# Problem 4 – Radical Marketing

You have been given control over a large partition of the financial markets at Wall Street. Your task is to track and make logs of the important people there. The business there is quite complex, so there are a few things you must follow.

You need to implement functionality that allows you to **register people** in your log database. These people must also be **uniquely named**. There is a **specified command input** that triggers this functionality.

**People** also like to **subscribe** to **other people**, so that they can follow their actions and analyze them, which is essential to marketing.

One person **can subscribe** to **as many people** **as he wants**. He **CANNOT** however subscribe to **himself** or subscribe to a person he is **already a subscriber** of.

### Input

* The **input** comes as array of strings. Each element will represent a **valid** command. The commands are in a specific format:
  + “{person}” – **registers** a person into the log database. In case the **given** **person already exists**, **IGNORE** that command.
  + “{firstPerson}-{secondPerson}” – subscribes the first person to the second. If **one** or **both** of the **given persons** **do NOT exist** in the log, you should **IGNORE** that command

When you’ve processed the whole input, you must find **the person with** **the most subscribers**, because he is the most important person on the markets. If, however, two persons have the same amount of subscribers, you must find **that one which is a subscriber of more people**, because that would mean he is more experienced on the markets and is following and analyzing a lot of people. If even then there are equal people, pick the **first entered**.

### Output

* The **output** person should be printed along with **his subscribers**, in the following format:

“{person}

1. {subscriber1}

2. {subscriber2}

...

* The s**ubscribers** should be printed in **order of subscription**.
* If there are no subscribers, just print the **person’s name**.

### Constraints

* There will be no invalid input.
* The subscribers will be strings, which **will always be Capital English alphabet letters**.
* Allowed time/memory: 200ms/16MB.

|  |  |
| --- | --- |
| **Input** | **Output** |
| A  B  C  D  A-B  B-A  C-A  D-A | A  1. B  2. C  3. D |

|  |  |
| --- | --- |
| **Input** | **Output** |
| J  G  P  R  C  J-G  G-J  P-R  R-P  C-J | J  1. G  2. C |