Solución taller 5 Java.

Programación orientada a objetos. Grupo 1

Alejandro Pérez Barrera, alperezba@unal.edu.co, C.C. 1023629729

1. Se le agregaría a la clase Moto su propio método pitar, de esta manera no se invocaría el método de la superclase Auto, sino el de la moto, donde se puede especificar la acción por parte de la moto.
   En la clase Moto:

```
public void pitar(){
    System.out.println(x:"Las Motos no pitan");
}
```

2. Si, hay problemas con la inicialización de la nueva clase.
3. Si. Es posible que ocurra algo llamado *Override*, donde como tanto la subclase como la superclase tienen, cada una, un método con la misma firma. En estos casos, el método de la subclase recibe la prelación, y se ejecuta en lugar del método de la superclase, efectivamente “Sobrescribiéndola”.

4. Porque este método es heredado de la clase Auto.

5. Es posible hacer uso de un getter para este atributo, o, y aprovechando el hecho de que este sea público,

6. Porque este método cuenta con el modificador de acceso predeterminado, por lo que no se puede acceder desde otro paquete, aun si se trata de una subclase la que lo intenta acceder. Para solucionarlo el método debe de ser público.

7. Porque el método arrancar es un método público, heredado de la superclase Auto.

8. Si se puede utilizar, el método pitar es heredado de auto.

9. Imprime 10, la velocidad predeterminada la clase auto, porque moto no tiene su propio getter que se sobreponga al de la superclase.

10. Se puede implementar sea el método toString() para el bus y la moto, o utilizar getters y setters para obtener los datos por separado.

    Con toString() para el bus:

```
public String toString(){
      return "Placa: "+this.placa+", Capacidad: "+
        this.capacidad;
    }
```

    Con toString() para la moto:

```
public String toString(){
      return "Placa: "+this.placa+", Modelo: "+
        this.modelo;
    }
```

    Imprimen:

```
Placa: ABC345, Capacidad: 20
Placa: XYZ123, Modelo: 2019
```

    Primero el bus, segundo la moto.