Batería ejercicios Java POO

1) Crea una clase llamada Cuenta que tendrá los siguientes atributos: titular y cantidad (puede tener decimales).

El titular será obligatorio y la cantidad es opcional. Crea dos constructores que cumpla lo anterior.

Crea sus métodos get, set y toString.

Tendrá dos métodos especiales:

- ingresar(double cantidad): se ingresa una cantidad a la cuenta, si la cantidad introducida es negativa, no se hará nada.
- retirar(double cantidad): se retira una cantidad a la cuenta, si restando la cantidad actual a la que nos pasan es negativa, la cantidad de la cuenta pasa a ser 0.

_____

–Clase Cuenta

```
public class Cuenta {

    //Atributos
    private String titular;
    private double cantidad;

    //Constructores
    public Cuenta(String titular) {
        this(titular, 0); //Sobrecarga
    }

    public Cuenta(String titular, double cantidad) {
        this.titular = titular;
        //Si la cantidad es menor que cero, lo ponemos a cero
        if (cantidad < 0) {
            this.cantidad = 0;
        } else {
            this.cantidad = cantidad;
        }
    }

    //Metodos
    public String getTitular() {
```

```
    return titular;
}

public void setTitular(String titular) {
    this.titular = titular;
}

public double getCantidad() {
    return cantidad;
}

public void setCantidad(double cantidad) {
    this.cantidad = cantidad;
}

/**
 * Ingresa dinero en la cuenta,
 * solo si es positivo la cantidad
 *
 * @param cantidad
 */
public void ingresar(double cantidad) {
    if(cantidad > 0){
        this.cantidad += cantidad;
    }
}

/**
 * Retira una cantidad en la cuenta, si se quedara en negativo se quedaria
 * en cero
 *
 * @param cantidad
 */
public void retirar(double cantidad) {
    if (this.cantidad - cantidad < 0) {
        this.cantidad = 0;
    } else {
        this.cantidad -= cantidad;
    }
}

/**
 * Devuelve el estado del objeto
 *
 * @return
 */
```

```
    @Override
    public String toString() {
        return "El titular " + titular + " tiene " + cantidad + " euros en la cuenta";
    }

}
```

–Clase ejecutable

```
public class CuentaApp {

    public static void main(String[] args) {

        Cuenta cuenta_1 = new Cuenta("ClaradelRey");
        Cuenta cuenta_2 = new Cuenta("Fernando", 300);

        //Ingresa dinero en las cuentas
        cuenta_1.ingresar(300);
        cuenta_2.ingresar(400);

        //Retiramos dinero en las cuentas
        cuenta_1.retirar(500);
        cuenta_2.retirar(100);

        //Muestro la informacion de las cuentas
        System.out.println(cuenta_1); // 0 euros
        System.out.println(cuenta_2); // 600 euros

    }

}
```

2) Haz una clase llamada **Persona** que siga las siguientes condiciones:

- Sus atributos son: **nombre, edad, DNI, sexo** (H hombre, M mujer)**, peso y altura.** No queremos que se accedan directamente a ellos. Piensa que modificador de acceso es el más adecuado, también su tipo. Si quieres añadir algún atributo puedes hacerlo.
- Por defecto, todos los atributos menos el DNI serán valores por defecto según su tipo (0 números, cadena vacía para String, etc.). Sexo sera hombre por defecto, usa una constante para ello.
- Se implantaran varios constructores:
    - Un constructor por defecto.

    - Un constructor con el nombre, edad y sexo, el resto por defecto.
    - Un constructor con todos los atributos como parámetro.
- Los métodos que se implementaran son:
    - **calcularIMC()**: calculara si la persona esta en su peso ideal (peso en kg/(altura^2 en m)), si esta fórmula devuelve un valor menor que 20, la función devuelve un -1, si devuelve un número entre 20 y 25 (incluidos), significa que esta por debajo de su peso ideal la función devuelve un 0 y si devuelve un valor mayor que 25 significa que tiene sobrepeso, la función devuelve un 1. Te recomiendo que uses constantes para devolver estos valores.
        - **esMayorDeEdad()**: indica si es mayor de edad, devuelve un booleano.
        - **comprobarSexo(char sexo)**: comprueba que el sexo introducido es correcto. Si no es correcto, sera H. No sera visible al exterior.
        - **toString()**: devuelve toda la información del objeto.
        - **generaDNI()**: genera un número aleatorio de 8 cifras, genera a partir de este su número su letra correspondiente. Este método sera invocado cuando se construya el objeto. Puedes dividir el método para que te sea más fácil. No será visible al exterior.
        - Métodos set de cada parámetro, excepto de DNI.

