ carácter
  fin tupla

  fecha = tupla
    dia, mes, anno ∈ entero
  fin tupla

  persona = tupla
    nombre, apellido1, apellido2, DNI ∈ carácter
    nacimiento ∈ fecha
  fin tupla

  alumno = tupla
    datos_personales ∈ persona
    asignatura ∈ carácter
    direccion_asturias, direccion_familiar ∈ direccion
  fin tupla
```

Para definir en FORTRAN 90 una tupla o registro se emplea la siguiente sintaxis:

```
type nombre_tupla
  tipo1 campo1
  tipo2 campo2
  ...
  tipoN campoN
end type nombre_tupla
```

El ejemplo anterior se traduciría a FORTRAN de esta forma:

```
type direccion
  character*64 calle
  integer portal, piso
  character*2 letra
  character*5 CP
  character*64 localidad, municipio, provincia
end type direccion

type fecha
  integer dia, mes, anno
end type fecha

type persona
  character*64 nombre, apellido1, apellido2
  character*8 DNI
  type(fecha) nacimiento
end type persona

type alumno
  type(persona) datos_personales
  character*32 asignatura
  type(direccion) direccion_asturias, direccion_familiar
end type alumno
```

Obsérvese que, al contrario que la notación algorítmica, al declarar una variable de un tipo registro en FORTRAN es necesario indicarlo de manera explícita mediante la sintaxis:

```
type(tipo_tupla) variable
```

Ahora bien, una vez se ha definido un tipo registro y se tienen variables de dicho tipo, ¿de qué manera se utilizan dichas variables y sus distintos campos? Existen dos formas de trabajar con los registros: procesando registros completos o procesándolos campo a campo.

En el primer caso es posible asignar un registro a otro registro del mismo tipo, escribirlos y leerlos (esto es un tanto desaconsejable puesto que el usuario no tiene ninguna información sobre los datos que se le están solicitando). A continuación se muestra un ejemplo (en la notación algorítmica y en FORTRAN) de procesamiento de registros completos.

```
tipos
  fecha = tupla
    dia, mes, anno ∈ entero
  fin tupla

variables
  nacimiento, hoy ∈ fecha

inicio
  ...
  nacimiento ← hoy
  escribir nacimiento
  ...
fin
```

```
program programa
  implicit none

  type fecha
    integer dia, mes, anno
  end type fecha

  type(fecha) nacimiento, hoy

  ...
  nacimiento = hoy
  print *, nacimiento
  ...
end
```

Para procesar los registros campo a campo es necesario utilizar los denominados “selectores de campo” que se construyen mediante el identificador de la variable y el del campo separados por un punto ‘.’ en el caso de la notación algorítmica y un carácter ‘%’ en el caso de FORTRAN. El ejemplo siguiente muestra la forma en que se accede a los campos de un registro desde una función (la forma más adecuada de cara al usuario para leer un registro por teclado).

```
tipos
  persona = tupla
    nombre, apellido1, apellido2, dni ∈ carácter
  fin tupla

variables
  cliente ∈ persona

inicio

  cliente ← leerdatos()
  escribir cliente

  persona funcion leerdatos()
  inicio
    escribir 'nombre?'
    leer leerdatos.nombre

    escribir 'primer apellido?'
    leer leerdatos.apellido1

    escribir 'segundo apellido?'
    leer leerdatos.apellido2

    escribir 'dni?'
    leer leerdatos.dni
  fin
fin
```

```
program programa
  implicit none

  type persona
    character*64 nombre, apellido1, apellido2
    character*8 dni
  end type persona

  type(persona) cliente

  cliente = leerdatos()
  print *, cliente

  contains

    type(persona) function leerdatos()
      print *, 'nombre?'
      read *, leerdatos%nombre

      print *, 'primer apellido?'
      read *, leerdatos%apellido1

      print *, 'segundo apellido?'
      read *, leerdatos%apellido2

      print *, 'dni?'
      read *, leerdatos%dni
    end function leerdatos
end
```

En caso de registros anidados el selector de campo debe indicar la "ruta" completa hasta llegar al último campo del registro; por ejemplo, en el primer ejemplo en el que un `alumno` tiene un campo `datos_personales` y éste tiene un campo `nacimiento` se utilizaría el siguiente selector de campo para acceder al campo `dia`:

```
tipos
  alumno = tupla
    ...
  fin tupla

variables
  alumno1 ∈ alumno

inicio
  ...
  leer alumno1.datos_personales.nacimiento.dia
  ...
fin
```

```
program programa
  implicit none

  type alumno
    ...
  end type alumno

  type(alumno) alumno1

  ...
  read *, alumno1%datos_personales%nacimiento%dia
  ...
end
```

Resumen

1. En muchas ocasiones es necesario representar datos que no se ajustan a ninguno de los tipos predefinidos (entero, real, lógico y carácter) pero que puede ser definido en función de estos. Todos los lenguajes modernos permiten la definición de tipos de usuario.
2. Los tipos definidos por el usuario más habituales son: enumerado, subrango y tupla o registro.
3. Un tipo enumerado consiste en la simple enumeración de los valores de sus elementos; son tipos enumerados los días de la semana, los palos de la baraja o los meses del año.
4. Al definir un tipo enumerado se establece una relación de orden entre sus valores; así, es posible utilizar los operadores relacionales con elementos enumerados así como las funciones `ord` (ordinal), `pred` (predecesor) y `succ` (sucesor).
5. Un tipo enumerado se define en la notación algorítmica empleando la siguiente sintaxis:

```
nombre_enumerado = {valor1, valor2, ..., valorN}
```

6. En FORTRAN 90 no es posible definir tipos enumerados.
7. Un subrango es un subconjunto de un tipo enumerable (entero, carácter, lógico y enumerado) que contiene todos los elementos entre dos valores datos.
8. Un subrango se define en la notación algorítmica empleando esta sintáxis.

```
nombre_subrango = valor_inicial .. valor_final
```

9. No es posible definir subrangos en FORTRAN 90.
10. Una tupla o registro es un conjunto de datos con un número fijo de componentes generalmente de tipos diferentes (incluyendo otros registros).
11. La sintáxis para definir una tupla en la notación algorítmica y en FORTRAN es la siguiente:

```
tipos
  nombre_tupla = tupla
    campo1 ∈ tipo1
    campo2 ∈ tipo2
    ...
    campoN ∈ tipoN
  fin tupla
```

```
type nombre_tupla
  tipo1 campo1
  tipo2 campo2
  ...
  tipoN campoN
end type nombre_tupla
```

12. Para declarar una variable de tipo registro en FORTRAN se utiliza esta sintaxis:

```
type(tipo_tupla) variable
```

13. Los registros o tuplas pueden procesarse de forma completa o campo por campo; para hacer lo último es necesario utilizar selectores de campo que se construyen uniendo los identificadores del registro y del campo mediante un punto '`.`' en la notación algorítmica y el carácter '`%`' en FORTRAN.