Respuestas.

1. Puedes modificarlo, sobrescribiendo el método dentro de la clase Pajaro de la siguiente manera:

```
def ruido(self):
    return “cantar y silbar”
```

2. Si se crea una clase Pez y no se definen métodos, constructores ni atributos, este tomara los atributos, los métodos y el constructor de su padre que en este caso es la clase Animal.
   Por tanto recibe el constructor:

```
def __init__ (self, nombre, edad, raza):
  self._nombre = nombre
  self._edad = edad
  self._raza = raza
  Animal._totalCreados += 1
```

   También recibe los métodos:

   setRaza, getRaza, setNombre, getNombre, caminar, correr, ruido y el metodo de clase getTotalCreados

3. Al momento en que la clase Animal hace uso de los atributos _nombre y _edad, se considera una redefinición de atributos, por tanto no son compartidos con la clase SerVivo. Para que estos atributos sean el mismo se debe usar el super de la siguiente manera:

```
def __init__ (self, nombre, edad, raza):
    super().__init__(nombre, edad)
    self._raza = raza
    Animal._totalCreados += 1
```

4. Se puede utilizar el super() dentro de los métodos para acceder a los valores de la clase SerVivo, de la siguiente manera:

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

5. Si se hereda debido a la falta de privacidad de Python, pero como _totalCreados es redefinido en cada clase, siempre es ocultado, por tanto para acceder al _totalCreados de la clase SerVivo, se usa el supero o el nombre de la clase

6. Si hay sobreescritura, pues al momento de llamar el metodo se toma en cuenta desde que instancia es llamado, es decir, si se llama desde un objeto de clase Pajaro, estaremos accediendo a _totalCreados de la clase Pajaro
7. Una clase que hereda de la clase Persona recibe los siguientes métodos:
    a. __init__(self, nombre, edad) -> Es decir, el constructor de persona
    b. setNombre
    c. getNombre
    d. setEdad
    e. getEdad
    f. getTotalCreados
    g. aduenarAnimal
8. Se le pueden pasar objetos de tipo Animal, pero no se le puede pasar directamente un objeto de tipo SerVivo, pues esta clase no cuenta con el método ruido, por tanto, produciría un error.
9. Se puede obtener usando **print(SerVivo.getTotalCreados()**
10. El método se sobrescribe, ya que en Python no hay sobrecarga directa de métodos, por tanto, si se trata de ejecutar el método getRaza desde la clase Perro sin argumentos, pasaría a dar error.