Ahora, crea una clase ejecutable que haga lo siguiente:

- Pide por teclado el nombre, la edad, sexo, peso y altura.
- Crea 3 objetos de la clase anterior, el primer objeto obtendrá las anteriores variables pedidas por teclado, el segundo objeto obtendrá todos los anteriores menos el peso y la altura y el último por defecto, para este último utiliza los métodos set para darle a los atributos un valor.
- Para cada objeto, deberá comprobar si esta en su peso ideal, tiene sobrepeso o por debajo de su peso ideal con un mensaje.
- Indicar para cada objeto si es mayor de edad.
- Por último, mostrar la información de cada objeto.

Puedes usar métodos en la clase ejecutable, para que os sea mas fácil.

–Clase Persona

```
public class Persona {

   //Constantes
   /**
    * Sexo por defecto
    */
   private final static char SEXO_DEF = 'H';

   /**
```

```
 * El peso de la persona esta por debajo del peso ideal
 */
public static final int INFRAPESO = -1;

/**
 * El peso de la persona esta en su peso ideal
 */
public static final int PESO_IDEAL = 0;

/**
 * El peso de la persona esta por encima del peso ideal
 */
public static final int SOBREPESO = 1;

//Atributos
/**
 * Nombre de la persona
 */
private String nombre;

/**
 * Edad de la persona
 */
private int edad;

/**
 * DNI de la persona, se genera al construir el objeto
 */
private String DNI;

/**
 * Sexo de la persona, H hombre M mujer
 */
private char sexo;

/**
 * Peso de la persona
 */
private double peso;

/**
 * Altura de la persona
 */
private double altura;

//Contructores
```

```
/**
 * Constructor por defecto
 */
public Persona() {
    this("", 0, SEXO_DEF, 0, 0);
}

/**
 * Constructor con 3 parametroe
 *
 * @param nombre de la persona
 * @param edad de la persona
 * @param sexo de la persona
 */
public Persona(String nombre, int edad, char sexo) {
    this(nombre, edad, sexo, 0, 0);
}

/**
 * Constructor con 5 parametros
 *
 * @param nombre de la persona
 * @param edad de la persona
 * @param sexo de la persona
 * @param peso de la persona
 * @param altura de la persona
 */
public Persona(String nombre, int edad, char sexo, double peso, double altura) {
    this.nombre = nombre;
    this.edad = edad;
    this.peso = peso;
    this.altura = altura;
    generarDni();
    this.sexo = sexo;
    comprobarSexo();
}

//Métodos privados
private void comprobarSexo() {

    //Si el sexo no es una H o una M, por defecto es H
    if (sexo != 'H' && sexo != 'M') {
        this.sexo = SEXO_DEF;
    }
}
```

```
private void generarDni() {
    final int divisor = 23;

    //Generamos un número de 8 digitos
    int numDNI = ((int) Math.floor(Math.random() * (100000000 - 10000000) + 10000000));
    int res = numDNI - (numDNI / divisor * divisor);

    //Calculamos la letra del DNI
    char letraDNI = generaLetraDNI(res);

    //Pasamos el DNI a String
    DNI = Integer.toString(numDNI) + letraDNI;
}

private char generaLetraDNI(int res) {
    char letras[] = {'T', 'R', 'W', 'A', 'G', 'M', 'Y',
        'F', 'P', 'D', 'X', 'B', 'N', 'J', 'Z',
        'S', 'Q', 'V', 'H', 'L', 'C', 'K', 'E'};

    return letras[res];
}

//Métodos publicos
/**
 * Modifica el nombre de la persona
 *
 * @param nombre a cambiar
 */
public void setNombre(String nombre) {
    this.nombre = nombre;
}

/**
 * Modifica la edad de la persona
 *
 * @param edad a cambiar
 */
public void setEdad(int edad) {
    this.edad = edad;
}

/**
 * Modifica el sexo de la persona, comprueba que es correcto
 *
 * @param sexo a cambiar
 */
```

```
public void setSexo(char sexo) {
    this.sexo = sexo;
}

/**
 * Modifica el peso de la persona
 *
 * @param peso a cambiar
 */
public void setPeso(double peso) {
    this.peso = peso;
}

/**
 * Modifica la altura de la persona
 *
 * @param altura a cambiar
 */
public void setAltura(double altura) {
    this.altura = altura;
}

/**
 * Calcula el indice de masa corporal
 *
 * @return codigo numerico
 * <ul><li>-1: la persona esta por debajo de su peso ideal</li>
 * <li>0: la persona esta en su peso ideal</li>
 * <li>1: la persona esta por encima de su peso ideal</li></ul>
 */
public int calcularIMC() {
    //Calculamos el peso de la persona
    double pesoActual = peso / (Math.pow(altura, 2));
    //Segun el peso, devuelve un codigo
    if (pesoActual >= 20 && pesoActual <= 25) {
        return PESO_IDEAL;
    } else if (pesoActual < 20) {
        return INFRAPESO;
    } else {
        return SOBREPESO;
    }
}

/**
 * Indica si la persona es mayor de edad
 *
```

```
     * @return true si es mayor de edad y false es menor de edad
     */
    public boolean esMayorDeEdad() {
        boolean mayor = false;
        if (edad >= 18) {
            mayor = true;
        }
        return mayor;
    }

    /**
     * Devuelve informacion del objeto
     *
     * @return cadena con toda la informacion
     */
    @Override
    public String toString() {
        String sexo;
        if (this.sexo == 'H') {
            sexo = "hombre";
        } else {
            sexo = "mujer";
        }
        return "Informacion de la persona:\n"
                + "Nombre: " + nombre + "\n"
                + "Sexo: " + sexo + "\n"
                + "Edad: " + edad + " años\n"
                + "DNI: " + DNI + "\n"
                + "Peso: " + peso + " kg\n"
                + "Altura: " + altura + " metros\n";
    }

}
```

