M1 COLLECTION MAGICTOUCH

**Problem Statement**

You need to develop a system to manage and query air conditioning (AC) maintenance records. Each record includes a service code and an AC type. Your system will be implemented in two classes: MaintenanceInfo and UserInterface.

**Class: MaintenanceInfo**

1. **Instance Variable:**
   * maintenanceMap: A Map<String, String> where the key is the service code (a unique identifier) and the value is the type of AC serviced.
2. **Methods to Implement:**
   * **addAcMaintenanceDetails(String serviceCode, String acType)**
     + **Purpose:** Add a new maintenance record to the maintenanceMap.
     + **Input:**
       - serviceCode (a string representing the unique identifier for the service).
       - acType (a string representing the type of AC serviced).
     + **Operation:** Insert the provided serviceCode and acType into maintenanceMap.
   * **findTheNumberOfRecordsBasedOnTheACType(String acType)**
     + **Purpose:** Determine the number of records that match a given AC type.
     + **Input:** A string acType representing the type of AC to search for.
     + **Operation:** Count the number of entries in maintenanceMap where the value matches the given acType. Return the count if records are found; return -1 if no matching records are found.
   * **findServiceCodesBasedOnACType(String acType)**
     + **Purpose:** Retrieve all service codes for a specific AC type.
     + **Input:** A string acType representing the type of AC to search for.
     + **Operation:** Collect all service codes from maintenanceMap that match the specified acType into a set and return it. The service codes should be in a set to ensure uniqueness.

**Class: UserInterface**

1. **Main Method:**
   * **Purpose:** Handle user interaction to add maintenance records, search for records, and display results.
   * **Operation:**
     + Prompt the user to input the number of records they wish to add.
     + Collect maintenance records from the user and add them using MaintenanceInfo.
     + Prompt the user to search for the number of records for a specific AC type and display the result.
     + Prompt the user to retrieve and display all service codes for a specific AC type.
2. **Details to Consider:**
   * Ensure the input is properly handled and validated.
   * Use appropriate methods from MaintenanceInfo to perform operations and display results to the user.

**Sample Input and Output**

**Test Case 1**

**Input:**

Enter number of records to be added

3

Enter the records (Service code: AC type)

A001:Split

A002:Window

A003:Split

Enter the AC type to be searched

Split

The records based on Split are 2

Enter the AC Type to identify the Service codes

Window

Records based on the Window are

A002

**Output:**

Enter number of records to be added

3

Enter the records (Service code: AC type)

A001:Split

A002:Window

A003:Split

Enter the AC type to be searched

Split

The records based on Split are 2

Enter the AC Type to identify the Service codes

Window

Records based on the Window are

A002

**Test Case 2**

**Input:**

Enter number of records to be added

2

Enter the records (Service code: AC type)

B101:Central

B102:Central

Enter the AC type to be searched

Central

The records based on Central are 2

Enter the AC Type to identify the Service codes

Split

No service codes were found for the type Split

**Output:**

Enter number of records to be added

2

Enter the records (Service code: AC type)

B101:Central

B102:Central

Enter the AC type to be searched

Central

The records based on Central are 2

Enter the AC Type to identify the Service codes

Split

No service codes were found for the type Split

**Test Case 3**

**Input:**

Enter number of records to be added

1

Enter the records (Service code: AC type)

C203:Window

Enter the AC type to be searched

Window

The records based on Window are 1

Enter the AC Type to identify the Service codes

Window

Records based on the Window are

C203

**Output:**

Enter number of records to be added

1

Enter the records (Service code: AC type)

