Taller 5 Python Ejercicio 2.

Emmanuel Valencia Lopera.

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).**
   En la clase Pajaro se debe agregar el método de instancia:
   def ruido(self):
   print("cantar y silbar")
   Ocultando así el método que ya estaba en la clase padre (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?**
   Esta clase heredaría el inicializador de la clase Animal que recibe como argumentos: nombre, edad, raza; dentro de la práctica (en Python todo es público pero considerando que se respeta la privacidad) se heredan los métodos set y get de los atributos: _nombre y _raza. Además, se heredan los métodos: caminar, correr, ruido, getTotalCreados; y también se heredan los métodos set y get de el atributo _edad que están definidos en la clase SerVivo.

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?**
   Se debería quitar la privacidad a los atributos _nombre y _edad tanto en la clase SerVivo como en la clase Animal. Así dentro en la práctica estos atributos hacen referencia al mismo espacio de memoria.

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()?**
   Si, se podría usar el super() en los métodos getNombre y setNombre para llamar estos mismos métodos de la clase SerVivo.
   Agregando lo siguiente dentro de estos métodos get y set en la clase Animal respectivamente:
   super().getNombre()
   super().setNombre()

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?**
   Si se toma en cuenta la privacidad no se hereda, pero dejando esto de lado este atributo de clase si se hereda y si este se llega a definir en las demás clases, este se oculta ya que es un atributo de clase que pertenece a cada una de las clases.

6. **¿Los métodos getTotalCreados sobrescriben al método de su padre?**
   Si lo sobrescribe ya que en Python un método de clase si es sobrescrito.

7. **¿Qué métodos hereda una clase que hereda de la clase Persona?**
   Hereda los métodos aduenarAnimal, getTotalCreados, y por cadena los métodos get y set de los atributos _nombre y _edad.

8. **¿Qué tipo de objetos podrían pasársele al método aduenarAnimal de la clase Persona? ¿Se le puede pasar un objeto serVivo?**
   Se pueden pasar cualquier tipo de objetos pero es ideal y para que no hayan errores de código que se pasen a este método objetos de la clase Animal o sus subclases.

9. **¿Cómo podría obtener la cantidad de seres vivos que se creen (objetos creados)?**
   Se debería añadir lo siguiente al inicializador de Animal:
   super().__init__(nombre,edad)
   Así se contarían en el inicializador de SerVivo todos los objetos creados, se puede obtener esta cantidad consultando el atributo _totalCreados.

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?
    Este método se sobrescribe ya que en la clase hija Perro se coloca un método que tiene la misma firma del método getRaza de la clase padre Animal.