Graph Theory - S007 (E035)

Solved Challenges **0/1**

Back To Challenges List

Intelligent Chef

ID:11131 **Solved By 468 Users**

There are **N** persons in a hotel. Each person has their own preferences for food. The hotel chef wants to prepare the food items as minimum as possible.The program must accept the food preferences of each person as the input. The program must print the minimum number of food items that must be prepared to serve everyone in the hotel.
Note: Each person eats only one food item but has many options.

Boundary Condition(s):
1 <= N <= 10^4
1 <= Length of the name of each food item <= 50

Input Format:
The first line contains N.
The next N lines, each containing the string value(s) representing the preferences of food items of a person.

Output Format:
The first line contains the minimum number of food items that must be prepared to serve everyone in the hotel.

Example Input/Output 1:
Input:
5
Dosa Poori
Idli Poori
Idli Poori
Idli Dosa
Poori

Output:
2

Explanation:
Here N = 5 representing the 5 persons in the hotel.
At least **2** food items (**Idli** & **Poori**) must be prepared to serve everyone in the hotel.

Example Input/Output 2:
Input:
10
Chapati Idli Pongal Poori Dosa
Poori Dosa Chapati
Idli Dosa Poori
Pongal

Dosa Pongal Poori Chapati
Idli Pongal Poori
Idli Pongal Chapati Dosa
Dosa Chapati
Chapati Idli Pongal Poori Dosa
Chapati

Output:
3

Max Execution Time Limit: 2000 millisecs

Ambiance

Java (12.0)

Reset

```
import java.util.*;

class FoodItem implements Comparable<FoodItem>
{
    String name;
    List<Integer> customers;

    @Override
    public int compareTo(FoodItem other)
    {
        return this.customers.size()-other.customers.size();
    }


}
public class Hello {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int N = sc.nextInt();
        sc.nextLine();

        Map<String,FoodItem> foodItemMap = new HashMap<>();
        List<Integer> remainingCustomers = new ArrayList<>();
        for(int customer = 1;customer<=N;customer++)
        {
            remainingCustomers.add(customer);
            String preferences[]=sc.nextLine().split("\\s+");
            for(String item:preferences)
            {
                if(!foodItemMap.containsKey(item))
                {
                    FoodItem fi = new FoodItem();
                    fi.name=item;
```

```
35                      fi.customers=new ArrayList<>();
36                      foodItemMap.put(item,fi);
37                  }
38                  foodItemMap.get(item).customers.add(customer);
39              }
40          }
41          int count=0;
42          while(!remainingCustomers.isEmpty())
43          {
44              List<FoodItem> items = new ArrayList<>(foodItemMap
45              values());
46              Collections.sort(items,Collections.reverseOrder())
47              FoodItem mostPreffered = items.get(0);
48              count++;
49              foodItemMap.remove(mostPreffered.name);
50              remainingCustomers.removeAll(mostPreffered.
51              customers);
52              for(String foodItem:foodItemMap.keySet())
53              {
54                  foodItemMap.get(foodItem).customers.
55                  removeAll(mostPreffered.customers);
56              }
57          }
58          System.out.println(count);
59
60      }
```

Code did not pass the execution

TestCase ID: 63993

Input:

5
Dosa Poori
Idli Poori
Idli Poori
Idli Dosa
Poori

Expected Output:

2

Your Program Output:

```
3
```

Save Run

Run with a custom test case (Input/Output)