**Exercises: Sets and Maps Advanced**

This document defines the exercises for ["Java Advanced" course @ Software University](https://softuni.bg/modules/59/java-advanced). Please submit your solutions (source code) of all below described problems in [Judge](https://judge.softuni.bg/Contests/1463/Sets-And-Maps-Exercises).

* **Unique Usernames**

Write a simple program that reads from the console a sequence of usernames and keeps a collection with only the unique ones. Print the collection on the console in order of insertion:

**Examples**

|  |  |
| --- | --- |
| **Input** | **Output** |
| 6  Hello  Hello  Hello  World  Hello  Greetings | Hello  World  Greetings |
| 10  Peter  Maria  Peter  George  Stephen  Maria  Alexander  Peter  Stephen  George | Peter  Maria  George  Stephen  Alexander |

* **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 element that appears in both of them, and print them in the same order at the console:

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. 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 |

* **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** |
| 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 |

* **Phonebook**

Write a program that receives some info from the console about **people** and their **phone numbers**.

You are free to choose how the data is entered. Each **entry** should have just **one name** and **one number** (both of them strings). If you receive a name that **already exists** in the phonebook, simply update its number.

After filling this simple phonebook, upon receiving the command "**search**", your program should be able to perform a search of contact by name and print her details in the format "**{name} -> {number}**". In case the contact isn't found, print "**Contact {name} does not exist.**".

**Examples**

|  |  |
| --- | --- |
| **Input** | **Output** |
| John-0888080808  **search**  Maria  John  stop | Contact Maria does not exist.  John -> 0888080808 |
| John-00359888001122  Peter-0040333111000  George-0049112233  Samuel-0047123123123  **search**  Samuel  samuel  PeTeR  Peter  stop | Samuel -> 0047123123123  Contact samuel does not exist.  Contact PeTeR does not exist.  Peter -> 0040333111000 |

* **A Miner Task**

You are given a sequence of strings, each on a **new** **line**. Every **odd** line on the console is representing a **resource** (e.g. Gold, Silver, Copper, and so on), and every **even** – **quantity**. Your task is to **collect** the resources and print them each on a **new** **line**.

**Print the resources and their quantities in the format:**

"**{resource} –> {quantity}**"

The quantities inputs will be in the range **[1 … 2 000 000 000]**.

**Examples**

|  |  |
| --- | --- |
| **Input** | **Output** |
| Gold  155  Silver  10  Copper  17  stop | Gold -> 155  Silver -> 10  Copper -> 17 |
| Gold  15555555  Silver  10000000  Copper  17000000  Copper  1700  stop | Gold -> 15555555  Silver -> 10000000  Copper -> 17001700 |

* **Fix Emails**

You are given a sequence of strings, each on a new line **until you receive the** "**stop**" **command**. The first string is a **name** of a person. On the second line, you receive his **email**. Your task is to **collect** their names and emails and **remove** emails whose domain ends with "us", "uk" or "com" (case insensitive). Print in the following format:

"**{name} – > {email}**"

**Examples**

|  |  |
| --- | --- |
| **Input** | **Output** |
| John  johnDoe@softuni.org  Peter Smith  smith.peter@softuni.org  Taylor Baker  baker@gmail.com  stop | John -> johnDoe@softuni.org  Peter Smith -> smith.peter@softuni.org |
| Peter Adamas  peter\_adams@gmail.com  Anna Foster  foster.anna@yahoo.com  Duke Jenkins  jenkins.duke@softuni.org  stop | Duke Jenkins -> jenkins.duke@softuni.org |