Single company stock (ie IBM, MSFT, FB) are traded on exchanges. Those exchanges also provide **indices** like [SPX](https://en.wikipedia.org/wiki/S%26P_500_Index) which give investors an indicator based on multiple companies; therefore the value of an Index is based on a formula which depends on the value of multiple stocks. Those formulas are pre-determined by the exchange.

The goal of this exercise is to implement a flexible system to create **Custom Indices**. Your program will first accept configuration inputs to set up the different rules for computing those custom indices. In the second phase, the program will process stock quotes and compute the value of the custom indices.

Your program will process data from STDIN. It can be separated in 2 phases: \* Configuration: this is the setup phase that defines all the stocks and custom indices (including the formula for calculating each custom index value).. \* Processing: this phase processes the new stock values input and calculates the custom indices values based on the formulas from the configuration phase.

*Note that those 2 phases are sequential and therefore no more configuration input will be submitted in the STDIN after the first quote.*

**Constraints**:

In order to help you design your solution here are some statistics for the problem size:

* 1 < The total numbers of quotes < 107
* 1 < The number of custom indices < 104
* 1 < The number of computation < 104

**Input Format**

In the **configuration** phase the data from STDIN will have the following format:

R|Identifier|Operation|Argument0 (optional)|Argument1 (optional)| .... |Argumentn (optional)

* R identifies that this is a configuration rule.
* Identifier is the unique identifier of either the stock or the custom index
* Operation: You program should support 3 operations
  + S: This is to indicate the Identifier correspond to a stock. No further arguments are expected
  + +: This is to indicate that the custom index is the addition of 2 identifiers. 2 Arguments are therefore expected; each argument is a previously definedIdentifier
  + -: This is to indicate that the custom index is the substraction of 2 identifiers. 2 Arguments are therefore expected; each argument is a previously definedIdentifier

*Note that no identifier can be redefined twice and you can assume so in your program.*

In the **processing** phase the data from STDIN will have the following format

Q|Identifier|Value

* Q identifies that this is a market quote for a previously defined Stock. No quote will be given for custom indices.
* Identifier: is the identifier for a Stock. The identifier should have been previously setup during the configuration step. ('S' operation).
* Value: is a floating point number (with double precision) indicating the latest stock value.

**Output Format**

At the end of the processing of all the input, you will print to STDOUT the value of all custom indices in alphabetical order with the following format:

Identifier:Value

* Identifier: identifier for the custom index as previously defined during the configuration phase
* Value: floating point value of the custom index. In the case that a custom index does not have a value you should output nan. Precision should be 2.

**Sample Input 00**

R|MSFT|S

R|IBM|S

R|MSFT\_PLUS\_IBM|+|MSFT|IBM

Q|MSFT|120.

Q|IBM|130.

**Sample Output 00**

MSFT\_PLUS\_IBM: 250.00

**Explanation 00**

The second quote (IBM) should trigger the computation of the composite ticker and the expected value is 130 + 120 = 250.

**Sample Input 01**

R|MSFT|S

R|IBM|S

R|MSFT\_PLUS\_IBM|+|MSFT|IBM

Q|MSFT|120.0

**Sample Output 01**

MSFT\_PLUS\_IBM: nan

**Explanation 01**

The first quote (MSFT) is not enough data to be able to compute the composite ticker (MSFT\_PLUS\_IBM), hence the nan value.

**Sample Input 02**

R|MSFT|S

R|IBM|S

R|MSFT\_PLUS\_IBM|+|MSFT|IBM

Q|MSFT|120.

Q|IBM|130.

Q|MSFT|110.

**Sample Output 02**

MSFT\_PLUS\_IBM: 240.00

**Explanation 02**

Small extension to the previous problem. The third quote on MSFT updates trigger the composite ticker recalculation with the new MSFT value (110) and current IBM one 130.

**Sample Input 03**

R|MSFT|S

R|IBM|S

R|FB|S

R|CP1|+|MSFT|IBM

R|CP2|-|FB|IBM

R|CP12|+|CP1|CP2

Q|MSFT|120.

Q|IBM|130.

Q|FB|80.

**Sample Output 03**

CP1: 250.00

CP12: 200.00

CP2: -50.00

**Explanation 03**

In this example we introduce a composite ticker CP12 which depends on previously defined 2 composite tickers CP1 and CP2. For the first quote (MSFT), none of the composite tickers can be computed.

After the second quote (IBM), only CP1 composite ticker can be computed. Both CP1 andCP12 eventually depends on (FB) market ticker which has no value yet.

After the third quote all 3 tickers can be recomputed.

**Sample Input 04**

R|MSFT|S

R|IBM|S

R|CP1|+|MSFT|IBM

R|CP2|+|CP1|IBM

R|CP3|+|CP2|IBM

R|CP4|+|CP3|IBM

R|CP5|+|CP4|IBM

R|CP6|+|CP5|IBM

Q|MSFT|10

Q|IBM|1

Q|IBM|2

**Sample Output 04**

CP1: 12.00

CP2: 14.00

CP3: 16.00

CP4: 18.00

CP5: 20.00

CP6: 22.00

**Explanation 04**

This example essentially replicate the following formula:CPn = MSFT + n\*IBM only using the '+' operation.