C203:Window

Enter the AC type to be searched

Window

The records based on Window are 1

Enter the AC Type to identify the Service codes

Window

Records based on the Window are

C203

**Test Case 4**

**Input:**

Enter number of records to be added

0

Enter the AC type to be searched

Split

No records were found for Split

Enter the AC Type to identify the Service codes

Central

No service codes were found for the type Central

**Output:**

Enter number of records to be added

0

Enter the AC type to be searched

Split

No records were found for Split

Enter the AC Type to identify the Service codes

Central

No service codes were found for the type Central

Boiler Plate:  
  
import java.util.Map;  
import java.util.Scanner;  
import java.util.Set;  
import java.util.TreeMap;  
  
public class UserInterface {  
 public static void main(String[] args) {  
 MaintenanceInfo info = new MaintenanceInfo();  
 Scanner sc=new Scanner(System.in);  
 System.out.println("Enter number of records to be added");  
 int n=sc.nextInt();  
 sc.nextLine();  
 System.out.println("Enter the records (Service code: AC type)");  
 String [] recordDetails = new String[n];  
 Map<String, String> map = new TreeMap<String, String>();  
 for(int i=0;i<n;i++) {  
 recordDetails[i] = sc.nextLine();  
 String[] strArr = recordDetails[i].split(":");  
 map.put(strArr[0], strArr[1]);  
 }  
 info.setMaintenanceMap(map);  
   
 \\ Fill the code   
  
 }  
}

import java.util.HashSet;  
import java.util.Map;  
import java.util.Set;  
import java.util.TreeMap;  
  
public class MaintenanceInfo {  
  
 private Map<String,String> maintenanceMap=new TreeMap<String,String>();  
   
 public Map<String, String> getMaintenanceMap() {  
 return maintenanceMap;  
 }  
  
 public void setMaintenanceMap(Map<String, String> maintenanceMap) {  
 this.maintenanceMap = maintenanceMap;  
 }  
   
 public void addAcMaintenanceDetails(String serviceCode, String acType) {  
 \\ Fill the code  
   
 }  
  
 public int findTheNumberOfRecordsBasedOnTheACType(String acType) {  
\\ Fill the code  
 return 0;  
   
 }  
  
 public Set<String> findServiceCodesBasedOnACType(String acType){

\\ Fill the code

return null;  
  
  
 }  
}

### M1 COLLECTION BookHaeven **Problem Statement**

You are tasked with developing a system to manage and query client information. Each client record consists of a client ID and a package name. Your solution will involve two classes: ClientInfo and UserInterface.

**Class: ClientInfo**

1. **Instance Variable:**
   * clientSet: A Set<String> to store client records. Each record is represented as a string in the format "ClientID:PackageName".
2. **Methods to Implement:**
   * **addClient(String clientDetails)**
     + **Purpose:** Add a new client record to the clientSet.
     + **Input:** A string clientDetails in the format "ClientID:PackageName".
     + **Operation:** Insert the provided clientDetails string into clientSet.
   * **getTotalClientsByPackageName(String packageName)**
     + **Purpose:** Determine the total number of clients who have a specific package.
     + **Input:** A string packageName representing the package to search for.
     + **Operation:** Count how many entries in clientSet have the specified packageName. Return the count if clients are found; return -1 if no matching clients are found.
   * **listClientIdsByPackageName(String packageName)**
     + **Purpose:** Retrieve all client IDs associated with a specific package.
     + **Input:** A string packageName representing the package to search for.
     + **Operation:** Collect all client IDs from clientSet that match the specified packageName into a list and return it. The client IDs should be in a list to maintain the order of insertion.

**Class: UserInterface**

1. **Main Method:**
   * **Purpose:** Handle user interaction to add client records, search for clients by package, and display results.
   * **Operation:**
     + Prompt the user to input the number of clients they wish to add.
     + Collect client records from the user and add them using ClientInfo.
     + Prompt the user to search for the total number of clients with a specific package name and display the result.
     + Prompt the user to retrieve and display all client IDs for a specific package name.
2. **Details to Consider:**
   * Ensure the input is properly handled and validated.
   * Use appropriate methods from ClientInfo to perform operations and display results to the user.

