

Programación Orientada a Objetos (POO)

La Programación Orientada a Objetos (POO) es un paradigma de programación que organiza el código en torno a “objetos”, que son instancias de “clases”. Una clase define las propiedades (atributos) y comportamientos (métodos) que sus objetos pueden tener. La POO permite modelar conceptos del mundo real de forma más natural y estructurada, promoviendo la reutilización de código mediante la herencia, y facilitando la modularidad y la encapsulación.

Conceptos de la POO:

- **Clase:** Es un plano o molde que define las características y comportamientos de un objeto.
- **Objeto:** Es una instancia concreta de una clase, es decir, una representación real de algo en la programación.
- **Atributos:** Son las características o propiedades que describen a un objeto (color, tamaño, etc.).
- **Métodos:** Son las acciones o funciones que un objeto puede realizar (correr, saltar, calcular, etc.).
- **Constructores y Destructores:** Son métodos especiales que se ejecutan al crear (constructor) o eliminar (destructor) un objeto.
- **Encapsulamiento:** Es el mecanismo que oculta los detalles internos de un objeto, permitiendo acceder a ellos solo a través de sus métodos.
- **Herencia:** Es la capacidad de una clase de heredar atributos y métodos de otra clase, creando una relación jerárquica.
- **Polimorfismo:** Es la capacidad de que objetos de diferentes clases puedan responder de manera diferente al mismo mensaje.
- **Abstracción:** Es el proceso de simplificar un sistema complejo, enfocándose en las características esenciales y ocultando los detalles innecesarios.

Clases

Para definir una clase se utiliza la sintaxis:

```
class Nombre_clase:  
    #Bloque de código
```

Ejemplo: El siguiente código crea una clase **Persona** vacía

```
class Persona:  
    pass
```

Objetos

Para crear un objeto se utiliza la sintaxis:

```
Nombre_objeto = Nombre_clase()
```

Ejemplo: Se crea un objeto **persona1** de la clase **Persona** y se imprime el tipo de datos

```
persona1 = Persona()  
print(type(persona1))
```

```
<class '__main__.Persona'>
```