Taller 5 Java Ejercicio 2 Sergio Morales Martínez

1. Si deseo modificar el mensaje del método pitar al crear un objeto moto sin alterar la clase Auto, ¿qué debo agregarle al código? (Por ejemplo, al llamar el método pitar imprima: Las motos no pitan).
   R// Se puede sobrescribir un el método pitar, dentro de la clase moto se debería de agregar lo siguiente:
   ```
   @Override
   public void pitar() {
           System.out.println("Las motos no pitan");
           }
   ```

2. 2. Suponga que se agrega una nueva clase al código, class Motoneta, y esta hereda de la clase Moto, ¿evidencia algún problema? ¿Por qué?
   R// Si, los problemas que veo son: Se heredara el el pitido de la Moto, lo cual no es lo mas adecuado para una motoneta, además de esto, los atributos placa y modelo en Moto se declaran como privados, por lo que la motoneta no tendría acceso a estos, por lo que seria necesario declararlos de vuelta.

3. Suponga que se definió el método:
   public void arrancar() { System.out.println("Arrancando"); } en la clase Moto, ¿es posible sobrescribir el método? ¿Por qué?
   R// Si, porque el método arrancar de clase auto es publica, por lo que se esta heredando. Gracias al polimorfismo este método puede ser sobrescrito en la clase

4. 4. En la línea 13 de la clase moto, ¿Por qué puedo utilizar el método pitar?
   R// Porque tanto el atributo como el método pitar() de la clase Auto se están heredando. Gracias a la herencia la moto puede utilizar este método.

5. Haciendo una pequeña modificación a la clase Auto y utilizando la variable num_autos, sin modificar su modificador de acceso, ¿cómo puedo obtener el número de autos creados desde la clase ObjTaller5H?
   R// Se puede añadir un método estatico getnum_autos el cual se encargue de retornar el numero de autos, este al ser publico se podría llamar desde el main de la manera:
   System.out.println(Auto.getNumAutos());

6. En la línea 7 de la clase ObjTaller5H, ¿Por qué no puedo utilizar el método adelantar, si este fue heredado?
   R// Esto se debe a que tiene el modificador por defecto (package) el cual admite visibilidad del método a las subclases, pero solo dentro del mismo paquete. Por lo que al estar la clase moto fuera del paquete1, esta no puede acceder a el, y por ello no se puede utilizar.

7. En la línea 8, ¿por qué es posible utilizar el método arrancar si la clase Bus no lo define?
   R// Porque el método arrancar lo hereda la clase Bus de la clase Auto, y al ser este público, es totalmente accesible.

8. En la línea 9 de la clase ObjTaller5H, ¿por qué no puedo utilizar el método pitar, si este fue heredado?
   R// El método pitar si se puede utilizar, e imprime “Piiiiii”

9. En la línea 10 de la clase ObjTaller5H, ¿qué imprime el método getVelocidad()? ¿Por qué?
   R// Imprime 10, esto debido a que el método al ejecutarse por el this, hace referencia a la clase dueña del método, por lo que imprime en su lugar el atributo de velocidad de la clase auto. Esto se podría solucionar sobrescribiendo este método en la clase moto.

10. Si quisiera obtener el valor de la placa de las clases Moto y Bus, además de su modelo y capacidad respectivamente, ¿Que debo agregar al código?
    R// En cuanto a la place, seria necesario agregar un método GET de este atributo ya que estos están encapsulados como privados, en cuando a su modelo y capacidad, seria necesario declarar el atributo modelo a la clase bus, junto a su respectivo método getter, de igual forma con el atributo capacidad en la clase moto.
    El código quedaría de la siguiente forma.

```
 private String placa;
private int capacidad;

public int getCapacidad() {
        return(this.capacidad);
}
public String getPlaca() {
        return(this.placa);
}
```