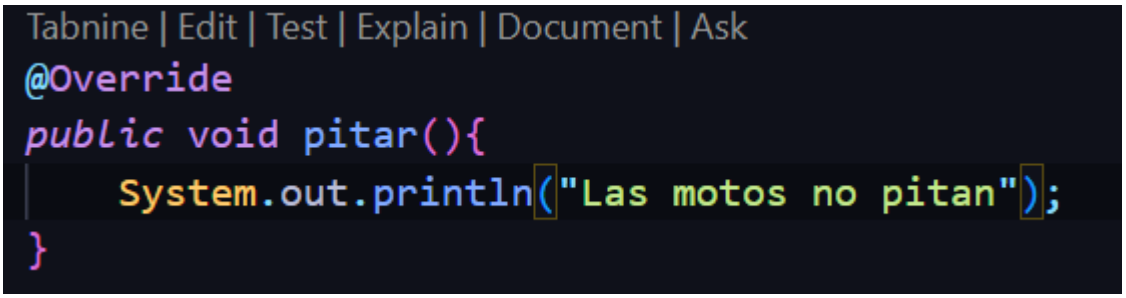
Preguntas de análisis

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).

- Se debería sobrescribir el método usando la palabra clave @override, esto para que al momento de definir la clase pitar podamos hacer que el código en vez de arrojar por consola el mensaje que hereda de la clase Auto, pueda escribir "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?

- No se evidencia ningún tipo de problema, tal vez inmediatamente, pero si se quiere acceder a atributos como placa o modelo, sería totalmente imposible, ya que Java no hereda ningún método o atributo que tenga el encapsulamiento privado.

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?

- Se puede sobrescribir, ya que al ser un método público se puede heredar y sobrescribir sin ningún tipo de inconveniente o error en la compilación.

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

- Esto se puede porque la clase Moto hereda de Auto, y pitar está definido como público en la clase. Por eso se puede hacer uso de 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?

- Al ser un atributo de clase se puede acceder desde el nombre de la clase Auto y hacer uso de ella sin ningún problema desde la clase ObjTaller5H.

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

- Porque el tipo de modificador de acceso es default, lo que quiere decir que solo se puede acceder desde el mismo paquete, Auto está en paquete 2 y las otras clases están en el paquete 1. Lo que hace que no pueda tener acceso a ese método y tampoco se está importando.

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

- Porque es heredado de Auto, con la única diferencia del punto 6 que en este caso, si se importa el paquete desde donde se encuentra la clase Auto.

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

- Aunque pitar es público, en este caso no se puede usar directamente porque se está intentando acceder a través de una instancia de Bus, y Bus está en un paquete diferente.

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

- Esto sucede porque el método getVelocidad() de la clase Auto accede al atributo de instancia velocidad definido en Auto, y no a los atributos velocidad de las clases Moto o Bus, debido a que no hay sobrescritura de atributos.

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?

- Agregar los respectivos getters y setters para poder modificar y acceder a estos valores.

Para la clase Moto:

```
Tabnine | Edit | Test | Explain | Document | Ask
public String getPlaca() {
    return this.placa;
}

Tabnine | Edit | Test | Explain | Document | Ask
public String getModelo() {
    return this.modelo;
}
```

Para la clase Bus: