# Exercise: Tuples and Sets

Problems for exercise and homework for the [Python Advanced Course @SoftUni](https://softuni.bg/courses/python-advanced). Submit your solutions in the SoftUni judge system at <https://judge.softuni.bg/Contests/1833>

## Unique Usernames

Write a program that reads from the console a sequence of **N usernames** and keeps a collection only of the **unique** ones. On the **first** line you will be given an integer **N**. On the next **N** lines, you will receive **one** username **per** **line**. Print the collection on the console (the order does **not matter**):

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 6  George  George  George  Peter  George  NiceGuy1234 | George  Peter  NiceGuy1234 |

## Sets of Elements

Write a program that prints a **set of elements**. On the first line you will receive two numbers - **n** and **m**, which represent the lengths of two separate sets. On the next **n** + **m** lines you will receive **n** numbers, which are the numbers in the **first** set, and **m** numbers, which are in the **second** set. Find all the **unique** **elements** that appear in **both of them** and **print** them on **separate lines** (the order **does not matter**).

**For example:**

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

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

Set that contains all the **elements** that repeat in **both** **sets** -> {**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

Write a program that keeps all the **unique** chemical **elements**. On the first line you will be given a number **n** - the **count** of input **lines** that you are going to receive. On the next **n** lines, you will be receiving **chemical** **compounds**, separated by a **single** **space**. Your task is to print all the **unique ones** on a separate lines (**order does not matter**):

### 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 |

## Phonebook

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

You are free to choose the way the data is entered; each **entry** should have just **one name** and **one number** (both 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 a contact by name and print her details in format "**{name} -> {number}**". In case the contact isn't found, print "**Contact {name} does not exist.**" Examples:

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Adam-0888080808  **search**  Mery  Adam  stop | Contact Mery does not exist.  Adam -> 0888080808 |
| Adam-+359888001122  Ralf-666  George-5559393  Silvester-02/987665544  **search**  Silvester  silvester  Rolf  Ralf  stop | Silvester -> 02/987665544  Contact silvester does not exist.  Contact Rolf does not exist.  Ralf -> 666 |

## Longest Intersection

Write a program that finds the **longest intersection**. You will be given a number **N**. On the next **N lines** you will be given **two ranges** in the format: **"{first\_start},{first\_end}-{second\_start},{second\_end}"**. Find the **intersection** of these two ranges and **save the longest one of all** N intersections. At the end print the **numbers** that are included in the longest intersection and its length in the format: **"Longest intersection is {longest\_intersection} with length {length\_longest\_intersection}"**

***Note: in each 2 ranges there will always be intersection. If there are two equal intersections, print the first one.***

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comment** |
| 3  0,3-1,2  2,10-3,5  6,15-3,10 | Longest intersection is [6, 7, 8, 9, 10] with length 5 | The intersection of [0-3] and  [1-2] is [1-2] (length 2)  The intersection of [2-10] and  [3-5] is [3-5] (length 3)  The intersection of [6-15] and  [3-10] is [6-10] (length 5) - which is the longest |
| 5  0,10-2,5  3,8-1,7  1,8-2,4  4,7-2,5  1,10-2,11 | Longest intersection is [2, 3, 4, 5, 6, 7, 8, 9, 10] with length 9 |  |