Paradigma de programación: Programación orientada a objetos

Clase : Sección de código donde se definen las características y procedimientos de un objeto; es decir, una clase se considera la estructura que da forma a un objeto.

¿Qué es un objeto?

Cualquier variable que posee

1. Características
2. Métodos (procedimientos)

Una clase NO es un objeto

Un objeto se conoce como instancia de una clase

EJEMPLO

Clase: perros

**Características:**

* tamaño
* raza

**Métodos:**

* ladrar
* rastrear

Objeto: firulais (mi perro)

Cuando un objeto se crea, en programación se dice que se ha producido una instancia de la clase

* *Instanciación*



Características permanecen ocultas del resto del código (variables privadas

Encapsulación: La única manera de acceder a ellas es a través de los métodos.

PRIVADAS : Para evitar la corrupción de las variables al intercambiarlos con otras secciones de código.

En Fortran las clases se definen en los módulos

EJEMPLO

Construir un programa que calcule el area y el perimetro de un círculo utilizando programación orientada a objetos (OOP) en Fortran.

EJEMPLO : :

MODULE clase\_circulo

IMPLICIT NONE

REAL : : Pi =3.14159

TYPE : : circulo

! Características del objeto

! Características que tendrán los objetos de mi clase

REAL : : radio

REAL : : area

REAL : : perimetro

CONTAINS

PROCEDURE : : gr => get\_the\_radius

! get\_the\_radius = nombre del método = nombre de subrutina

PROCEDURE : : ga => get\_the\_area

PROCEDURE : : gp => get\_the\_perimeter

! => manera de asignarle un alias a una subrutina con un nombre largo/complicado, pero no es necesario

END TYPE

CONTAINS

! Construyendo las subrutinas

SUBROUTINE get\_the\_radius (algo)

! algo = variable tipo dummy

! futuro objeto declarado en el principal

IMPLICIT NONE

CLASS (circulo) : : algo

PRINT\*, “Dame el radio del circulo”

READ\*, algo%radio

! % indica qué variable/característica se quiere leer del objeto

END SUBROUTINE

SUBROUTINE get\_the\_area (algo)

IMPLICIT NONE

CLASS (circulo) : : algo

algo%area = Pi\*algo%radio\*\*2

PRINT\*, “el area es”, algo%area

END SUBROUTINE

SUBROUTINE get\_the\_perimeter (algo)

IMPLICIT NONE

CLASS (circulo) : : algo

algo%perimetro = 2\*Pi\*algo%radio

PRINT\*, “el perimetro es”, algo%perimetro

END SUBROUTINE

END MODULE

PROGRAM principal

USE clase\_circulo

! única instrucción que va antes del Implicit None

IMPLICIT NONE

! Generamos el objeto

TYPE (circulo) : : circulo 1

! circulo1 = nombre del objeto

! circulo1 = instanciación de la clase

CALL circulo1%gr

! Objeto llamando al método

CALL circulo1%ga

CALL circulo1%gp

END PROGRAM