Preguntas de análisis

1. Si deseo modificar el mensaje del método ruido al crear un objeto Pajaro sin alterar la clase Animal ¿Qué debo agregarle al código? (Por ejemplo, al llamar el método ruido imprima, cantar y silbar).

R//= Para lograr esto sin modificar la clase base (Animal), puedes sobrescribir el método ruido en la clase Pajaro. Así, cuando crees un objeto de tipo Pajaro, este usará el método ruido de su propia clase en lugar del método heredado de Animal.

2. Si se crea una nueva clase Pez, y no se definen nuevos métodos, constructor y atributos. ¿Qué constructor tendrá esta clase, qué argumentos recibe? ¿Qué otros métodos y atributos tendrán estos mismos?

R//= Si no se definen nuevos métodos ni atributos en la clase Pez, esta clase heredará todos los métodos y atributos de su clase padre (Animal).

3. ¿Qué ocurre con el atributo nombre y edad de la clase SerVivo, al momento de definirse en la Clase Animal? ¿Cómo cambiaría el código del constructor para que estos atributos sean el mismo?

R//= Si los atributos nombre y edad están definidos en la clase SerVivo y luego en la clase Animal, entonces Animal sobrescribe estos atributos, pero si no se maneja correctamente, puede haber ocultación o duplicación de estos atributos.

Si se quiere que nombre y edad sean los mismos en ambas clases, se puede asegurar de que la clase Animal reciba estos valores a través de su constructor y los pase a la clase base SerVivo utilizando super().

4. En la clase Animal se sobrescribieron los métodos setNombre y getNombre, ¿Cómo modificaría estos métodos para que su funcionamiento no oculte algún valor de la clase padre? ¿Podría plantearse esta solución usando super()?

R//= Para evitar que los métodos de Animal oculten los métodos de la clase padre (SerVivo o similar), puedes usar super() para invocar los métodos de la clase base y mantener su funcionalidad mientras puedes añadir o modificar algo adicional.

5. El atributo totalCreados de la clase SerVivo ¿es heredado por las demás clases? ¿En este caso ocurre ocultación de los atributos al definirlo de nuevo en las clases hijas?

R//= Sí, el atributo totalCreados de la clase base SerVivo es heredado por las clases hijas. Si en alguna clase hija se define nuevamente este atributo, ocurrirá ocultación, es decir, el atributo definido en la clase hija "ocultará" el atributo heredado de la clase base.

6. ¿Los métodos getTotalCreados sobrescriben al metodo de su padre?

R//= Sí, si una clase hija define un método con el mismo nombre que un método de la clase base, este sobrescribe el comportamiento del método en la clase base. Esto ocurre si el método en la clase hija tiene la misma firma (nombre y parámetros).

7. ¿Qué métodos hereda una clase que hereda de la clase Persona?

R//= Una clase que hereda de Persona heredará todos los métodos públicos y protegidos de la clase Persona. Esto incluirá tanto los métodos que están explícitamente definidos en Persona como

aquellos que se implementan en sus clases intermedias si las hay.

8. ¿Qué tipo de objetos podrían pasársele al método aduenarAnimal de la clase Persona?¿Se le puede pasar un objeto serVivo?

R//= El tipo de objeto que se le puede pasar al método aduenarAnimal dependerá de su implementación. Si el método está diseñado para aceptar objetos de tipo Animal, también puede aceptar cualquier objeto que sea una subclase de Animal (como Pajaro, Perro, etc.).

Si aduenarAnimal está diseñado para aceptar un objeto SerVivo, entonces sí, un objeto SerVivo se podría pasar, pero esto depende de la relación de clases que tenga la jerarquía (por ejemplo, si Animal hereda de SerVivo).

9. ¿Cómo podría obtener la cantidad de seres vivos que se creen (objetos creados)?

R//= Puedes mantener un contador de los objetos creados dentro de la clase base (SerVivo) y actualizarlo cada vez que se crea una nueva instancia.

10. Si se define el siguiente método en la clase Perro:

```
def getRaza(self, tipo):
    return self._raza + " " + tipo
```

¿Qué ocurre con el método getRaza de la clase Animal? ¿Este método se sobreescribe o se sobrecarga?

R//= Este método se sobreescribe. Si Perro es una subclase de Animal y define un método con el mismo nombre (getRaza), la implementación del método en Perro reemplazará (sobreescribirá) la implementación heredada de Animal.

Si, en cambio, getRaza tuviera una firma diferente (número o tipo de parámetros), sería una sobrecarga del método, pero eso no es lo que está sucediendo en este caso.