***TP JAVA***

**Cátedra:** Lenguaje de Programación Java

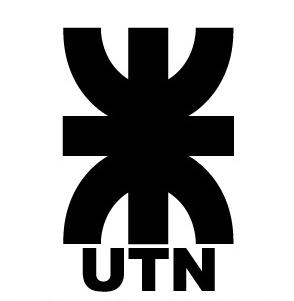
**Comisión:** 2

**Legajo:** 40207

Selle, Javier

**Base de Datos:** <https://github.com/javier1993/TPJavaBaseDeDatos.git>

**Web:** https://github.com/javier1993/TPJavaWeb.git



**Enunciado**

Crearemos una superclase llamada **Electrodoméstico** con las siguientes características:

Sus atributos son **precio base, color, consumo energético** (letras entre A y F) y **peso**. Indica que se podrán heredar.

Por defecto, el color sera blanco, el consumo energético sera F, el precioBase es de 100 U$S y el peso de 5 kg. Usa constantes para ello.

Los colores disponibles son blanco, negro, rojo, azul y gris. No importa si el nombre esta en mayúsculas o en minúsculas.

Los constructores que se implementaran serán:

Un constructor por defecto.

Un constructor con el precio y peso. El resto por defecto.

Un constructor con todos los atributos.

Los métodos que implementara serán:

·Métodos get de todos los atributos.

**comprobarConsumoEnergetico(char letra)**: comprueba que la letra es correcta, sino es correcta usara la letra por defecto. Se invocara al crear el objeto y no sera visible.

**comprobarColor(String color)**: comprueba que el color es correcto, sino lo es usa el color por defecto. Se invocara al crear el objeto y no sera visible.

**precioFinal()**: según el consumo energético, aumentara su precio, y según su tamaño, también. Esta es la lista de precios:

|  |  |
| --- | --- |
| Letra | Precio |
| A | 100 |
| B | 80 |
| C | 60 |
| D | 50 |
| E | 30 |
| F | 10 |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | |  |  | | --- | --- | | Tamaño | Precio | | 0-19 kg | 10 | | 20-49 kg | 50 | | 50-79 kg | 80 | | Mayor que 80 kg | 100 | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Crearemos una subclase llamada **Lavarropas** con las siguientes características:

Su atributo es **carga**, ademas de los atributos heredados.

Por defecto, la carga es de 5 kg. Usa una constante para ello.

Los constructores que se implementaran serán:

Un constructor por defecto.

Un constructor con el precio y peso. El resto por defecto.

Un constructor con la carga y el resto de atributos heredados. Recuerda que debes llamar al constructor de la clase padre.

Los métodos que se implementara serán:

Método get de carga.

**precioFinal()**:, si tiene una carga mayor de 30 kg, aumentara el precio 50 $, sino es así no se incrementara el precio. Llama al método padre y añade el código necesario. Recuerda que las condiciones que hemos visto en la clase Electrodoméstico también deben afectar al precio.

Crearemos una subclase llamada **Television** con las siguientes características:

Sus atributos son **resolucion** (en pulgadas) y **sintonizador TDT** (booleano), ademas de los atributos heredados.

Por defecto, la resolución sera de 20 pulgadas y el sintonizador sera false.

Los constructores que se implementaran serán:

Un constructor por defecto.

Un constructor con el precio y peso. El resto por defecto.

Un constructor con la resolución, sintonizador TDT y el resto de atributos heredados. Recuerda que debes llamar al constructor de la clase padre.

Los métodos que se implementara serán:

Método get de resolución y sintonizador TDT.

**precioFinal()**: si tiene una resolución mayor de 40 pulgadas, se incrementara el precio un 30% y si tiene un sintonizador TDT incorporado, aumentara 50 U$S. Recuerda que las condiciones que hemos visto en la clase Electrodomestico también deben afectar al precio.

(?) Tener en cuenta la realización de dos clases una para el color y otra para el consumo que permitan cargar los atributos en el electrodoméstico.

Ahora crea una clase ejecutable que realice lo siguiente:

Permita el alta de un electrodoméstico con todas sus características.

Haga un listado de los diferentes electrodomésticos, solicitando un rango de importes, y/o consumo energético y listar ordenado por la descripción del artefacto.

Permita la baja y modificación de electrodomésticos existentes.

Nota: la aplicación se realizará con el concepto de capas y con la capa de datos no necesariamente usando bases de datos.

**Capturas de Imágenes**