–Clase ejecutable con JOptionPane

```
import javax.swing.JOptionPane;

public class PersonaApp {

    public static void main(String[] args) {

        //Introducimos los datos
        String nombre = JOptionPane.showInputDialog("Introduce el nombre");

        String texto = JOptionPane.showInputDialog("Introduce la edad");
```

```
    int edad = Integer.parseInt(texto);

    texto = JOptionPane.showInputDialog("Introduce el sexo");
    char sexo = texto.charAt(0);

    texto = JOptionPane.showInputDialog("Introduce el peso");
    double peso = Double.parseDouble(texto);

    texto = JOptionPane.showInputDialog("Introduce la altura");
    double altura = Double.parseDouble(texto);

    //Creamos objetos con cada constructor
    Persona persona1 = new Persona();
    Persona persona2 = new Persona(nombre, edad, sexo);
    Persona persona3 = new Persona(nombre, edad, sexo, peso, altura);

    //Los datos que no esten completos los insertamos con los metodos set
    persona1.setNombre("Laura");
    persona1.setEdad(30);
    persona1.setSexo('M');
    persona1.setPeso(60);
    persona1.setAltura(1.60);

    persona2.setPeso(90.5);
    persona2.setAltura(1.80);

    //Usamos metodos para realizar la misma accion para cada objeto
    System.out.println("Persona1");
    MuestraMensajePeso(persona1);
    MuestraMayorDeEdad(persona1);
    System.out.println(persona1.toString());

    System.out.println("Persona2");
    MuestraMensajePeso(persona2);
    MuestraMayorDeEdad(persona2);
    System.out.println(persona2.toString());

    System.out.println("Persona3");
    MuestraMensajePeso(persona3);
    MuestraMayorDeEdad(persona3);
    System.out.println(persona3.toString());
}

public static void MuestraMensajePeso(Persona p) {
    int IMC = p.calcularIMC();
    switch (IMC) {
```

```
            case Persona.PESO_IDEAL:
                System.out.println("La persona esta en su peso ideal");
                break;
            case Persona.INFRAPESO:
                System.out.println("La persona esta por debajo de su peso ideal");
                break;
            case Persona.SOBREPESO:
                System.out.println("La persona esta por encima de su peso ideal");
                break;
        }
    }

    public static void MuestraMayorDeEdad(Persona p) {

        if (p.esMayorDeEdad()) {
            System.out.println("La persona es mayor de edad");
        } else {
            System.out.println("La persona no es mayor de edad");
        }
    }

}
```

