

Segundo Examen de POO - I

Enunciado completo: Gestión de vehículos con herencia múltiple y conceptos avanzados en Python

Objetivo:

Crear un sistema que gestione diferentes tipos de vehículos, aplicando conceptos como clases abstractas, herencia múltiple, métodos mágicos, funciones lambda, filter(), comprensiones de listas, bucles y condicionales.

1. Clases y herencia:

- **Clase abstracta Vehiculo:**
Sin atributos, que tenga un método abstracto descripcion() y un método __str__() que indique que es una instancia de Vehiculo.
 - **Clase VehiculoTerrestre:**
Hereda de Vehiculo y tiene un atributo marca. Implementa el método descripcion() para devolver una cadena que indique la marca del vehículo.
 - **Clase VehiculoAereo:**
Hereda de Vehiculo y tiene atributos modelo y marca. Implementa descripcion() para devolver detalles del modelo y la marca.
 - **Clase VehiculoMixto:**
Hereda de Vehiculo mediante herencia múltiple: VehiculoTerrestre y VehiculoAereo.
Tiene atributos marca, modelo y tipo (que puede ser 'terrestre', 'aéreo' o 'ambos').
Implementa descripcion() para describir sus atributos.
Redefine el método __str__() para mostrar información combinada y el método __add__() para fusionar con otro vehículo VehiculoMixto, retornando un mensaje de unión.
-

2. Creación de objetos:

- Crear una lista vehiculos que contenga 2 objetos de VehiculoTerrestre, VehiculoAereo y VehiculoMixto, con atributos variados. Asegúrate de que todos tengan atributos marca y modelo según corresponda.

3. Filtrar vehículos:

- Usando filter() y función lambda, filtrar solo los vehículos cuya marca sea 'Toyota'.
- Sin usar hasattr(), asume que estos objetos tienen marca.

4. Listas por comprensión:

- Crear una lista marcas_modelos usando comprensión de listas, que contenga las marca de los vehículos filtrados.

5. Mostrar detalles con bucles:

- Recorrer en un bucle los vehículos filtrados, mostrando por pantalla el resultado de __str__() y descripcion() de cada uno.

6. Condicionales:

- Verificar si la lista filtrada está vacía y, en ese caso, mostrar un mensaje indicando que no se encontraron vehículos con esa marca.
- Si no, mostrar las marcas en la lista.

7. Operación entre objetos:

- Crear dos objetos VehiculoMixto con diferentes atributos.
- Fusionarlos usando __add__() y mostrar el resultado.

Notas adicionales:

- La clase VehiculoAereo ahora debe tener atributos modelo y marca.
- El método __str__() en VehiculoMixto debe mostrar información combinada de marca, modelo y tipo.
- El método __add__() en VehiculoMixto debe retornar un mensaje de unión si ambos objetos son del tipo VehiculoMixto.