# Problem 4 – Hornet Armada

The Hornet Overlord Nostalgia, who is famed for his absolute discipline and strict orders, owns the most sorted army in the Hornet history. Help Nostalgia “computerize” the process of sorting out his army.

You will be given **N** – an integer.  
On the next **N** lines you will be given input containing information about soldiers in the following format:

**{lastActivity} = {legionName} -> {soldierType}:{soldierCount}**

The **last activity** is an **integer**. The **legion name** and **soldier type**, will both be **strings**. The **soldier count** will be an **integer**. You must **store** **every** **legion** with its **activity**, and **every** **soldier type** with its **count**, in its **legion**.

If a **given legion already exists**, you must **add** the new **soldier type**, with its count. If the soldier type exists **ALSO**, you should just **add** the **soldier count**.

**IN** **BOTH** cases, stated above, you should **update** the **last** **activity**, with the newly entered one, **ONLY** if the **entered** **one** is **GREATER** than the **previous one**.

After you’ve read **all N** input lines, you will receive a line in one of the following formats:

* {activity}\{soldierType}
* {soldierType}

In the **first case,** you must print all **legions**, and the **count of soldiers** they have from the **given** **soldier** **type**, who’s **last activity** is **LOWER** than the **given activity**. The legions must be printed in **descending order** by **soldier count**.

In the **second case**, you must print all legions which **have** the **given soldier type**, with **last activity**, and **legion name**. The legions must be printed in **descending** **order** oftheir **activity**.

### Input

* On the first line you will receive **N** –the **integer**.
* On the next **N** lines you will receive data about **soldiers** and **legions**.
* On the last line you will receive **one** of the **two commands**, which will **determine** the **output**.

### Output

* If you are given the **last activity** and **soldier type** on the last command, you must print the legions in this format:
* {legionName} -> {soldierCount}
* If you are given **only** the **soldier type** on the last command, you must print the legions in this format:
* {lastActivity} : {legionName}

### Constrains

* The first integer – **N**, will be in **range [0; 10,000]**.
* The **legion names** and **soldier types** may consist of **any ASCII** character, except “**=**”, “**-**”, “**>**”, “**:**”, “ ”(**space**).
* The **soldier count** and **last activity** will be integers in **range [0; 1,000,000,000]**.
* All input data will be exactly as stated above. There will be **NO invalid** input lines.
* Data which has **NO specified order** must be sorted in **order of** **input**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 6  1 = BlackBeatles -> Soldier:2000  2 = BlackBeatles -> Worker:1000  1 = Red\_Ones -> Soldier:10000  5 = Rm -> Soldier:30000  2 = Red\_Ones -> Soldier:20000  10 = RND -> Soldier:100000  10\Soldier | Red\_Ones -> 30000  Rm -> 30000  BlackBeatles -> 2000 |
| 7  1000 = F1rstL3gion -> Aisers:15000  500 = F1rstL3gion -> Aisers:1000  200 = F1rstL3gion -> Guards:2000  2000 = Second!egion -> Guards:2000  1500 = Second!egion -> Aisers:15000  2500 = Second!egion -> Spies:2000  1000 = Forked\_Ones -> Guards:10000000  Guards | 2500 : Second!egion  1000 : F1rstL3gion  1000 : Forked\_Ones |