Programación Orientad a Objetos

Taller 5 Java Parte 2

Santiago Abelardo Salcedo Rodríguez

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).

R= Tendriamos que redefinir en la clase moto el método pitar de esta manera

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

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= No Habria alguna razón en especifico pero hay que tener en cuenta que 2 atributos de moto son privados y por tanto no se heredarían. Acelerar si se heredaría y también lo que hereda moto de auto,

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 es posible sobre-escribir el método, primero, el método puede ser accedido por la sub-clase moto y por tanto si se estaría sobre-escribiendo (aunque faltaría el decorador @override pero este no afecta) debido a que la firma del método es la misma y el tipo de retorno es compatible (en particular es el mismo) void.

4. En la línea 13 de la clase moto, ¿Por qué puedo utilizar el método pitar?

R= Pues moto hereda de su clase padre el método pitar() entonces no habría ningún inconveniente en utilizarlo, sabemos que se hereda pues este no es privado

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= Sin modificar el modificador de acceso ni cualquier otra clase podemos modificar el método getVelocidad() de tal manera que en objTaller5H cuando sea llamado nos de el numero de autos creados (aunque no tenga mucho sentido llamarlo getVelocidad()) una posible solución es:

Public int getVelocidad(){return(num_autos);} asi cuando se ejecute la línea 10 de la clase objTaller5H se devolverá el numero de autos

6. En la línea 7 de la clase ObjTaller5H, ¿Por qué no puedo utilizar el método adelantar, si este fue heredado?

R= Debido a que Auto pertenece al paquete1 y el modificador de acceso del método adelantar esta por default entonces no son visibles fuera del paquete donde se declaran, aunque lo hereden moto y bus como no se puede acceder entonces no lo pueden utilizar

7. En la línea 8, ¿por qué es posible utilizar el método arrancar si la clase Bus no lo define?

R= Pues es heredado de su clase padre y este método si tiene el modificador de acceso public entonces puede ser accedido asi sean de paquetes diferentes, por tanto no hay problema

8. En la línea 9 de la clase ObjTaller5H, ¿por qué no puedo utilizar el método pitar, si este fue heredado?

R= El método si puede ser ejecutado con normalidad pues su modificador de acceso es publico

9. En la línea 10 de la clase ObjTaller5H, ¿qué imprime el método getVelocidad()? ¿Por qué?

R= El método no IMPRIME nada pues este solo retorna el valor de la velocidad y pues este si se ejecutaria ya que moto lo hereda de Auto pero nada mas retorna el valor ,no lo imprime.

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= Otra manera es eliminar el atributo placa de la definición de las clases moto y bus pero agregarlo en la clase auto para que ambas la hereden junto con su getter, la cuestión es que placa tiene el nivel de accesibilidad private y entiendo que por esto preferí no hacerlo ya que se prefiere un encapsulamiento mas privado de este atributo, entonces seria asi:

En la clase Moto:

```
public String getModelo(){

return modelo;

}

public String getPlaca(){

return placa;

}
```

En la Clase Bus:

```
public String getPlaca(){

return placa;

}

public int getCapacidad(){

return capacidad;

}
```