# Exercise: Tuples and Sets

Problems for exercise and homework for the [Python Advanced Course @SoftUni](https://softuni.bg/trainings/3963/python-advanced-january-2023).

Submit your solutions in the SoftUni judge system at <https://judge.softuni.org/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 receive an integer **N**. On the next **N** lines, you will receive **a username**. Print the collection on the console (the order does **not matter**):

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 6  George  George  George  Peter  George  NiceGuy1234 | George  Peter  NiceGuy1234 |
| 10  Peter  Maria  Peter  George  Steve  Maria  Alex  Peter  Steve  George | Peter  Maria  George  Steve  Alex |

## Sets of Elements

Write a program that prints a **set of elements**. On **the first line**, you will receive **two numbers** - **n** and **m**, separated by a single space - representing 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** 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 will receive. On the following **n** lines, you will be receiving **chemical** **compounds** separated by a **single** **space**. Your task is to print all the **unique ones** on separate lines (the **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 a **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 | Why do you like Python? | : 4 time/s  ?: 1 time/s  P: 1 time/s  W: 1 time/s  d: 1 time/s  e: 1 time/s  h: 2 time/s  i: 1 time/s  k: 1 time/s  l: 1 time/s  n: 1 time/s  o: 3 time/s  t: 1 time/s  u: 1 time/s  y: 3 time/s |

## Longest Intersection

Write a program that finds the **longest intersection**. You will be given a number **N**. On each of the next **N lines** you will be given **two ranges** in the format: **"{first\_start},{first\_end}-{second\_start},{second\_end}"**. You should find the **intersection** of these two ranges. The **start** and **end** **numbers** in the ranges are **inclusive**.

Finally, you should **find the longest intersection** **of all N** intersections, print the **numbers** that are included and its length in the format: **"Longest intersection is [{longest\_intersection\_numbers}] with length {length\_longest\_intersection}"**

***Note: in each range, there will always be an 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 |  |

## Battle of Names

You will receive a **number N**. On the following **N** lines, you will be receiving **names**. You should **sum the ASCII values** of each letter in the name and **integer divide it by** **the number** of the **current row (starting from 1)**. **Save the result** to a set of **either odd or even** numbers, depending on if the resulting number is **odd or even**. After that, **sum the values of each set**.

* If the **sums of the two sets are equal**, print the **union of the values**, separated by **", "**.
* If the **sum of the odd numbers is bigger than the sum of the even numbers**, print the **different values**, separated by **", "**.
* If the **sum of the even numbers is bigger than the sum of the odd numbers**, print the **symmetric-different values**, separated by **", "**.

***NOTE: On every operation, the starting set should be the odd set***

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comment** |
| 4  Pesho  Stefan  Stamat  Gosho | 304, 128, 206, 511 | **First** name: Pesho. The sum of the ASCII values is: 80 + 101 + 115 + 104 + 111 = 511. Integer divide the sum to the current row (1): 511 / 1 = 511.  **Second** name: Stefan. The sum of the ASCII values is: 83 + 116 + 101 + 102 + 97 + 110 = 609. Integer divide the sum to the current row (2): 609 / 2 = 304.  **Third** name: Stamat. The sum of the ASCII values is: 83 + 116 + 97 + 109 + 97 + 116 = 618. Integer divide the sum to the current row (3): 618 / 3 = 206.  **Fourth** name: Gosho. The sum of the ASCII values is: 71 + 111 + 115 + 104 + 111 = 512. Integer divide the sum to the current row (4): 512 / 4 = 128.  **The odd set**: 511  **The even set**: 304, 206, 128  The sum of the even numbers is larger, so we print the **symmetric-different** **values**. |
| 6  Preslav  Gosho  Ivan  Stamat  Pesho  Stefan | 733, 101 |  |