3) Haz una clase llamada **Password** que siga las siguientes condiciones:

- Que tenga los atributos **longitud** y **contraseña** . Por defecto, la longitud sera de 8.
- Los constructores serán los siguiente:
    - Un constructor por defecto.
- Un constructor con la longitud que nosotros le pasemos. Generara una contraseña aleatoria con esa longitud.
- Los métodos que implementa serán:
    - **esFuerte()**: devuelve un booleano si es fuerte o no, para que sea fuerte debe tener mas de 2 mayúsculas, mas de 1 minúscula y mas de 5 números.
    - **generarPassword()**: genera la contraseña del objeto con la longitud que tenga.
    - Método get para contraseña y longitud.
    - Método set para longitud.

Ahora, crea una clase clase ejecutable:

- Crea un array de Passwords con el tamaño que tu le indiques por teclado.
- Crea un bucle que cree un objeto para cada posición del array.
- Indica también por teclado la longitud de los Passwords (antes de bucle).
- Crea otro array de booleanos donde se almacene si el password del array de Password es o no fuerte (usa el bucle anterior).

- Al final, muestra la contraseña y si es o no fuerte (usa el bucle anterior). Usa este simple formato:

contraseña1 valor_booleano1

contraseña2 valor_bololeano2

…

_____

–Clase Password

```
public class Password {

    //Constantes

    /**
     * Longitud por defecto
     */
    private final static int LONG_DEF=8;

    //Atributos

    /**
     * Longitud de la contraseña
     */
    private int longitud;
    /**
     * caracteres de la contraseña
     */
    private String contraseña;

    //Metodos publicos

    /**
     * Devuelve la longitud
     * @return longitud de la contraseña
     */
    public int getLongitud() {
        return longitud;
    }

    /**
     * Modifica la longitud de la contraseña
```

```
     * @param longitud a cambiar
     */
    public void setLongitud(int longitud) {
        this.longitud = longitud;
    }

    /**
     * Devuelve la contraseña
     * @return contraseña
     */
    public String getContraseña() {
        return contraseña;
    }

    /**
     * Genera una contraseña al azar con la longitud que este definida
     * @return contraseña
     */
    public String generaPassword (){
        String password="";
        for (int i=0;i<longitud;i++){
            //Generamos un numero aleatorio, segun este elige si añadir una minuscula,
mayuscula o numero
            int eleccion=((int)Math.floor(Math.random()*3+1));

            if (eleccion==1){
                char minusculas=(char)((int)Math.floor(Math.random()*(123-97)+97));
                password+=minusculas;
            }else{
                if(eleccion==2){
                    char mayusculas=(char)((int)Math.floor(Math.random()*(91-65)+65));
                    password+=mayusculas;
                }else{
                    char numeros=(char)((int)Math.floor(Math.random()*(58-48)+48));
                    password+=numeros;
                }
            }
        }
        return password;
    }

    /**
     * Comprueba la fortaleza de la contraseña
     * @return
     */
    public boolean esFuerte(){
```

```
        int cuentanumeros=0;
        int cuentaminusculas=0;
        int cuentamayusculas=0;
        //Vamos caracter a caracter y comprobamos que tipo de caracter es
        for (int i=0;i<contraseña.length();i++){
            if (contraseña.charAt(i)>=97 && contraseña.charAt(i)<=122){
                cuentaminusculas+=1;
            }else{
                if (contraseña.charAt(i)>=65 && contraseña.charAt(i)<=90){
                    cuentamayusculas+=1;
            }else{
                cuentanumeros+=1;
                }
            }
          }
          //Si la constraseña tiene mas de 5 numeros, mas de 1 minuscula y mas de 2
mayusculas
          if (cuentanumeros>=5 && cuentaminusculas>=1 && cuentamayusculas>=2){
          return true;
        }else{
          return false;
        }
    }

    //Constructores
    /**
     * Crea una contraseña al azar
     */
    public Password (){
        this(LONG_DEF);
    }

    /**
     * La contraseña sera la pasada por parametro
     * @param longitud
     */
    public Password (int longitud){
        this.longitud=longitud;
        contraseña=generaPassword();
    }
}
```

