**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

! Métodos del objeto

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

! Instrucción necesaria para agregar subrutinas o funciones a un módulo

! Le dice al compilador que las subrutinas son parte del módulo

! 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

! Queremos el radio de este objeto/variable dummy que aún no es declarado

! % 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 del circulo es:", algo%perimetro

End Subroutine

**End Module**

**Program** princlase

Use Clase\_Circulo

Implicit None

! Generamos el objeto

Type(circulo) :: circulo1

! circulo1 = nombre del objeto

! circulo1 = instanciación de la clase

Call circulo1%gr

! Objeto llamando al método

Call circulo1%ga

Call circulo1%gp

**End Program**