Problem A: Aaah!

Algorithms for Programming Contests

Restrictions

Time: 2 seconds
Memory: 512 MB

Problem description

Jon Marius shouted too much at the recent Justin Bieber concert, and now needs to go to the doctor because of his sore throat. The doctor's instruction is to say "aaah". Unfortunately, the doctors sometimes need Jon Marius to say "aaah" for a while, which he has never been good at. Each doctor requires a certain level of "aah" – some require "aaaaaah", while others can actually diagnose his throat with just a "h". (They often diagnose wrongly, but that is beyond the scope of this problem.) Since Jon Marius does not want to go to a doctor and have his time wasted, he wants to compare how long he manages to hold the "aaah" with the doctor's requirements. (After all, who wants to be all like "aaah" when the doctor wants you to go "aaaaaah"?)
Each day Jon Marius calls up a different doctor and asks them how long his "aaah" has to be. Find out if Jon Marius would waste his time going to the given doctor.

Input

The input consists of

- one line containing the "aaah" Jon Marius is able to say that day
- one line containing the "aah" the doctor wants to hear.

Only lowercase 'a' and 'h' will be used in the input, and each line will contain between 0 and 999 'a's, inclusive, followed by a single 'h'.

Output

Output "go" if Jon Marius should go to that doctor, and "no" otherwise.

Sample input and output

Input	Output
aaah aaaaah	no
aaah ah	go

Problem B: Backwards

Algorithms for Programming Contests

Restrictions

Time: 2 seconds
Memory: 512 MB

Problem description

Upon turning on his e-book reader yesterday evening, Mr. Enilteg discovered that a strange new bug hat reversed his entire library, letter by letter, book by book. Since he is not very tech-savvy, he asked you to help bringing everything in the right order again.

Input

The input consists of all the books, each in a separate line. Each book is a string consisting of characters, numbers and spaces. The overall size of the input is at most 1 MB.

Output

You have to reverse both the order of the books and the order of the letters in each book, so the last line should be the first, each first character of the line should be the last and so on.

Sample input and output

Input	**Output**
pey melborp drah	hard problem yep
lol a mmmh	hmmm a lol

Problem C: Simple Sorting

Algorithms for Programming Contests

Restrictions

Time: 2 seconds
Memory: 512 MB

Problem description

There is a sequence of integers. You have to sort it in descending order.

Input

The input consists of

- one line containing an integer N ($1 \leq N \leq 10^5$) – the length of the sequence
- one line containing N integers $a_1, ..., a_N$ with absolute values less than 10^9 – the sequence to be sorted.

Output

Output the integers in descending order.

Sample input and output

Input	Output
10 1 8 2 1 4 7 3 2 3 6	8 7 6 4 3 3 2 2 1 1

Problem D: Hash codes

Algorithms for Programming Contests

Restrictions

Time: 2 seconds
Memory: 512 MB

Problem description

Your university uses a very sophisticated online course management system that basically also functions as a social network. However, due to people's general unwillingness to use complicated passwords, accounts to this website frequently get hacked, leading to unfavorable impersonations in the website's forums and sometimes even the involuntary exposition of sensitive student data.

To put an end to this, the university's administration has ordered the IT team, which you belong to, to drastically improve the website's security measures. Beyond not allowing people to use overly short passwords, they also want you to limit the use of longer, but very common ones (like "password"). To realize this, a colleague of yours has already written a program that assigns each password a numeric hash code (an integer whose absolute value doesn't exceed 10^9) and saves the hash codes for all passwords in a list that is sorted by the timeline of when the respective passwords were chosen by their user. Now they want to use this list to modify the password creation in such a way that one can't choose a password that has the same hash code as one that was recently used, according to the list (due to the large number of possible hash codes it's not a problem that each of them represents many different passwords). They asked you to write a program, that, for a given hash code, finds the last (rightmost) occurrence of that hash code in the list.

Input

The input consists of

- one line containing integers N and M ($1 \leq N, M \leq 10^5$) – the length of the list of hash codes as well as the number of requests to be answered
- one line containing N integers $a_1, ..., a_N$ ($-10^9 \leq a_i \leq 10^9$) – the elements of the list

- M lines giving the requests, with the i-th of them containing an integer b_i ($-10^9 \leq b_i \leq 10^9$), denoting that the i-th request is asking for the last occurrence of hash code b_i in the list $a_1, ..., a_N$.

Output

For each request output a single line which contains the (1-based) index of the last occurrence of the hash code in the list. If the requested hash code isn't on the list, output 0.

Sample input and output

Input	**Output**
3 3 1 3 5 1 5 7	1 3 0
4 2 1 1 3 3 1 3	2 4

Problem E: Cookie selection

Algorithms for Programming Contests

Restrictions

Time: 2 seconds
Memory: 512 MB

Problem description

As chief programmer at a cookie production plant you have many responsibilities, one of them being that the cookies produced and packaged at the plant adhere to the very demanding quality standards of the Nordic Cookie Packaging Consortium (NCPC).
At any given time, your production line is producing new cookies which are stored in a holding area, awaiting packaging. From time to time there are also requests from the packaging unit to send a cookie from the holding area to be packaged. Naturally, you have programmed the system so that there are never any packaging requests sent if the holding area is empty. What complicates the matter is the fact that representatives of the NCPC might make a surprise inspection to check if your cookies are up to standard. Such an inspection consists of the NCPC representatives demanding that the next few cookies sent to the packaging unit instead be handed over to them; if they are convinced that these cookies look (and taste) the same, you pass the inspection, otherwise you fail.
Fortunately, the production plant has invested in a new measurement thingamajig capable of measuring cookie diameters with a precision of 1 nanometer (nm). Since you do not have the time to always be on the lookout for cookie craving NCPC inspectors, you have decided that a sensible strategy to cope with the inspections is to always send a cookie having the median diameter among all cookies currently in the holding area to the packaging unit on a request. If there is no cookie exhibiting the median diameter in the holding area, you instead send the smallest cookie larger than the median to the packaging unit, hoping it will please the NCPC inspectors. This means that, if the cookies were sorted in order of ascending diameter, and if there was an odd number c of cookies in the holding area, you would send the cookie at position $(c+1)/2$ in the sorted sequence, while if there was an even number c of cookies in the holding area, you would send the cookie at position $(c/2)+1$ in the sorted sequence to the packaging unit on a request.

Input

Each line of the input contains either a positive integer d, indicating that a freshly baked cookie with diameter d nm has arrived at the holding area, or the symbol '#', indicating a request from the packaging unit to send a cookie for packaging. There are at most 600 000 lines of input, and you may assume that the holding area is empty when the first cookie in the input arrives to the holding area. Furthermore, you have read somewhere that the cookie oven at the plant cannot produce cookies that have a diameter larger than 30 centimeters (cm) (or 300 000 000 nm).

Output

A sequence of lines, each indicating the diameter in nm of a cookie sent for packaging, in the same order as they are sent.

Sample input and output

Input	Output
1 2 3 4 # # # #	3 2 4 1
1 # 2 # 3 # 4 #	1 2 3 4