Problem B. Coindays Destroyed

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	256 mebibytes

Little Alya is very fond of cryptocurrencies. Last year, she even created her own cryptocurrency: alcoin! So far, the only thing to do with an alcoin is to transfer it from one address to another. At every moment, each address contains a non-negative amount of alcoins, and a transfer from one address to another changes the amounts correspondingly. The service address `00000000` is special: transfers from it create new alcoins, and transfers to this address remove the transferred coins from circulation.

For a few months already, Alya and her friends use the distributed alcoin network for accounting between each other. But Alya wants more: to make the whole world use alcoins. One of the steps to success is to introduce alcoin to a cryptocurrency exchange, in order to facilitate buying and selling alcoin for other currencies.

The exchange service which caught Alya's attention asks, among other things, to calculate and submit several parameters of the proposed cryptocurrency, in order to estimate its reach and utility. One amazing parameter in this list is *coindays destroyed*. They are calculated as follows.

Consider an address in the cryptocurrency network and the events happening to it: the coins arrive at this address, stay there for a time, and are then transferred to other addresses. If x coins stay at the same address for y days, one can say that these coins accumulated a potential of $x \cdot y$ coindays. As soon as these x coins are transferred to another address, this potential is destroyed: that's what coindays destroyed are. The potential of other coins at the same address which didn't take part in the transfer does not get destroyed, and continues growing. If an address contains coins from several transfers at different moments, it is assumed that the outgoing transfers first use the "youngest" coins: the ones that arrived later.

For example, assume the address `abcdefgh` got 5 alcoins on Monday at midnight, and another 25 alcoins on Tuesday at midnight. After that, on Wednesday at noon, 2 alcoins are taken from this address and immediately transferred back to it: at this moment, $2 \cdot 1.5 = 3$ coindays are destroyed. After half a day more, on Thursday at midnight, 1 alcoin is transferred from this address: it destroys $1 \cdot 0.5 = 0.5$ coindays.

Finally, let the next transfer from `abcdefgh` happen instantly after the previous one and take 28.1 alcoins. Right before this happens, the address contains the following sums: 5 alcoins are there for 3 days, another 23 alcoins for 2 days, and $2 - 1 = 1$ more alcoin for 0.5 days. As a result, the transfer destroys 58.8 coindays: $1 \cdot 0.5 + 23 \cdot 2 + 4.1 \cdot 3 = 0.5 + 46 + 12.3 = 58.8$.

In the alcoin network, only few transfers happened so far, and it is easy to calculate coindays destroyed by every one of them. But Alya got thinking: how to calculate this characteristic efficiently in the future, when there will be hundreds of thousands of transfers?

Solve a generalized version of Alya's problem. Given a list of transfers, find the number of coindays destroyed by each of them.

Input

The input contains from 1 to 200 000 lines. Each line describes one transfer and has the following format: "m: s |a> r". Here, m is the moment of transfer: an integer from 1 to $2 \cdot 10^9$ denoting the number of seconds passed since the beginning of January 1, 1970, up to this transfer. The number a is the amount of alcoins being transferred: a real number given with at most four digits after the decimal point ($10^{-4} \le a \le 10^{+4}$). Finally, s and r are the sender address and the recipient address: each of them consists of exactly eight characters, and each of these characters is either a digit or a lowercase English letter.

The transfers are listed in chronological order. Initially, there are no alcoins in the system. It is guaranteed that, immediately before each transfer, the sender address contains at least the amount of alcoins required for the transfer, except when it is the service address `00000000`.

Output

For each transfer, print a single line with one real number: the number of coindays destroyed by this transfer. The transfers from the service address `00000000` are considered to destroy 0 coindays. The answer will be considered correct if, for each number, absolute or relative error does not exceed 10^{-4}.

Example

standard input	standard output
1514764800: 00000000 |5.0> abcdefgh 1514851200: 00000000 |25.0> abcdefgh 1514980800: abcdefgh |2.0> abcdefgh 1515024000: abcdefgh |1.0> 00000000 1515024000: abcdefgh |28.1> ijklmnop	0 0 3 0.5 58.8