Problem 1 – Shockwave

Doctor T. decided to experiment with shockwaves. He performs his studies in a **rectangular** room of size **N * M**. However what is interesting is that the shockwaves are also rectangular, through some unknown properties of matter. There are pressure plates which indicate each time a shockwave hits a certain area in the room.

You will be given **N** and **M** – **integers** indicating the **size** of the room – in **rows** and **columns**. There is a plate in each column. In the beginning the plates have value – **0**, because no shockwaves have been executed.

After that you will begin receiving lines of exactly **4 integers**, **separated** by a **single space** – **X1**, **Y1**, **X2**, **Y2**. These are **coordinates** of the **points** which are the **upper-left** and the **bottom-right corners** of the rectangular shockwave. You must **increase** the **value** of **each plate** in **the area** the **rectangular shockwave affects**, by **1**. You must perform this for **every shockwave input**.

When you receive the command "**Here We Go**", that means Doctor T. is ready to check the results of the shockwaves. You must **print all the plates** on **rows**, each plate's value **separated** by a **space**.

Input

- On the first line of input you will receive **N** and **M** – **separated** by a **space**.
- On the next several lines you will receive the **4 integers** – separated by a space, until you receive the command "**Here We Go**".

Output

- Print all of the rows of the room, **each** on a **new line**, with all **columns** separated by a **single space**.

Constrains

- The dimensions of the room - **N** and **M**, will be valid integers in **range [1, 25]**.
- Each of the given integers – **X1**, **Y1**, **X2**, **Y2**, will be **inside** the **valid room indexes**.
- **X2** will **always** be **greater than** or **equal to X1**, and **Y2** will **always** be **greater than** or **equal to Y1**.

Examples

Input	Output	Comments

5 5
0 0 2 2
2 2 4 4
Here We Go
	1 1 1 0 0
	1 1 1 0 0
	1 1 2 1 1
	0 0 1 1 1
	0 0 1 1 1
		We have a room with size – 5, 5. And we receive a shockwave from [0, 0] to [2, 2], so we **increase** all of the plates' values in that range with **1**.
		1 1 1 0 0
		1 1 1 0 0
		1 1 1 0 0
		0 0 0 0 0
		0 0 0 0 0
		Then we receive the next one which is from [2, 2] to [4, 4].
		1 1 1 0 0
		1 1 1 0 0
		1 1 2 1 1
		0 0 1 1 1
		0 0 1 1 1
		So we just increase with the plates' values in the given range with **1**. One of the plates gets hit a

		second time so its value becomes **2.**

7 7
0 0 1 1
1 1 2 2
2 2 3 3
3 3 4 4
4 4 5 5
5 5 6 6
Here We Go
	1 1 0 0 0 0 0
	1 2 1 0 0 0 0
	0 1 2 1 0 0 0
	0 0 1 2 1 0 0
	0 0 0 1 2 1 0
	0 0 0 0 1 2 1
	0 0 0 0 0 1 1	

Problem 2 - Little Alchemy

Professor X. has finally come up with an idea how to become rich. He designed a special **acid**, which applied in a precise amount, transforms **stone** into pure **gold**. Calculating how much acid is needed for each stone is hard, that's why Professor X has **labeled each stone with a number**, showing how many **doses** it needs to turn into the valuable metal.

The stones are laid one after another on a table and every time he takes **the first (leftmost) stone**, he applies acid on it and returns it **on the (right) end** of the sequence. For example, he starts with stones **2 3 4**. On the first turn he will take **2**, apply acid and change the label on it to **1** (because there is only one dose left to be applied). Then, he places it on the **end of the sequence** and continue doing so.

2 3 4 -> 3 4 1 -> 4 1 2 -> 1 2 3 …

When the required amount of acid is applied on a stone **(label becomes 0)**, it is moved from the table to the **storage** – a secret place, where Professor X keeps the gold. Because oxygen can revert the effect of the acid, he keeps the storage **air free**.

Unfortunately, there are some **air leaks** into the storage, which make the gold **pieces turn back into stones**. Every time there is a leak, the **last piece that got into the storage gets damaged** and the Professor has to calculate again the acid doses he needs for it to turn in back into gold. After labeling the stone again, he returns it back to the table - **at the end** of the sequence.

Input

- On the first line of the input you will receive the **sequence of the stones** the Professor has at the beginning, each represented by an **integer – its label**. The stones will be separated **by space**.
- Afterwards you will get a series of commands of the following types:
  - **"Apply acid {n}"** – Prof. X applies **n** doses of acid on **n** number of stones and moves each at the **end**. If there are **no** stones at the table, **ignore** the command.
  - **"Air Leak {m}"** – there is a leak in the storage, the **last piece gets damaged** and must be treated with **n** more doses to turn into gold again. If the storage is **empty**, **ignore** the command.
  - "**Revision**" – end of input.

Output

After receiving the "**Revision**" command you should print

- On the first line - all **stones** left, **in their current order on the table**
- On the second line - the amount of **gold pieces in the storage**

Constrains

- Each of the integers in the input will be in the range **[1…3000]**.
- The sequence will consist of **[1…10000]** integers.
- **All data** must be processed **by order of input**.
- Allowed time/memory: **200ms/16MB**.

