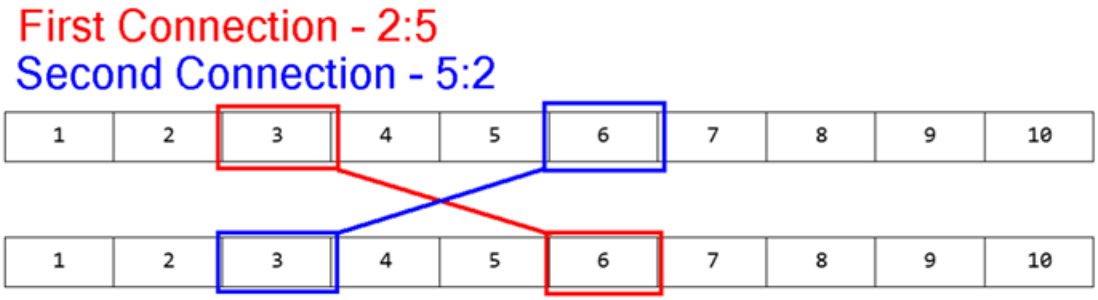
Softuniada 2019

Nexus

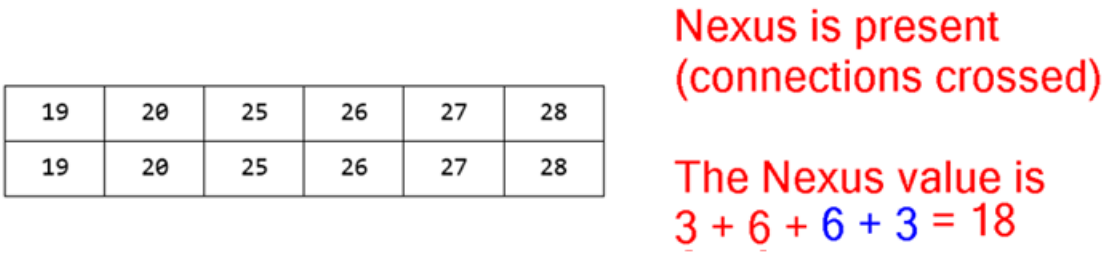
While studying arrays of various atoms, Doctor Sanity managed to discover a strange behaviour – while connecting 2 pairs of atoms from 2 parallel arrays, a nexus is initiated. If the 2 connections cross each other, the nexus disperses and loads all other atoms with nuclear value. He wanted to create an algorithm simulating this behaviour, but he's too lazy… So you'll have to write it for him.

You will receive **2 sequences** of **integers**, separated by **spaces** – the **2 arrays** of atoms.

After that you will start receiving **2 pairs** of **indices** – the **2 connections** in the **arrays**. Each connection will be in the **form** of an **index** from the **first array** and another **index** from the **second array**.



You must check if the 2 connections **cross each other**. If they **do**, you must **remove all elements between** them from **both arrays**, and you must **increase all integers**, that are **left afterwards** in **both arrays**, with the **summed up value** of the **connected elements** (**Nexus value**).



In case of **no crossing** of **connections**, you should **do nothing**.

You will continue receiving connections until you receive the command "**nexus**", at which point you must **print** what is **left** of the **arrays** and end the program.

Input

The input consists of several sequences:

- First you will receive **2 lines**, containing **sequences** of **integers**, separated by **spaces** – the arrays.
- On the next **several lines** you will receive the **pairs** of **connections** in the following format: **`{firstArrayIndex}:{secondArrayIndex}|{firstArrayIndex}:{secondArrayIndex}`**
- When you receive the command "**nexus**" the input must end.

Output

The output should consist of **2 lines**, containing what is left of the **arrays**, with its elements – **separated** by each other with a **comma** and a **space**.

Constraints

- The input will always be in valid format.
- The arrays will contain integers in **range [0, 10000]**.
- The arrays will **not necessarily** have the **same length**.
- The **indices** in the **connections** will always be **valid** and **inside** the arrays.
- Allowed time / memory: 100ms / 16MB.

Examples

Input	Output	Comment
1 2 3 4 5 6 7 8 9 10 1 2 3 4 5 6 7 8 9 10 2:5|5:2 nexus	19, 20, 25, 26, 27, 28 19, 20, 25, 26, 27, 28	See the examples above.
5 10 15 20 25 30 40 35 30 25 20 15 10 5 1:6|2:1 nexus	75, 90, 95, 100 110, 75	The **range** of the **elements** in the **second array** is **larger** than that of the **first array**. Naturally, **more elements** are **removed** from the second array.
9 5 10 4 5 6 7 10 3 3 3 4 5 6 7 8 0:1|1:0 0:1|1:0 0:1|1:0 nexus	634, 637 634, 635	