–Clase ejecutable

```
import javax.swing.JOptionPane;
```

```
public class PasswordApp {

    public static void main(String[] args) {

        //Introducimos el tamaño del array y la longitud del password
        String texto=JOptionPane.showInputDialog("Introduce un tamaño para el array");
        int tamanio=Integer.parseInt(texto);

        texto=JOptionPane.showInputDialog("Introduce la longitud del password");
        int longitud=Integer.parseInt(texto);

        //Creamos los arrays
        Password listaPassword[]=new Password[tamanio];
        boolean fortalezaPassword[]=new boolean[tamanio];

        //Creamos objetos, indicamos si es fuerte y mostramos la contraseña y su fortaleza.
        for(int i=0;i<listaPassword.length;i++){
            listaPassword[i]=new Password(longitud);
            fortalezaPassword[i]=listaPassword[i].esFuerte();
            System.out.println(listaPassword[i].getContraseña()+" "+fortalezaPassword[i]);
        }
    }

}
```

4) Crearemos una supeclase llamada **Electrodomestico** 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 € 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
Entre 0 y 19 kg	10 €
Entre 20 y 49 kg	50 €
Entre 50 y 79 kg	80 €
Mayor que 80 kg	100 €

Crearemos una subclase llamada **Lavadora** 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 Electrodomestico también deben afectar al precio.

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

- Sus atributos son **resolución** (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 €. Recuerda que las condiciones que hemos visto en la clase Electrodomestico también deben afectar al precio.

Ahora crea una clase ejecutable que realice lo siguiente:

- Crea un array de Electrodomesticos de 10 posiciones.
- Asigna a cada posición un objeto de las clases anteriores con los valores que desees.
- Ahora, recorre este array y ejecuta el método precioFinal().
- Deberás mostrar el precio de cada clase, es decir, el precio de todas las televisiones por un lado, el de las lavadoras por otro y la suma de los Electrodomesticos (puedes crear objetos Electrodomestico, pero recuerda que Television y Lavadora también son electrodomésticos). Recuerda el uso operador instanceof.

Por ejemplo, si tenemos un Electrodomestico con un precio final de 300, una lavadora de 200 y una televisión de 500, el resultado final sera de 1000 (300+200+500) para electrodomésticos, 200 para lavadora y 500 para televisión.

