EEM 480 HW 2

In this homework you have to implement a simple database using **link list** structure where this list will be used in an inventory program of EE Department.

Your job is implementing the interface in `Inventory` class which is given below: (`Inventory` class will implement `InventoryInterface` interface.

```java
public class Device {
    private String Name;
    private String Type;
    private String Value;
    private Integer Count;


    public Device(String n, String t, String v, Integer c){
        Name = n;
        Type = t;
        Value = v;
        Count = c;
    }
}

public interface InventoryInterface {

    public void addResistor(String val, Integer cnt );
    public void addCapacitor(String val, String typ, Integer cnt );
    public void addInductor(String val, Integer cnt );
    public void addTransistor(String typ, Integer cnt );
    public int deleteResistor(String val, Integer cnt ); //returns amount of
                //resistors in the inventory otherwise returns -1
    public int deleteCapacitor(String val, String typ, Integer cnt);
    //returns amount of capacitors in the inventory otherwise returns -1
    public int deleteInductor(String val, Integer cnt ); //returns amount of
                //inductors in the inventory otherwise returns -1
    public int deleteTransistor(String typ, Integer cnt ); //returns amount
                // of Transistors in the inventory otherwise returns -1
    public void printInventory(); // Prints the every device with their
                // types /values/count in group of chunks
}
```

The requirements are given as:

`public void addResistor(String val, Integer cnt );` : The `cnt` amount of resistor with `val` has to be added to the Inventory. If there is the same type of the resistor before, its count has to be increased by `cnt` amount.

`public void addCapacitor(String val, Integer cnt );` : The `cnt` amount of capacitors with `val` has to be added to the Inventory. If there is the same type of the capacitor before, its count has to be increased by `cnt` amount.

`public void addInductor(String type, Integer cnt );` : The `cnt` amount of transistors with `type` has to be added to the Inventory. If there is the same type of the transistor before, its count has to be increased by `cnt` amount.

`public void addTransistor(String val, Integer cnt );` : The `cnt` amount of resistor with `val` has to be added to the Inventory. If there is the same type of the resistor before, its count has to be increased by `cnt` amount.

`public int deleteResistor(String val, Integer cnt );` : The `val` type resistors has to be searched. If found and the amount of resistors in the inventory is greater than `cnt`, the number of resistors has to be decreased by `cnt`. The amount of resistors of value `val` has to be returned. Otherwise -1 has to be returned.

`public int deleteCapacitors(String val, Integer cnt );` : The `val` type capacitors has to be searched. If found and the amount of capacitors in the inventory is greater than `cnt`, the number of capacitors has to be decreased by `cnt`. The amount of capacitors of value `val` has to be returned. Otherwise -1 has to be returned.

`public int deleteInductor(String val, Integer cnt );` : The `val` type inductors has to be searched. If found and the amount of inductors in the inventory is greater than `cnt`, the number of inductors has to be decreased by `cnt`. The amount of inductors of value `val` has to be returned. Otherwise -1 has to be returned.

`public int deleteTransistor(String typ, Integer cnt );` : The `typ` type transistors has to be searched. If found and the amount of transistors in the inventory is greater than `cnt`, the number of transistors has to be decreased by `cnt`. The amount of transistors of type typ has to be returned. Otherwise -1 has to be returned.

`public void printInventory(String val, Integer cnt );` : The devices in the Inventory has to be printed in an ordered fashion. First Resistors then Capacitors then Inductors and finally Transistors in **O(n) complexity**.

An Example which use the DB class:

```
public static void main(String[] args) {
        // TODO code application logic here
        Inventory myInventory = new Inventory();
        myInventory.addResistor("2K", 20);
        myInventory.addTransistor("BC128", 45);
        Integer remain;
        remain = myInventory.deleteResistor("2.5K", 23);
            if (remain >= 0){
                System.out.println("There are " + remain + " amount of 2.5 K
resistors in Inventory");
        }
    }
```

Rules for HW Submission

- You have to write your HW in NetBeans environment.
- You have to write a report with name "Report_HW2.pdf" explaining your HW (purpose, how did you solve it, what complexity you have, etc.) and what environment you used (NetBeans, for example). The person who read your report can easily use the class you have written.
- Discuss the result you have obtained.
- Submission should be in the form of a zip. When extracted, the result should be a single folder with the name "HW1".
- Don't forget to put your report into the zip file.
- The name of your project will be "**Name_Surame_HW2**. e.g. *Lutfullah_Arici_HW2*. If you do not obey the rule I will not grade your homework.
- You have to bundle your whole project folder into your HW2.zip file.
- If I extract your project file, then import to my environment and if it doesn't work, you will be graded on 30 not 100. (Double check. It saves life)
- Do HW by yourself. Be honest.