# Exercises: Sets and Dictionaries Advanced

Problems for exercises and homework for the ["C# Advanced" course @ SoftUni](https://softuni.bg/courses/csharp-advanced).

You can check your solutions here: <https://judge.softuni.bg/Contests/1182/>

## Unique Usernames

On the first line you will be given an integer **N**. On the next **N** lines you will receive one username per line. Write a simple program that reads from the console a sequence of **N** usernames and keeps a collection only of the unique ones. Print the collection on the console in order of insertion:

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 6  Ivan  Ivan  Ivan  SemoMastikata  Ivan  Hubaviq1234 | Ivan  SemoMastikata  Hubaviq1234 |

## Sets of Elements

On the first line you are given the length of two sets **n** and **m**. On the next **n** + **m** lines there are **n** numbers that are in the first set and **m** numbers that are in the second one. Find all non-repeating elements that appear in both of them and print those from the **n** set.

Set with length n = 4: {1, **3**, **5**, 7}

Set with length m = 3: {**3**, 4, **5**}

Set that contains all repeating elements -> {**3**, **5**}

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 4 3  1  3  5  7  3  4  5 | 3 5 |
| 2 2  1  3  1  5 | 1 |

## Periodic Table

You are given an **n** number of chemical compounds joined with space (' '). You need to keep track of all chemical elements used in the compounds and at the end print all unique ones in ascending order:

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 4  Ce O  Mo O Ce  Ee  Mo | Ce Ee Mo O |
| 3  Ge Ch O Ne  Nb Mo Tc  O Ne | Ch Ge Mo Nb Ne O Tc |

## Even Times

You are given a list of **N** integer numbers all but one of which appears an odd number of times. Write a program to find the one integer which appears an **even number of times**.

### Examples

|  |  |  |  |
| --- | --- | --- | --- |
| **Input** | **Output** | **Input** | **Output** |
| 3  2  -1  2 | 2 | 5  1  2  3  1  5 | 1 |

## Count Symbols

Write a program that reads some text from the console and counts the occurrences of each character in it. Print the results in **alphabetical** (lexicographical) order.

### Examples

|  |  |  |  |
| --- | --- | --- | --- |
| **Input** | **Output** | **Input** | **Output** |
| SoftUni rocks | : 1 time/s  S: 1 time/s  U: 1 time/s  c: 1 time/s  f: 1 time/s  i: 1 time/s  k: 1 time/s  n: 1 time/s  o: 2 time/s  r: 1 time/s  s: 1 time/s  t: 1 time/s | Did you know Math.Round rounds to the nearest even integer? | : 9 time/s  .: 1 time/s  ?: 1 time/s  D: 1 time/s  M: 1 time/s  R: 1 time/s  a: 2 time/s  d: 3 time/s  e: 7 time/s  g: 1 time/s  h: 2 time/s  i: 2 time/s  k: 1 time/s  n: 6 time/s  o: 5 time/s  r: 3 time/s  s: 2 time/s  t: 5 time/s  u: 3 time/s  v: 1 time/s  w: 1 time/s  y: 1 time/s |

## Wardrobe

You just bought a new wardrobe. The weird thing about it is that it came prepackaged with a big box of clothes (just like those refrigerators, which ship with free beer inside them)! So, you’ll need to find something to wear.

### Input

On the first line of the input, you will receive n – the **number of lines** of clothes, which came prepackaged for the wardrobe.

On the next n lines, you will receive the clothes for each color in the format:

* “{color} -> {item1},{item2},{item3}…”

If a color is added a **second** time, **add** **all items** from it and **count** the **duplicates**.