–Clase Electrodomestico

```
public class Electrodomestico {

    //Constantes

    /**
     * Color por defecto
     */
    protected final static String COLOR_DEF="blanco";

    /**
     * Consumo energetico por defecto
     */
    protected final static char CONSUMO_ENERGETICO_DEF='F';

    /**
     * Precio base por defecto
     */
    protected final static double PRECIO_BASE_DEF=100;

    /**
     * Peso por defecto
     */
    protected final static double PESO_DEF=5;

    //Atributos

    /**
```

```
 * El precio base del electrodomestico
 */
protected double precioBase;

/**
 * Color del electrodomestico
 */
protected String color;

/**
 * Indica el consumo energetico del electrodomestico
 */
protected char consumoEnergetico;

/**
 * Peso del electrodomestico
 */
protected double peso;

//Métodos privados

private void comprobarColor(String color){

    //Colores disponibles
    String colores[]={"blanco", "negro", "rojo", "azul", "gris"};
    boolean encontrado=false;

    for(int i=0;i<colores.length && !encontrado;i++){

        if(colores[i].equals(color)){
            encontrado=true;
        }

    }

    if(encontrado){
        this.color=color;
    }else{
        this.color=COLOR_DEF;
    }


}

/**
 * Comprueba el consumo energetico
```

```
     * Solo mayusculas, si es una 'a' no lo detecta como una 'A'
     * @param consumoEnergetico
     */
    public void comprobarConsumoEnergetico(char consumoEnergetico){

        if(consumoEnergetico>=65 && consumoEnergetico<=70){
            this.consumoEnergetico=consumoEnergetico;
        }else{
            this.consumoEnergetico=CONSUMO_ENERGETICO_DEF;
        }

    }

    //Métodos publicos
    /**
     * Devuelve el precio base del electrodomestico
     * @return precio base del electrodomestico
     */
    public double getPrecioBase() {
        return precioBase;
    }
    /**
     * Devuelve el color del electrodomestico
     * @return color del elesctrodomestico
     */
    public String getColor() {
        return color;
    }

     /**
     * Devuelve el consumo energetico del electrodomestico
     * @return consumo energetico del electrodomestico
     */
    public char getConsumoEnergetico() {
        return consumoEnergetico;
    }
    /**
     * Devuelve el peso del electrodomestico
     * @return peso del electrodomestico
     */
    public double getPeso() {
        return peso;
    }
    /**
     * Precio final del electrodomestico
     * @return precio final del electrodomestico
```

```
     */
    public double precioFinal(){
        double plus=0;
        switch(consumoEnergetico){
            case 'A':
                plus+=100;
                break;
            case 'B':
                plus+=80;
                break;
            case 'C':
                plus+=60;
                break;
            case 'D':
                plus+=50;
                break;
            case 'E':
                plus+=30;
                break;
            case 'F':
                plus+=10;
                break;
        }

        if(peso>=0 && peso<19){
            plus+=10;
        }else if(peso>=20 && peso<49){
            plus+=50;
        }else if(peso>=50 && peso<=79){
            plus+=80;
        }else if(peso>=80){
            plus+=100;
        }

        return precioBase+plus;
    }

    //Constructores

    /**
     * Contructor por defecto
     */
    public Electrodomestico(){
        this(PRECIO_BASE_DEF, PESO_DEF, CONSUMO_ENERGETICO_DEF,
COLOR_DEF);
    }
```

```
    /**
     * Contructor con 2 parametros
     * @param precioBase del electrodomestico
     * @param peso del electrodomestico
     */
    public Electrodomestico(double precioBase, double peso){
        this(precioBase, peso, CONSUMO_ENERGETICO_DEF, COLOR_DEF);
    }

    /**
     * Constructor con 4 parametros
     * @param precioBase
     * @param peso
     * @param consumoEnergetico
     * @param color
     */
    public Electrodomestico(double precioBase, double peso, char consumoEnergetico,
String color){
        this.precioBase=precioBase;
        this.peso=peso;
        comprobarConsumoEnergetico(consumoEnergetico);
        comprobarColor(color);
    }

}
```

–Clase Lavadora

```
public class Lavadora extends Electrodomestico{

    //Constantes

    /**
     * Carga por defecto
     */
    private final static int CARGA_DEF=5;

    //Atributos

    /**
     * Carga de la lavadora
     */
    private int carga;

    //Métodos publicos
```

```
    /**
     * Devuelve la carga de la lavadora
     * @return
     */
    public int getCarga() {
        return carga;
    }

    /**
     * Precio final de la lavadora
     * @return precio final de la lavadora
     */
    public double precioFinal(){
        //Invocamos el método precioFinal del método padre
        double plus=super.precioFinal();

        //añadimos el código necesario
        if (carga>30){
            plus+=50;
        }

        return plus;
    }

    //Constructor

    /**
     * Contructor por defecto
     */
    public Lavadora(){
        this(PRECIO_BASE_DEF, PESO_DEF, CONSUMO_ENERGETICO_DEF,
COLOR_DEF, CARGA_DEF);
    }

    /**
     * Constructor con 2 parametros
     * @param precioBase
     * @param peso
     */
    public Lavadora(double precioBase, double peso){
        this(precioBase, peso, CONSUMO_ENERGETICO_DEF, COLOR_DEF,
CARGA_DEF);
    }

    /**
```

```
     * Constructor con 5 parametros
     * @param precioBase
     * @param peso
     * @param consumoEnergetico
     * @param color
     * @param carga
     */
    public Lavadora(double precioBase, double peso, char consumoEnergetico, String color,
int carga){
        super(precioBase,peso, consumoEnergetico,color);
        this.carga=carga;
    }
}
```

