**Consulta de Programación Orientada a Objetos**

**Nombre:** Gabriela Loyola Romero

* **Sobre la implementación de los conceptos de Herencia y Polimorfismo lenguaje de programación Python**

La herencia es una técnica de la Programación Orientada a Objetos en la que una clase (conocida como clase hija o subclase) hereda todos los métodos y propiedades de otra clase (conocida como padre o clase base).

La sintaxis para definir una clase que herede de otra es la siguiente:

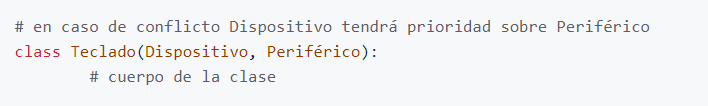
## 

La subclase puede agregar funcionalidades de esta forma:

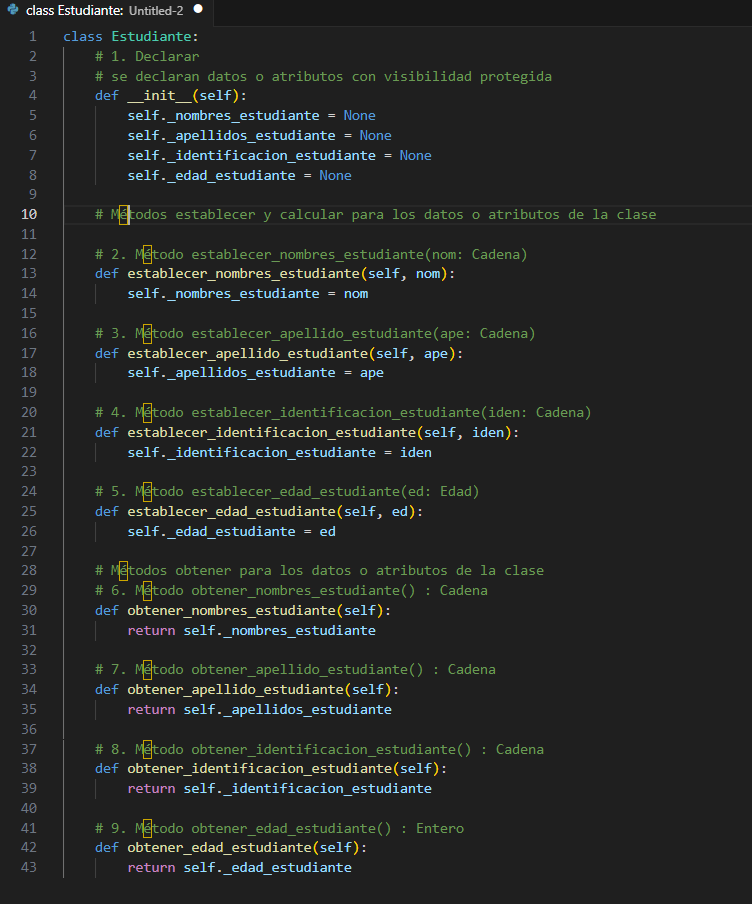


## **Herencia múltiple**

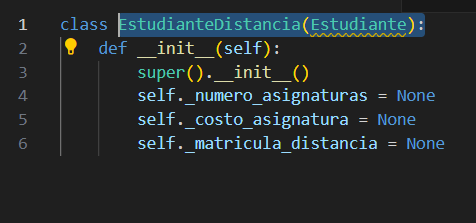
Python soporta la herencia múltiple, métodos y atributos de más de un padre. En caso de heredar atributos o métodos con el mismo nombre, Python dará prioridad al posicionado más a la izquierda en la declaración.

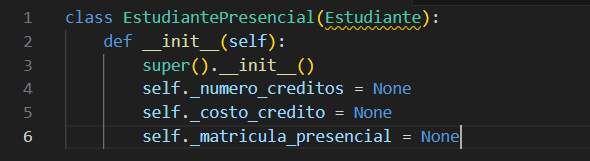


Este es un ejemplo de la clase Padre llamada **Estudiante** que hicimos en clases:



Y sus clases hijas que son **EstudianteDistancia** y **EstudiantePresencial:**



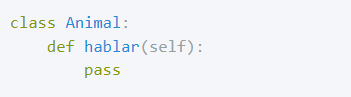


Como podemos darnos cuenta de una diferencia notable es la sustitución de la palabra "extends" por unos parentesis donde lleva el nombre de la clase Padre.

El polimorfismo visto desde el punto de vista de Python es complicado de explicar sin hablar del duck typing, por lo que te recomendamos la lectura.

Al ser un lenguaje con tipado dinámico y permitir duck typing, en Python no es necesario que los objetos compartan un interfaz, simplemente basta con que tengan los métodos que se quieren llamar.]

Una manera sencilla de explicarlo es mediante este ejemplo donde tenemos una clase llamada “Animal”



Y dos clases llamada “Perro” y “Gato” que heredan de la clase “Animal” implementando el método hablar.

Tal como mencione anteriormente Python se basa en “tipificación por comportamiento” o “duck typing” es un tipo de enfoque, no es necesario especificar el tipo exacto. Usará el método, si un objeto da una definición de ese método, todavía puede usarlo como su comportamiento. Puedes trabajar con objetos de diferentes herencias que exigen implementar el mismo comportamiento sin conocer el tipo en tiempo de ejecución.

Python es un lenguaje de tipado dinámico. Se puede trabajar con objetos de diferentes clases que implementen el mismo método sin necesidad de conocer su tipo exacto en tiempo de compilación.