Median Rainfall

With the changing climate patterns the researchers are interested in comparing the median amounts of rainfall to the historical data. As a junior developer on the team, you are tasked with writing a program that, for each new hourly rainfall amount generates a new median rainfall amount. Each hour a new data entry comes in with the amount of rain for that hour. The program needs to produce a new median since it might have changed from the last record.

If the current number of hourly records is odd, then the median is the middle element when the values are in sorted order. For example, for the sequence {1, 3, 6, 2, 7}, the median is 3.

If the current number of hourly records is even, then the median is the average of the two middle elements when the values are in sorted order. For example, for the sequence {1, 3, 6, 2, 7, 8}, the median is (3+6)/2 = 4 (note that the last operation uses integer division).

Input

The input consists of series of rainfall measurements `R` (`0 <= R <= 2^31`). The total number of hourly measurements `N` is less than 100,000. Each line contains only a single integer, but the numbers may have leading or trailing spaces.

Output

For each input line (each new rainfall measrument), print the current value of the median rainfall amount.

Example 1

```
Input:
1
3
4
60
70
50
2

Output:
1
2
3
3
4
27
4
```

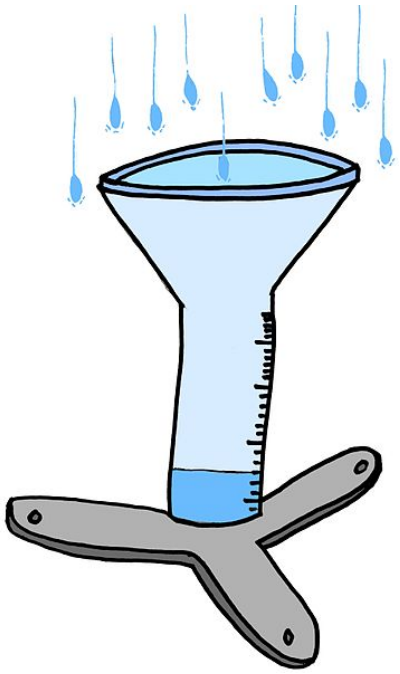

Figure 1: Cleonard1973, CC BY-SA 4.0 <https://creativecommons.org/licenses/by-sa/4.0>, via Wikimedia Commons

Laughing Monsters

The world has been attacked by Laughing Monsters. They camouflage as humans at movie theaters. Anybody sitting within D feet from a Laughing Monster will get infected and will have to endure uncontrollable laughter attacks for the next week. Moreover, once infected, these people will infect other humans sitting within D feet of them.

The movie theater owners need to be able to provide information about anybody who might have been exposed to this infection if the authorities identify a particular person to be really a camouflaged Laughing Monster.

Your job is to write a program that will identify the groups of theater guests who are in risk of infection if one of them is identified as a Laughing Monster.

Two theater guests belong to the same group if distance between their seats is less than or equal to D feet.

Input

The first line contains the number of theater guests `N` (`0 <= N <= 1,000`) and the distance `D` (a real number `0.0 <= D <= 1,000.00`). Next `N` lines have a pair of real coordinates `X Y` (where `-1,000.00 <= X, Y <= 1,000.00`) for location of each guest in the theater.

All real numbers in the input will have at most 2 (two) digits after a decimal point.

Output

Output the number of distinct groups.

Example 1

```
Input:
5 1.5
1.0 0.1
2.0 0.0
5.0 0.2
6.0 0.4
3.0 -0.1

Output:
2
```

Example 2

```
Input:
3 4.0
121.12 254.06
645.04 301.85
912.49 568.96

Output:
3
```

Ferry

Ferries used to carry cars across the river. In your village, there is still a ferry that can take up to `N` cars and needs `T` minutes to cross the river. A car may arrive at either river bank and wait to be carried to the opposite bank. The ferry operates continuously between the banks as long it is carrying at least one car or there is at least one car waiting on either side. Whenever the ferry arrives at one bank, it unloads cars carried and loads up to `N` cars waiting at that bank. When there are more than `N` cars waiting, they are loaded on the first-come-first-serve basis. If there is no car waiting on either bank, the ferry stops and waits until one car arrives. The ferry is initially on the left bank. You are asked to determine at what time each car arrives at the other bank.

Input The first line of input contains three integers `N`, `T` and `M` (`1 <= N, T, M <= 10,000`). Each of the following `M` lines gives the arrival time of a car and the bank at which the car arrives (`left` or `right`). The cars are ordered by their arrival times (so the arrival times are non-decreasing) and the time spent on loading and unloading can be ignored.

Output For each car, you should print one line containing one number, the time at which the car is unloaded at the opposite bank.

Example 1

```
Input:
2 10 10
0 left
10 left
20 left
30 left
40 left
50 left
60 left
70 left
80 left
90 left

Output:
10
30
30
50
50
70
70
90
90
110
```

Example 2

```
Input:
2 10 3
10 right
25 left
40 left

Output:
30
40
60
```

Dictionary Positions

Sarah and her friends decided to play a game they invented. In this game, a random word is chosen and everyone has to figure out its position in the dictionary. The pattern followed by their dictionary is that all the words in it are in a strict lexicographic order. Also for a given word, a valid word is the one in which the letters are in strictly ascending order with respect to their position in alphabet list. For instance, abc is a valid word but aab and acb are not.

Since the task of manually finding the position of each word is tedious and time consuming, Sarah decided to write a program to check the answer for each word. Your task is to help her develop such program that finds the corresponding entry number of the word in the dictionary for a given word.

Assume that the dictionary only holds words with maximum length of 5. Also, assume that position in the dictionary starts from 1. Positions of some of the words are as follows:

```
a -> 1
b -> 2
.
.
z -> 26
ab -> 27
ac -> 28
.
.
vwxyz -> 83681
.
.
.
```

If the input word is not valid for the dictionary as per the above mentioned rules, return 0.

Input

A string `s` corresponding to the word whose position is to be found. The string only consists of lowercase English letters

1 <= s.length <= 5

Output

An integer representing the position of the input string in dictionary or 0 if it is not a valid word.

Example 1

```
Input:
b

Output:
2
```

Example 2

```
Input:
bbbb

Output:
0
```

Example 3

```
Input:
abcd

Output:
2952
```

Example 4

```
Input:
bcdef

Output:
30552
```

Time Master

Eto manages all timers in the Light Kingdom. People usually need her to repeatedly alert them at a fixed interval. They can register a reminder with her using the following instruction:

Register *id interval*

where *id* is the ID of that reminder and *interval* is the interval between two consecutive alerts. Eto will first alert the client after *interval* number hours of registering request and then alert them every *interval* hours.

There are a bunch of different reminder-requests currently on file. Your task is to process the first N reminders for Eto. If there are more than one alerts occurring at the same time, you should list them in the ascending order of *id*.

Input

The input consists of two parts. Each line of the first part contains a single request to register a reminder in the format described above where 1 <= #Requests <= 1000, 1 <= *id* <= 3000 and 1 <= *interval* <= 3000. There are no duplicate *id*'s.

The first part is followed by a single line containing `#` character and a line containing a single integer N (1 <= N <= 10000), representing the number of alerts you need to process for Eto.

Output

You should print the *id*'s of the first N alerts. One *id* per line.

Example 1

```
Input:
Register 2004 200
Register 2005 300
#
5

Output:
2004
2005
2004
2004
2005
```