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// Sobreescribiendo el método ruido en la clase Pajaro, de la siguiente manera:

```
def ruido(self):
    return "cantar y silbar"
```

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// La pregunta no dice explícitamente que sea una subclase de ninguna otra (pero suponiendo que es sublase de animal como las demás), heredaría los métodos de instancia y clase de Animal, además de los atributos de clase e instancia del mismo.

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// La clase animal define nuevamente los atributos nombre y edad que venían heredados desde la clase SerVivo, para evitar esto, se debe añadir super().__init__ en el constructor para que sean el mismo.

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

R// Añadir super(). Al retorno de los métodos setNombre y getNombre es la mejor solución para evitar que los métodos oculten algún valor de la clase padre

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// El atributo es heredado por las demás subclases, por que el "_" significa que el atributo tiene modificador protected y esto permite que sus hijos puedan acceder a el, si ocurre ocultación por que hay clases hijas que re definen el atributo dentro de su propia clase sin usar super().

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

R// Si al poseer el decorador @classmethod se convierte en un método de clase y sobreescribe el método de la clase padre

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

R// Hereda los métodos de la clase persona y los métodos de la clase padre de persona (SerVivo)

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

R// Se le puede pasar cualquier objeto que posea el método ruido()

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

R// Para obtener el numero total de seres vivos (creados de la clase SerVivo y sus subclases), se puede acceder al atributo de clase _totalCreados, ya que sus subclases siguen incrementando el contador de la clase padre.

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// El método se sobreescribe en la subclase (ocultando el método de la clase padre), en Python no existe la sobrecarga de métodos como ocurre normalmente en java.