**Sample Input and Output**

**Test Case 1**

**Input:**

Enter number of clients to be added

3

Enter the client details

C001:Gold

C002:Silver

C003:Gold

Enter the package name to find the total count of clients

Gold

Total number of clients with package name Gold is 2

Enter the package name to search for client Ids

Silver

Client Ids based on package name Silver

C002

**Output:**

Enter number of clients to be added

3

Enter the client details

C001:Gold

C002:Silver

C003:Gold

Enter the package name to find the total count of clients

Gold

Total number of clients with package name Gold is 2

Enter the package name to search for client Ids

Silver

Client Ids based on package name Silver

C002

**Test Case 2**

**Input:**

Enter number of clients to be added

2

Enter the client details

C101:Platinum

C102:Platinum

Enter the package name to find the total count of clients

Platinum

Total number of clients with package name Platinum is 2

Enter the package name to search for client Ids

Gold

No Clients found with package name Gold

**Output:**

vbnet

Copy code

Enter number of clients to be added

2

Enter the client details

C101:Platinum

C102:Platinum

Enter the package name to find the total count of clients

Platinum

Total number of clients with package name Platinum is 2

Enter the package name to search for client Ids

Gold

No Clients found with package name Gold

**Test Case 3**

**Input:**

Enter number of clients to be added

1

Enter the client details

C203:Basic

Enter the package name to find the total count of clients

Basic

Total number of clients with package name Basic is 1

Enter the package name to search for client Ids

Basic

Client Ids based on package name Basic

C203

**Output:**

Enter number of clients to be added

1

Enter the client details

C203:Basic

Enter the package name to find the total count of clients

Basic

Total number of clients with package name Basic is 1

Enter the package name to search for client Ids

Basic

Client Ids based on package name Basic

C203

**Test Case 4**

**Input:**

Enter number of clients to be added

0

Enter the package name to find the total count of clients

Premium

No clients found with package name Premium

Enter the package name to search for client Ids

Standard

No Clients found with package name Standard

**Output:**

Enter number of clients to be added

0

Enter the package name to find the total count of clients

Premium

No clients found with package name Premium

Enter the package name to search for client Ids

Standard

No Clients found with package name Standard

Boiler plate:   
import java.util.ArrayList;  
import java.util.HashSet;  
import java.util.List;  
import java.util.Set;  
  
  
public class ClientInfo {  
 private Set<String> clientSet = new HashSet<String>();  
  
 public Set<String> getClientSet() {  
 return clientSet;  
 }  
  
 public void setClientSet(Set<String> clientSet) {  
 this.clientSet = clientSet;  
 }  
  
 public void addClient(String clientDetails){  
 //Fill the code here  
   
 }  
  
 public int getTotalClientsByPackageName(String packageName) {  
 //Fill the code here  
  
 return 0;  
 }  
  
 public List<String> listClientIdsByPackageName(String packageName) {  
 //Fill the code here  
  
   
 return null;  
 }  
}

import java.util.List;  
import java.util.Scanner;  
  
public class UserInterface{  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.in);  
   
 //Fill the code here   
 System.out.println("Enter number of clients to be added");  
 int n=sc.nextInt();  
 sc.nextLine();  
 ClientInfo cf=new ClientInfo();  
 System.out.println("Enter the client details");  
 for(int i=0;i<n;i++)  
 {  
 String details=sc.nextLine();  
 cf.addClient(details);  
   
 }

\\ Fill the code here

}  
}

Question 3.

You are tasked with creating a simple system to manage pizza orders. Each order consists of an order ID and a pizza type, and your program should be able to add, search, and retrieve order details based on the pizza type. Your solution will involve two classes: PizzaOrderInfo and UserInterface.

