Taller 5 - Java -1

☑ Completado	☐
Cursos	👨🏻‍💻 POO
Fecha de entrega	@10 de diciembre de 2024

1. Para este fin debemos reescribir el metodo de pitar que fue declarado en la clase Auto por lo tanto escribimos las siguientes lineas de codigo en la clase Moto:

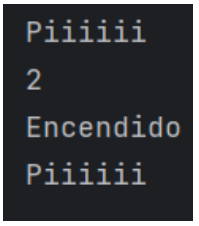
```
@Override
public void pitar() {
    System.out.println("Las motos no pitan");
}
```

2. El problema que se presenta es relacionado con el constructor de la clase Moto, ya que java automáticamente la momento de hacer la herencia de la clase Motoneta trata de llamar un constructor por defecto de su padre Moto, pero Moto no tiene constructor sin parámetros, es un problema de inicialización de el objeto ya que requiere la información del padre + la información especifica del objeto hijo, y no se tiene la información del padre si ese no llama almenos un constructor, la forma de que funcione es crear un constructor sin parámetros o indicar la llamada del constructor del padre con super.
3. Si si es posible, de hecho con esa definición estaríamos sobrescribiendo el método arrancar que viene de la clase padre (Auto), es posible ya que coinciden los retornos (al ser void no retorna nada), además no son estáticos, y el decorador el igual o mas publico que el método que viene del padre.
4. Porque hereda el método pitar de la clase Auto.
5. No hay necesidad de hacer ningún cambio, ese valor puede ser accesado por cualquier instancia hija de Auto, o se puede llamar la clase Auto

explícitamente:

```
sout(Auto.numAutos)
```

6. Porque no es publico desde el paquete2 donde se encuentra la clase ObjTaller5, no tiene decorador de accesibilidad por lo tanto solo tiene visibilidad de paquete.
7. Porque es heredada de su padre, la clase Auto.
8. Otra vez que no estoy seguro de la pregunta



En mi caso si puedo usar el método pitar, y no tengo modificaciones en la clase Auto...

9. No imprime nada, solo estas retornando un entero, y retornar un entero no hace que este se imprima.
10. Para bus, primero añade modelo, porque le falta esta variable, a moto le faltaría capacidad, si hablamos en que los 2 tuvieran los mismos atributos. Además estos atributos son privados, si no te importa acabar con el encapsulamiento cambia los modificadores a default o public. Si si te interesa el encapsulamiento deberías entonces crear los métodos crud respectivos para cada atributo.