**Finally**, you will receive the **color** and **item** of the clothing, that you need to look for.

### Output

Go through all the **colors** of the clothes and print them in the following format:

|  |
| --- |
| **{color}** clothes:  \* **{item1}** - **{count}**  \* **{item2}** - **{count}**  \* **{item3}** - **{count}**  …  \* **{itemN}** - **{count}** |

If the **color** lines up with the **clothing item**, print “(found!)” alongside the item. See the examples to better understand the output.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 4  Blue -> dress,jeans,hat  Gold -> dress,t-shirt,boxers  White -> briefs,tanktop  Blue -> gloves  Blue dress | Blue clothes:  \* dress - 1 (found!)  \* jeans - 1  \* hat - 1  \* gloves - 1  Gold clothes:  \* dress - 1  \* t-shirt - 1  \* boxers - 1  White clothes:  \* briefs - 1  \* tanktop - 1 |

|  |  |
| --- | --- |
| **Input** | **Output** |
| 4  Red -> hat  Red -> dress,t-shirt,boxers  White -> briefs,tanktop  Blue -> gloves  White tanktop | Red clothes:  \* hat - 1  \* dress - 1  \* t-shirt - 1  \* boxers - 1  White clothes:  \* briefs - 1  \* tanktop - 1 (found!)  Blue clothes:  \* gloves - 1 |

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5  Blue -> shoes  Blue -> shoes,shoes,shoes  Blue -> shoes,shoes  Blue -> shoes  Blue -> shoes,shoes  Red tanktop | Blue clothes:  \* shoes - 9 |

## \*The V-Logger

As you know vlogging (making short videos instead of written blogposts) is the new super trend among young people. You have always been very passionate about vloggers and their videos and now you decided to create a website, called "The V-Logger", so to rank the most followed vloggers. Good thing that you are also a programmer, so you can easily track what vloggers do.

You need to implement functionality that allows **vloggers to register** in your website. Note that, every vlogger must have a **unique vloggername**, so to be recognizable by his followers.

**Vloggers** also like to **follow other vloggers**, so that they can see immediately when a new video is posted.

A vlogger **can follow** **as many other vloggers** **as he wants**, but he **cannot** follow **himself** or follow someone he is **already a follower** of.

### Input

The **input** will come as e sequence of strings, where each string will represent a **valid** command. The commands will be presented in the following format:

* "{vloggername}" **joined The V-Logger** – a vlogger got registered into your website.
  + Vloggernames **consist** **of only one word**.
  + If the **given** vloggername **is already taken ignore that command.**
* "{vloggername} followed {vloggername}" – The first vlogger followed the second vlogger.
  + If **any** of the **given vlogernames** **does not exist** in the log **ignore that command.**
* **"Statistics" -** Upon receiving this command you have to print a statistic about the registered vloggers in **"The** **V-Logger".**

### Output

* On the first line you have to print **the total amount of vloggers** using “The V-Logger” in format:

**"The V-Logger has a total of {registered vloggers} vloggers in its logs."**

* Then you have to find most famous vlogger (thevlogger with the most followers). He should be printed along with **his followers**, in the following format:

"**1.** {vlogger} : {followers} followers, {vloggers he is a follower of} following

\* {follower1}

\* {follower2} ... "

* + If more than one vloggers have the same number of followers, you find **the one following less people**
  + The **followers** should be printed in **lexicographical order**.
  + If the vlogger has **no followers**, print only the **first line.**
* Last print **all other vloggers**, ordered by **number of followers (descending)** and **number of followings (ascending)** in the following format:

"**{No}.** {vlogger} : {followers} followers, {vloggers he is a follower of} following**"**

### Constraints

* There will be no invalid input.
* There will be no situation where two vloggers have equal amount of followers and equal amount of followings
* The subscribers will be strings.
* Allowed time/memory: **100ms/16MB**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| EmilConrad joined The V-Logger  VenomTheDoctor joined The V-Logger  Saffrona joined The V-Logger  Saffrona followed EmilConrad  Saffrona followed VenomTheDoctor  EmilConrad followed VenomTheDoctor  VenomTheDoctor followed VenomTheDoctor  Saffrona followed EmilConrad  Statistics | The V-Logger has a total of 3 vloggers in its logs.  1. VenomTheDoctor : 2 followers, 0 following  \* EmilConrad  \* Saffrona  2. EmilConrad : 1 followers, 1 following  3. Saffrona : 0 followers, 2 following |
| JennaMarbles joined The V-Logger  JennaMarbles followed Zoella  AmazingPhil joined The V-Logger  JennaMarbles followed AmazingPhil  Zoella joined The V-Logger  JennaMarbles followed Zoella  Zoella followed AmazingPhil  Christy followed Zoella  Zoella followed Christy  JacksGap joined The V-Logger  JacksGap followed JennaMarbles  PewDiePie joined The V-Logger  Zoella joined The V-Logger  Statistics | The V-Logger has a total of 5 vloggers in its logs.  1. AmazingPhil : 2 followers, 0 following  \* JennaMarbles  \* Zoella  2. Zoella : 1 followers, 1 following  3. JennaMarbles : 1 followers, 2 following  4. PewDiePie : 0 followers, 0 following  5. JacksGap : 0 followers, 1 following |