**Class: PizzaOrderInfo**

1. **Instance Variable:**
   * orderSet: A Set<String> to store pizza orders. Each order is represented as a string in the format "OrderId"  
     ".
2. **Methods to Implement:**
   * **addPizzaOrderDetails(String orderDetail)**
     + **Purpose:** Add a new pizza order to the orderSet.
     + **Input:** A string orderDetail in the format "OrderId"  
       ".
     + **Operation:** Add the provided orderDetail string to orderSet.
   * **findTheNumberOfOrdersBasedOnThePizzaType(String pizzaType)**
     + **Purpose:** Determine the number of orders that match a given pizza type.
     + **Input:** A string pizzaType representing the type of pizza to search for.
     + **Operation:** Count how many orders in orderSet contain the specified pizza type. Return the count if orders are found; return -1 if no matching orders are found.
   * **findOrderDetailsBasedOnPizzaType(String pizzaType)**
     + **Purpose:** Retrieve all order details for a specific pizza type.
     + **Input:** A string pizzaType representing the type of pizza to search for.
     + **Operation:** Collect all order details from orderSet that match the specified pizza type into a list and return it. The order details should be formatted as "OrderId
     + ".

Test case 1:

Enter number of orders to be added

3

Enter the orders (Order Id: Pizza Type)

101:Margherita

102:Pepperoni

103:Margherita

Enter the Pizza type to be searched

Margherita

The orders based on Margherita are 2

Enter the Pizza type to identify the Order details

Pepperoni

Orders based on the Pepperoni are

Order Id: 102, Pizza Type: Pepperoni

TEST CASE 2:

Enter number of orders to be added

4

Enter the orders (Order Id: Pizza Type)

201:Vegetarian

202:Margherita

203:Vegetarian

204:BBQ Chicken

Enter the Pizza type to be searched

Vegetarian

The orders based on Vegetarian are 2

Enter the Pizza type to identify the Order details

Margherita

Orders based on the Margherita are

Order Id: 202, Pizza Type: Margherita

TEST CASE 3:

Enter number of orders to be added

2

Enter the orders (Order Id: Pizza Type)

301:Hawaiian

302:Supreme

Enter the Pizza type to be searched

Pepperoni

No orders were found for Pepperoni

Enter the Pizza type to identify the Order details

Hawaiian

Orders based on the Hawaiian are

Order Id: 301, Pizza Type: Hawaiian

TEST CASE 4:

Enter number of orders to be added

0

Enter the Pizza type to be searched

Margherita

No orders were found for Margherita

Enter the Pizza type to identify the Order details

Margherita

No Order details were found for the pizza - Margherita

**Boiler plate :**

import java.util.ArrayList;  
import java.util.List;  
import java.util.Set;  
import java.util.TreeSet;  
  
public class PizzaOrderInfo {  
   
 private Set<String> orderSet=new TreeSet<String>();  
   
 public Set<String> getOrderSet() {  
 return orderSet;  
 }  
  
 public void setOrderSet(Set<String> orderSet) {  
 this.orderSet = orderSet;  
 }  
   
 public void addPizzaOrderDetails(String orderDetail) {  
 //Fill the code here  
   
 }  
  
 public int findTheNumberOfOrdersBasedOnThePizzaType(String pizzaType) {  
   
   
 return 0;   
 }  
  
 public List<String> findOrderDetailsBasedOnPizzaType(String pizzaType){  
 //Fill the code here  
  
 return null;  
 }  
}

import java.util.List;  
import java.util.Scanner;  
  
public class UserInterface {  
 public static void main(String[] args) {  
 Scanner sc=new Scanner(System.in);  
 System.out.println("Enter number of orders to be added");  
 int n=sc.nextInt();  
 sc.nextLine();  
   
 System.out.println("Enter the orders (Order Id: Pizza Type)");  
   
 PizzaOrderInfo p = new PizzaOrderInfo();  
 for(int i=0;i<n;i++) {  
 String s = sc.nextLine();  
 p.addPizzaOrderDetails(s);  
 }  
   
 //Fill the code here  
   
  
  
  
 }  
  
}