Examples

Input	Output	Comments
1 2 1 4 5 Apply acid 2 Apply acid 1 Air leak 5 Apply acid 3 Revision	5 3 4 2	**First** and **second** stones get **1** dose of acid, so the first **turns into gold** and the second is put at the end of the sequence with **decremented** label: **1 4 5 1.** **First** stone turns into gold: **4 5 1.** Air leaks into the storage and the l**ast piece of gold** transforms into **stone** with label "**5**": **4 5 1 5.** **First**, **second** and **third** stones get 1 dose of acid and only the **second turns into gold**: **5 3 4.** At the end there are **2** pieces of **gold** in the **storage**.
5 5 1 2 5 Apply acid 2 Air leak 5 Apply acid 1 Revision	2 5 4 4 1	**1 2 5 4 4** **1 2 5 4 4** (There is **nothing** in the storage, so the command is **ignored**) **2 5 4 4**

Problem 3 – Ascent

Doctor T. likes to communicate with encrypted messages with his lackeys. But the lackeys are relatively stupid, so you need to help them understand the Doctor's messages, by decrypting his encrypted data.

You have been tasked to decrypt several messages. You will be given the messages as several input lines. You must read and decrypt messages until you receive the command "**Ascend**".

When you receive a given message you must **decrypt** it and **print** it **immediately**. The messages have several encoded strings in them, which you must find. The encoded string consist of – a **single comma** (**,**) or an **underscore** (_), a **sequence of English Alphabet letters**, and a **digit** at the end of it. **Examples:** "**,asd9**", "**_z8**", "**,xzc9**".

The decoding of a message is done by **adding** or **subtracting** the **digit** in it, from the **ASCII code** of **every character** in the **sequence of letters**. If the front character is a **comma** (**,**) you **add** the **digit** to the **ASCII codes** of the characters, if it's an **underscore** (**_**), you **subtract** it.

When you find an **encoded string**, you must "**memorize**" it. In other words, you find the encoded string "**_qlfh5**", and decode it – it results in "**lgac**". If you find the same encoded string (**_qlfh5**), **again**, in a message **afterwards**, you must **replace** it with the **decoded value** corresponding to it (**lgac**), **BEFORE** beginning to decode the message itself. **Example:** "**abcdefg,_qlfh52hijklmn**" = "**abcdefg,lgac2hijklmn**" = "**abcdefgnicehijklm**".

You know that "**_qlfh5**" is "**lgac**", so the message would turn into "**abcedfg,lgac2hijklmn**" which after decoding turns into "**abcdefgnicehijklm**".

Input

- The input comes in the form of input lines containing the encoded messages.
- When you receive the command "**Ascend**" the input sequence ends.

Output

- You must print every message, **immediately** after you've decoded it.

Constrains

- The input lines may consist of any **ASCII character**.
- Upon **replacing** with **memorized** patterns, you must do it in **the order** in which you've **found the patterns**.

Examples

Input	Output
,plcqrkf3 _twn2z <3 Ascend	softuni rulz <3
This _vet4 is a ,I2_iller0 Ya ,eswx2 listenin _zu6 _Vjl9 ,,I22iller Ascend	This rap is a Killer Ya guyz listenin to Mac Miller

Problem 4 – Football Stats

You know that football is "The King of Sports". In our days football is not just a sport, it is business and like every business there is some stats that are needed. More important stats than everything is result of matches between teams. So now you will need to write a program that take care of all results between teams. At begging you will need to read all results from matches in format:

{firstTeam} - {secondTeam} result {firstTeamGoals}:{secondTeamGoals}

You will get results till you read **"Season End" command**. Then you will get one more single line with all **teams** that you need to print, **separated with ", "**. You have to print all games, that team play in format:

{team} - {opponent} result {teamGoals}:{opponentGoals}

All opponents for current team have to be print in alphabetical order.

Input

The input comes from the console as **varying number** of lines. You have to parse every command until the command that follows is "Season End"**.** The input will be in the format displayed above, there is no need to check it explicitly.

Output

For every team stats you need to print, you need to sort its opponents in alphabetical order. Result for all matches have to be in format **{teamGoals}:{opponentGoals}.**

Constraints

- The number of teams will be in the range [1..50]
- The number of results for each teams will be in the range [1..1000]
- Time limit: 0.3 sec. Memory limit: 16 MB.

Examples

Input	Output
RealMadrid - Barcelona result 5:0 RealMadrid - Atletico result 3:0 Barcelona - Atletico result 1:3 Season End RealMadrid, Atletico, Barcelona	RealMadrid - Atletico -> 3:0 RealMadrid - Barcelona -> 5:0 Atletico - Barcelona -> 3:1 Atletico - RealMadrid -> 0:3 Barcelona - Atletico -> 1:3 Barcelona - RealMadrid -> 0:5
RealMadrid - Barcelona result 5:0 Barcelona - RealMadrid result 1:2 RealMadrid - Atletico result 3:0 Atletico - RealMadrid result 0:1 Barcelona - Atletico result 1:3 Atletico - Barcelona result 0:0 Season End RealMadrid, Atletico, Barcelona	RealMadrid - Atletico -> 3:0 RealMadrid - Atletico -> 1:0 RealMadrid - Barcelona -> 5:0 RealMadrid - Barcelona -> 2:1 Atletico - Barcelona -> 3:1 Atletico - Barcelona -> 0:0 Atletico - RealMadrid -> 0:3 Atletico - RealMadrid -> 0:1 Barcelona - Atletico -> 1:3 Barcelona - Atletico -> 0:0 Barcelona - RealMadrid -> 0:5 Barcelona - RealMadrid -> 1:2