**POLIMORFISMO**

El concepto de polimorfismo significa tomar muchas formas y puede aplicarse a métodos.

En este ejemplo, se declaran 2 clases Gerente y Empleado (la primera hereda de la segunda) con sus respectivos métodos \_\_str\_\_.

El ejemplo de polimorfismo se aplica al momento de declarar un método llamado **detallar()** que se declara en la clase Padre (Empleado) e invoca al método \_\_str\_\_ recibiendo una variable self como parámetro.

Esta es la variable que hace que, dependiendo desde el tipo de instancia que se llame el método (Empleado o Gerente) El resultado sea distinto.

class Empleado:

    def \_\_init\_\_(self, nombre, sueldo) -> None:

        self.nombre = nombre

        self.sueldo = sueldo

    def \_\_str\_\_(self) -> str:

        return f'Empleado: "{self.nombre}" Sueldo:{self.sueldo}'

    def detallar(self):

        return self.\_\_str\_\_()

class Gerente(Empleado):

    def \_\_init\_\_(self, nombre, sueldo, depto) -> None:

        super().\_\_init\_\_(nombre, sueldo)

        self.depto = depto

    def \_\_str\_\_(self) -> str:

        return f'Gerente Departamento: "{self.depto}" {super().\_\_str\_\_()}'

Ahora el código de prueba:

emp1 = Empleado('Ale', 90000)

ger = Gerente('john', 150000, 'Sist')

print(emp1.detallar())

print(ger.detallar())

**METODO ISINSTANCE**

El método isinstance retorna True o False. Recibe dos parámetros, la variable “a analizar” y la clase de la que se quiere saber si es instancia.

Ej:

print(isinstance(emp1, Empleado))

print(isinstance(emp1, Gerente))

Estas líneas de código retornan True y False respectivamente, ya que esta es la declaración de la variable emp1:

emp1 = Empleado('Ale', 90000)