## \*Ranking

Here comes the final and the most interesting part – the Final ranking of the candidate-interns. The final ranking is determined by the points of the interview tasks and from the exams in SoftUni. Here is your final task. You will receive some lines of input in the format **“{contest}:{password for contest}”** until you receive **“end of contests”**. Save that data because **you will need it later**. After that you will receive other type of inputs in format **“{contest}=>{password}=>{username}=>{points}”** until you receive **“end of submissions”**. Here is what you need to do.

* Check if the **contest is valid (if you received it in the first type of input)**
* Check if the **password is correct for the given contest**
* Save the user with the contest they take part in **(a user can take part in many contests)** and the points the user has in the given contest. If you receive the **same contest and the same user update the points only if the new ones are more than the older ones.**

At the end you have to print the info for the user with the **most points** in the format **“Best candidate is {user} with total {total points} points.”**. After that print **all students ordered by their names**. **For each user print each contest with the points in descending order**. See the examples.

### Input

* strings in format **“{contest}:{password for contest}”** until the **“end of contests”** command. There will be no case with two equal contests
* strings in format **“{contest}=>{password}=>{username}=>{points}”** until the **“end of submissions”** command.
* **There will be no case with 2 or more users with same total points!**

### Output

* On the first line print the best user in format **“Best candidate is {user} with total {total points} points.”.**
* Then print all students ordered as mentioned above in format:

**{user1 name}**

**# {contest1} -> {points}**

**# {contest2} -> {points}**

**{user2 name}**

**…**

### Constraints

* the strings may contain any ASCII character except from **(:, =, >)**
* the numbers will be in range **[0 - 10000]**
* second input is always valid

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Part One Interview:success  Js Fundamentals:Pesho  C# Fundamentals:fundPass  Algorithms:fun  end of contests  C# Fundamentals=>fundPass=>Tanya=>350  Algorithms=>fun=>Tanya=>380  Part One Interview=>success=>Nikola=>120  Java Basics Exam=>pesho=>Petkan=>400  Part One Interview=>success=>Tanya=>220  OOP Advanced=>password123=>BaiIvan=>231  C# Fundamentals=>fundPass=>Tanya=>250  C# Fundamentals=>fundPass=>Nikola=>200  Js Fundamentals=>Pesho=>Tanya=>400  end of submissions | Best candidate is Tanya with total 1350 points.  Ranking:  Nikola  # C# Fundamentals -> 200  # Part One Interview -> 120  Tanya  # Js Fundamentals -> 400  # Algorithms -> 380  # C# Fundamentals -> 350  # Part One Interview -> 220 |
| Java Advanced:funpass  Part Two Interview:success  Math Concept:asdasd  Java Web Basics:forrF  end of contests  Math Concept=>ispass=>Monika=>290  Java Advanced=>funpass=>Simona=>400  Part Two Interview=>success=>Drago=>120  Java Advanced=>funpass=>Petyr=>90  Java Web Basics=>forrF=>Simona=>280  Part Two Interview=>success=>Petyr=>0  Math Concept=>asdasd=>Drago=>250  Part Two Interview=>success=>Simona=>200  end of submissions | Best candidate is Simona with total 880 points.  Ranking:  Drago  # Math Concept -> 250  # Part Two Interview -> 120  Petyr  # Java Advanced -> 90  # Part Two Interview -> 0  Simona  # Java Advanced -> 400  # Java Web Basics -> 280  # Part Two Interview -> 200 |