

Ejercicio de Programación Java: Gestión de Eficiencia Eléctrica

1. Introducción y Objetivos

Una tienda de electrónica desea realizar un estudio detallado sobre la eficiencia eléctrica de distintas marcas de electrodomésticos. Para llevar a cabo este análisis, se requiere el desarrollo de una aplicación en Java que implemente la estructura de clases y la lógica de negocio descrita a continuación.

2. Definición de Clases

A) Clase Electrodomestico (Abstracta)

Esta será la clase padre de la jerarquía.

- **Atributos:**
 - fabricante (String): Nombre del fabricante del electrodoméstico.
 - potencia (double): Potencia consumida por hora en kWh.
 - fechaCompra (LocalDate): Fecha de adquisición (formato por defecto yyyy-MM-dd).
 - **Constructores:**
 1. Constructor completo con parámetros para todos los atributos.
 2. Constructor parcial: Recibe todos los atributos excepto la fecha de compra, la cual se asignará automáticamente al día actual (LocalDate.now()).
 - **Métodos:**
 - getters y setters para todos los atributos.
 - getPrecioMedioDiario() (Abstracto): No recibe parámetros. Debe devolver el coste en euros del consumo diario basado en la potencia del aparato.
 - toString(): Devuelve una cadena con el valor de todos los atributos en una sola línea, separados por el carácter **.
-

B) Clase Lavadora

Hereda de la clase Electrodomestico.

- **Atributos adicionales:**

- aguaCaliente (boolean): Indica si el lavado se realiza con agua caliente.
 - **Constructores:**
 1. Constructor completo con parámetros para todos los atributos (propios y heredados).
 2. Constructor simplificado: Recibe solo potencia y fabricante.
 - La fechaCompra será la fecha actual.
 - aguaCaliente se inicializa por defecto a true.
 - **Métodos:**
 - getters y setters necesarios.
 - getPrecioMedioDiario(): Implementación del método abstracto.
 - *Lógica:* Si funciona con agua caliente, el consumo por hora (kWh) se incrementa en un **20%**.
 - *Nota:* Debe usar constantes para el cálculo. Debe lanzar la excepción especificada en el apartado (d)*.
 - toString(): Devuelve todos los atributos (incluido el nuevo) en una línea separados por **.
-

C) Clase Frigorifico

Hereda de la clase Electrodomestico.

- **Atributos adicionales:**
 - modoVacaciones (boolean): Indica si el aparato está en modo de bajo consumo.
- **Constructores:**
 1. Constructor completo con parámetros para todos los atributos.
 2. Constructor simplificado: Recibe solo potencia y fabricante.
 - La fechaCompra será la fecha actual.
 - modoVacaciones se inicializa por defecto a false.
- **Métodos:**
 - getters y setters necesarios.

- `getPrecioMedioDiario()`: Implementación del método abstracto.
 - *Lógica*: Si el modo vacaciones está activo, el consumo por hora (kWh) se reduce en un **10%**.
 - *Nota*: Debe usar constantes para el cálculo. Debe lanzar la excepción especificada en el apartado (d)*.
 - `toString()`: Devuelve todos los atributos en una línea separados por ******.
-

3. Gestión de Datos

Clase Tienda

Esta clase actuará como gestora de la colección de datos.

- **Atributos:**
 - `listaElectrodomesticos`: Un `ArrayList` donde se almacenarán los objetos de tipo `Lavadora` y `Frigorifico`.
 - **Requisitos:**
 - El `ArrayList` **no debe ser accesible** directamente desde fuera de la clase (encapsulamiento).
 - La clase debe contar con los métodos necesarios para realizar operaciones sobre la lista (añadir, listar, buscar, etc.).
-

Nota: El enunciado menciona una excepción en el apartado (d), asegúrese de verificar los requisitos de dicha excepción para implementarla correctamente en los métodos de cálculo de precio.