STIVEN SANTIAGO ROSERO QUEMAG
👩‍💻PROGRAMACIÓN ORIENTADA A OBJETOS - UNALMED 2024 -2👩‍💻
Preguntas de análisis
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)
Tenemos que sobrescribir el método pitar en la clase Moto. Cuando hacemos el llamado al método pitar con la anotación @Override, estamos indicando que queremos sobrescribir el método. De esta manera, podemos proporcionar la implementación deseada para el método pitar en la clase Moto, imprimiendo el mensaje que queremos cuando creamos una moto."

2. Suponga que se agrega una nueva clase al código, class Motoneta, y esta hereda de la clase
Moto, ¿evidencia algún problema? ¿Por qué?
si como la clase Motoneta hereda de la clase Moto, también heredará el método pitar() de la clase Moto, que es el que sobrescribe el método pitar() de la clase Auto.
Entonces, en este caso, la clase Motoneta heredará el método pitar() de la clase Moto, y no el método pitar() de la clase Auto.

3. Suponga que se definió el método:
public void arrancar() {
System.out.println("Arrancando");
}
¿Es posible sobrescribir el método? ¿Por qué? sí,
Es posible sobrescribir el método arrancar() en la clase Moto, porque el método existe en la superclase Auto y puede ser sobrescrito en la subclase Moto.
en la clase Moto
4. En la línea 13 de la clase moto, ¿Por qué puedo utilizar el método pitar?

porque este método es heredado de la clase Auto, que es la superclase de Moto.
Al heredar de la clase Auto, la clase Moto tiene acceso a todos los métodos y propiedades públicas y protegidas de la clase Auto, incluyendo el método pitar().

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?
Se puede obtener el número de autos creados creando un método que retorne el valor de Auto.num_autos y luego llamando a ese método desde el main.
6. En la línea 7 de la clase ObjTaller5H, ¿Por qué no puedo utilizar el método adelantar, si este fue heredado?
Porque la clase Moto no tiene un inicializador o constructor que llame al método adelantar() de la clase Auto
7. En la línea 8, ¿por qué es posible utilizar el método arrancar si la clase Bus no lo define?
Porque en la clase Bus, se llama implícitamente al constructor de la clase Auto, lo que permite utilizar el método arrancar()
8. En la línea 9 de la clase ObjTaller5H, ¿por qué no puedo utilizar el método pitar, si este fue

heredado?
Porque antes ya habíamos sobrescrito el método pitar() en la clase Moto, por lo que no se puede acceder al método pitar() original de la clase Auto
9. En la línea 10 de la clase ObjTaller5H, ¿qué imprime el método getVelocidad()? ¿Por qué?
imprime 30 porque es un atributo propio ya estipulado 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?
En la clase Moto tenemos que agregar lo siguiente

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

public String getModelo() {
   return this.modelo;
}
```

En la clase Bus tenemos que agregar lo siguiente

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

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

luego en ObjTaller5H agregamos

```
System.out.println("Placa de la moto: " + moto.getPlaca());
System.out.println("Modelo de la moto: " + moto.getModelo());

System.out.println("Placa del bus: " + bus.getPlaca());
System.out.println("Capacidad del bus: " + bus.getCapacidad());
```