Ya hemos trabajado con diferentes unidades de programación

1. Program
2. Subroutine
3. Function
4. Module

¿Para qué sirve esta unidad?

Nos ayudará a facilitar la escritura de programas en el paradigma de la programación modular (facilita la programación modular)

Estructura

MODULE nombre

**! No requiere de una lista de argumentos, a diferencia de las subrutinas y funciones**

Implicit Nones

Bloque de instrucciones

End Module nombre

Ventajas

Se pueden definir variables/constantes **globales** del programa

Estas variables están disponibles para todo el código mediante una sola declaración.

Deben de declararse usando Selected Real Kind

El intercambio de datos entre subrutinas, funciones o programas es ahora más fácil si son globales y con módulos

EJEMPLO

**Module** mis\_definiciones

Implicit None

Save

**! Le decimos explícitamente que tenga cuidado con los valores transmitidos, procurando que la información que estamos transfiriendo no se corrompa**

**! PROCUREN USARLO MUCHO**

Integer, Parameter :: d=Selected\_Real\_Kind(18,200)

**End Module**

**Program** ejemplo

Use mis\_definiciones

**! Se especifica el módulo que se va a usar, instrucción que va antes del implicit none**

Implicit None

Real(d) :: X, Y, Z

**! variables con el Selected Real Kind declarado en el módulo**

Call sub(x,y,z)

**End Program**

**Subroutine** sub(x,y,z)

Use mis\_definiciones

Real(d) :: x,y,z

**End Subroutine**

Las variables locales no deben llamarse igual que las globales.

¿Qué más podemos usar en un módulo?

Se pueden incluir procedimientos en los módulos

* SUbrutinas
* FUnciones

**! También se pueden agregar a los módulos**

MODULE mis\_def

Implicit None

Save

Real :: pi=3.1415

Contains

Subroutine mod(x,y)

Implicit None

End Subroutine

End Module

¿De qué sirve poner subrutinas/funciones en un módulo?

De esta manera la información de la subrutina queda expuesta al compilados y permite detectar errores que de otra forma permanecen ocultos

EJEMPLO

**PROGRAM** ???

Implicit None

Real :: x=1

Call prueba(x)

END PROGRAM

**! No hay error**

**SUBROUTINE** prueba(xo)

IMPLICIT NONE

Integer :: xo

PRINT\*, xo

END SUBROUTINE

**! No hay error**

**! Están mal declaradas las variables, pero el compilador no captaría el error**

**! El resultado va a ser el correcto, el compilador no**

**! Metiendo la subrutina al módulo si se mostraría el error**

¿Como Compilar con módulos?

¡¡¡orden al compilar importa!!!

gfortran (módulo) (demás unidades de programación)

**! El módulo va primero**

**Ejercicio**

Escriba un programa que utilice un módulo para calcular

1. Área de un círculo
2. Área de un cuadrado
3. Volumen de una esfera

Usen variables globales y procedimientos(sub,func) en un mismo módulo