Respuestas

1) Sobrescribir el método pitar en la clase Moto para personalizar su comportamiento. Esto se logra redefiniendo el método en la subclase:

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

2) No hay problemas directos en la herencia, pero es importante considerar que Motoneta también heredará las propiedades y métodos de Auto a través de Moto. Si no se define un constructor explícito en Motoneta, se invocará automáticamente el constructor de Moto.

3) Sí, es posible sobrescribirlo, ya que el método arrancar en la clase Auto no está declarado como final.

4) Porque la clase Moto hereda el método pitar de la clase Auto.

5) Accediendo directamente a la variable num_autos porque es static y pertenece a la clase Auto.

6) Porque el método adelantar tiene el modificador de acceso package-private (sin especificador) en la clase Auto, lo que lo limita a clases dentro del mismo paquete.

7) Porque el método arrancar es heredado de la clase Auto.

8) Sí se puede utilizar el método pitar. Lo que sucede es que, al ejecutarse el constructor de Moto, ya se invocó el método pitar, por lo que podría parecer redundante.

9) Imprime 10. El método getVelocidad de Auto utiliza el atributo velocidad de la clase base (no las versiones redefinidas en las subclases). Por lo tanto, devuelve el valor de velocidad de Auto, que es 10.

10) Agregar métodos getter en las clases Moto y Bus.