¿Qué es una agregación en Python?

La agregación es una relación 'todo-parte' donde una clase contiene a otra, pero ambas pueden existir de forma independiente. Si se destruye el objeto contenedor, el contenido puede seguir existiendo.

Ejemplo:

```
class Motor:
    def encender(self):
        print("Motor encendido")

class Auto:
    def __init__(self, motor):
        self.motor = motor

mi_motor = Motor()
mi_auto = Auto(mi_motor)
del mi_auto
mi_motor.encender()
```

¿Qué es una composición en Python?

La composición es una relación 'todo-parte' fuerte. Si se destruye el objeto contenedor, las partes también se destruyen. El objeto contenido no puede existir sin el contenedor.

Ejemplo:

```
class Motor:
    def encender(self):
        print("Motor encendido")


class Auto:
    def __init__(self):
        self.motor = Motor()


mi_auto = Auto()
mi_auto.motor.encender()
```

¿Qué es una relación de dependencia en Python?

Es una relación temporal donde una clase usa otra clase, por ejemplo, como argumento en un método. No hay una relación estructural duradera.

Ejemplo:

```
class Impresora:
    def imprimir(self, documento):
        print(f"Imprimiendo: {documento}")

class Usuario:
    def enviar_a_imprimir(self, impresora, texto):
        impresora.imprimir(texto)

impresora = Impresora()
usuario = Usuario()
usuario.enviar_a_imprimir(impresora, "Hola mundo")
```

¿Qué es una herencia en Python?

La herencia es cuando una clase (subclase) hereda atributos y métodos de otra clase (superclase).

Es una relación 'es un tipo de'.

Ejemplo:

```
class Animal:
    def hablar(self):
        print("Hace un sonido")

class Perro(Animal):
    def hablar(self):
        print("Guau")

mi_perro = Perro()
mi_perro.hablar()
```