Taller Poo en Python

Santiago Barrientos Medina

Universidad Nacional de Colombia

Programación orientada a objetos

Jaime Alberto Guzmán

Grupo 2

Facultad de Minas

Ingeniería en sistemas en informática

SOLUCION

1). Para hacer esto sin modificar la clase padre que es animal, podemos sobreescribir el método de la siguiente manera:

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

2).al crear la nueva clase pez , tendrá el constructor de animal que es el que se hereda, es decir, al nosotros no ponerle nada no habría sobreescritura asi que todo lo heredaría y quedaría con el constructor de animal que recibe : **nombre, edad y raza**, además de eso contaria con los atributos de Animal y con sus métodos

3). Lo que ocurre con el atributo nombre y edad de la clase ser vivo es que en la clase animal, le damos una redefinición, es decir, reescribimos el valor de nombre de la clase padre, lo que yo haría en el constructor para que los atributos sean el mismo, sería:

```
 def __init__(self, nombre, edad, raza, pelaje):
        super.__init__(nombre, edad, raza)
        self._pelaje = pelaje
        Gato._totalCreados += 1
```

Aquí no sobreescribo el valor sino que llamo al constructor de la clase padre

4). Sí podría plantearse la solución a este problema, para que no haya ocultamiento de la clase hija a la clase padre, esto de la siguiente manera :

```
def setNombre(self, nombre):
        super().setnombre(nombre)
 def getNombre(self):
        super().getNombre()
```

con esto estaríamos llamando al método de padre y sabríamos que estamos usando los atributos de él, sin de pronto cometer un ocultamiento que nosotros no queramos

5). Los atributos de clase son propios de cada clase, estos no se heredan pero pueden ser accesados desde el super de cada clase o con el llamado desde la clase. En este caso no ocurriría ocultación si yo lo defino en las clases hijas ya que como el atributo de clase es propio de cada clase, cada una tendría el suyo propio

6).los métodos get total creados si sobreescriben a los métodos del padre, y le permiten a la clase hija poseer sus propios métodos con el mismo nombre pero que hagan cosas diferentes

7).los métodos que hereda una subclase de persona son: los que persona ya heredo de ser vivo al ser una subclase de esta pero es importante aclarar que hereda aquellos que no hayan sido sobreescritos porque dado esto, heredaría los métodos como los sobreescribio persona y aquello que no seria heredado con total normalidad, además también heredaría los que nuevos que haya creado persona, es decir , los que vienen de la clase persona que los podemos ver, que bien pueden ser nuevos o sobreescritos y los que no podemos ver que vienen de la clase padre de persona pero que persona no haya sobreescrito

8). Se le podría pasar cualquier objeto que tenga o haya heredado el método ruido , debemos recordar que los objetos se pasan por referencia en pyhton , es decir, tendríamos un apuntador al objeto , y el problema estaría al momento de llamar al método hacer ruido, la clase ser vivo ni lo tiene definido ni lo ha heredado asi que el código en este punto explotaría si le pasamos ser vivo como parámetro

9). Como técnicamente el total creados de ser vivo es un atributo privado , no podemos accesar tan fácil a el, una manera seria con el método get de este atributo, haciendo un llamado a este desde la clase desde cualquier otra clase

10). Aquí habría sobreescritura porque el método tiene el mismo nombre y en pyhton esto es lo único necesario para sobreescribir el método