–Clase Television

```
public class Television extends Electrodomestico{

    //Constantes

    /**
     * Resolucion por defecto
     */
    private final static int RESOLUCION_DEF=20;

    //Atributos

    /**
     * Resolucion del televisor
     */
    private int resolucion;

    /**
     * Indica si tiene o no sintonizadorTDT
     */
    private boolean sintonizadorTDT;

    //Métodos publicos

    /**
     * Precio final de la television
     * @return precio final de la television
     */
    public double precioFinal(){
        //Invocamos el método precioFinal del método padre
        double plus=super.precioFinal();
```

```
        //Añadimos el codigo necesario
        if (resolucion>40){
            plus+=precioBase*0.3;
        }
        if (sintonizadorTDT){
            plus+=50;
        }

        return plus;
    }

    //Constructor

    /**
     * Constructor por defecto
     */
    public Television(){
        this(PRECIO_BASE_DEF, PESO_DEF, CONSUMO_ENERGETICO_DEF,
COLOR_DEF, RESOLUCION_DEF, false);
    }

    /**
     * Constructor con 2 parametros
     * @param precioBase
     * @param peso
     */
    public Television(double precioBase, double peso){
        this(precioBase, peso, CONSUMO_ENERGETICO_DEF, COLOR_DEF,
RESOLUCION_DEF, false);
    }

    /**
     * Contructor con 6 parametros
     * @param precioBase
     * @param peso
     * @param consumoEnergetico
     * @param color
     * @param resolucion
     * @param sintonizadorTDT
     */
    public Television(double precioBase, double peso, char consumoEnergetico, String color,
int resolucion, boolean sintonizadorTDT){
        super(precioBase, peso, consumoEnergetico, color);
        this.resolucion=resolucion;
        this.sintonizadorTDT=sintonizadorTDT;
```

```
    }
}
```

–Clase Ejecutable

```
public class ElectrodomesticosApp {

    public static void main(String[] args) {

        //Creamos un array de Electrodomesticos
        Electrodomestico listaElectrodomesticos[]=new Electrodomestico[10];

        //Asignamos cada una de las posiciones como queramos
        listaElectrodomesticos[0]=new Electrodomestico(200, 60, 'C', "Verde");
        listaElectrodomesticos[1]=new Lavadora(150, 30);
        listaElectrodomesticos[2]=new Television(500, 80, 'E', "negro", 42, false);
        listaElectrodomesticos[3]=new Electrodomestico();
        listaElectrodomesticos[4]=new Electrodomestico(600, 20, 'D', "gris");
        listaElectrodomesticos[5]=new Lavadora(300, 40, 'Z', "blanco", 40);
        listaElectrodomesticos[6]=new Television(250, 70);
        listaElectrodomesticos[7]=new Lavadora(400, 100, 'A', "verde", 15);
        listaElectrodomesticos[8]=new Television(200, 60, 'C', "naranja", 30, true);
        listaElectrodomesticos[9]=new Electrodomestico(50, 10);

        //Creamos las variables que usaremos para almacenar la suma de los precios
        double sumaElectrodomesticos=0;
        double sumaTelevisiones=0;
        double sumaLavadoras=0;

        //Recorremos el array invocando el metodo precioFinal
        for(int i=0;i<listaElectrodomesticos.length;i++){
            /*
             * Cuando una Television o una Lavadora este en la posicion del array actual,
             * pasara por su clase y por la de electrodomestico, ya que una television es un
electrodomestico.
             * Ejecutamos en cada uno su propia version del metodo precioFinal
             */

            if(listaElectrodomesticos[i] instanceof Electrodomestico){
                sumaElectrodomesticos+=listaElectrodomesticos[i].precioFinal();
            }
            if(listaElectrodomesticos[i] instanceof Lavadora){
                sumaLavadoras+=listaElectrodomesticos[i].precioFinal();
            }
            if(listaElectrodomesticos[i] instanceof Television){
```

```
                sumaTelevisiones+=listaElectrodomesticos[i].precioFinal();
            }
        }

        //Mostramos los resultados
        System.out.println("La suma del precio de los electrodomesticos es de
"+sumaElectrodomesticos);
        System.out.println("La suma del precio de las lavadoras es de "+sumaLavadoras);
        System.out.println("La suma del precio de las televisiones es de "+sumaTelevisiones);

    }

}
```