Problem A. Anton and Letters

Time limit	2000 ms
Mem limit	262144 kB
Input file	stdin
Output file	stdout

Recently, Anton has found a set. The set consists of small English letters. Anton carefully wrote out all the letters from the set in one line, separated by a comma. He also added an opening curved bracket at the beginning of the line and a closing curved bracket at the end of the line.

Unfortunately, from time to time Anton would forget writing some letter and write it again. He asks you to count the total number of distinct letters in his set.

Input

The first and the single line contains the set of letters. The length of the line doesn't exceed 1000. It is guaranteed that the line starts from an opening curved bracket and ends with a closing curved bracket. Between them, small English letters are listed, separated by a comma. Each comma is followed by a space.

Output

Print a single number — the number of distinct letters in Anton's set.

Examples

Input	Output
{a, b, c}	3

Input	Output
{b, a, b, a}	2

Input	Output
{}	0

Problem B. Two-gram

Time limit 1000 ms
Mem limit 262144 kB

Two-gram is an ordered pair (i.e. string of length two) of capital Latin letters. For example, "AZ", "AA", "ZA" — three distinct two-grams.

You are given a string s consisting of n capital Latin letters. Your task is to find **any** two-gram contained in the given string **as a substring** (i.e. two consecutive characters of the string) maximal number of times. For example, for string s = "BBAABBBA" the answer is two-gram "BB", which contained in s three times. In other words, find any most frequent two-gram.

Note that occurrences of the two-gram can overlap with each other.

Input

The first line of the input contains integer number n ($2 \le n \le 100$) — the length of string s. The second line of the input contains the string s consisting of n capital Latin letters.

Output

Print the only line containing exactly two capital Latin letters — **any** two-gram contained in the given string s **as a substring** (i.e. two consecutive characters of the string) maximal number of times.

Examples

Input	Output
7 ABACABA	AB

Input	Output
5 ZZZAA	ZZ

Note

In the first example "`BA`" is also valid answer.

In the second example the only two-gram "`ZZ`" can be printed because it contained in the string "`ZZZAA`" two times.

Problem C. Dawid and Bags of Candies

Time limit 1000 ms
Mem limit 262144 kB

Dawid has four bags of candies. The i-th of them contains a_i candies. Also, Dawid has two friends. He wants to give each bag to one of his two friends. Is it possible to distribute the bags in such a way that each friend receives the same amount of candies in total?

Note, that you can't keep bags for yourself or throw them away, each bag should be given to one of the friends.

Input

The only line contains four integers a_1, a_2, a_3 and a_4 ($1 \leq a_i \leq 100$) — the numbers of candies in each bag.

Output

Output `YES` if it's possible to give the bags to Dawid's friends so that both friends receive the same amount of candies, or `NO` otherwise. Each character can be printed in any case (either uppercase or lowercase).

Examples

Input	Output
1 7 11 5	YES

Input	Output
7 3 2 5	NO

Note

In the first sample test, Dawid can give the first and the third bag to the first friend, and the second and the fourth bag to the second friend. This way, each friend will receive 12 candies.

In the second sample test, it's impossible to distribute the bags.

Problem D. Thor

Time limit 2000 ms
Mem limit 262144 kB

Thor is getting used to the Earth. As a gift Loki gave him a smartphone. There are n applications on this phone. Thor is fascinated by this phone. He has only one minor issue: he can't count the number of unread notifications generated by those applications (maybe Loki put a curse on it so he can't).

q events are about to happen (in chronological order). They are of three types:

1. Application x generates a notification (this new notification is unread).
2. Thor reads all notifications generated so far by application x (he may re-read some notifications).
3. Thor reads the first t notifications generated by phone applications (notifications generated in first t events of the first type). It's guaranteed that there were at least t events of the first type before this event. Please note that he doesn't read first t unread notifications, he just reads the very first t notifications generated on his phone and he may re-read some of them in this operation.

Please help Thor and tell him the number of unread notifications after each event. You may assume that initially there are no notifications in the phone.

Input

The first line of input contains two integers n and q ($1 \le n, q \le 300\,000$) — the number of applications and the number of events to happen.

The next q lines contain the events. The i-th of these lines starts with an integer $type_i$ — type of the i-th event. If $type_i = 1$ or $type_i = 2$ then it is followed by an integer x_i. Otherwise it is followed by an integer t_i ($1 \le type_i \le 3$, $1 \le x_i \le n$, $1 \le t_i \le q$).

Output

Print the number of unread notifications after each event.

Examples

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

Input	Output
4 6 1 2 1 4 1 2 3 3 1 3 1 3	1 2 3 0 1 2

Note

In the first sample:

1. Application 3 generates a notification (there is 1 unread notification).
2. Application 1 generates a notification (there are 2 unread notifications).
3. Application 2 generates a notification (there are 3 unread notifications).
4. Thor reads the notification generated by application 3, there are 2 unread notifications left.

In the second sample test:

1. Application 2 generates a notification (there is 1 unread notification).
2. Application 4 generates a notification (there are 2 unread notifications).
3. Application 2 generates a notification (there are 3 unread notifications).
4. Thor reads first three notifications and since there are only three of them so far, there will be no unread notification left.
5. Application 3 generates a notification (there is 1 unread notification).
6. Application 3 generates a notification (there are 2 unread notifications).

Problem E. Equalize the Array

Time limit 2000 ms
Mem limit 262144 kB

Polycarp was gifted an array a of length n. Polycarp considers an array beautiful if there exists a number C, such that each number in the array occurs either zero or C times. Polycarp wants to remove some elements from the array a to make it beautiful.

For example, if $n = 6$ and $a = [1, 3, 2, 1, 4, 2]$, then the following options are possible to make the array a array beautiful:

- Polycarp removes elements at positions 2 and 5, array a becomes equal to $[1, 2, 1, 2]$;
- Polycarp removes elements at positions 1 and 6, array a becomes equal to $[3, 2, 1, 4]$;
- Polycarp removes elements at positions $1, 2$ and 6, array a becomes equal to $[2, 1, 4]$;

Help Polycarp determine the minimum number of elements to remove from the array a to make it beautiful.

Input

The first line contains one integer t ($1 \le t \le 10^4$) — the number of test cases. Then t test cases follow.

The first line of each test case consists of one integer n ($1 \le n \le 2 \cdot 10^5$) — the length of the array a.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) — array a.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output one integer — the minimum number of elements that Polycarp has to remove from the array a to make it beautiful.

Examples

Input	Output
3 6 1 3 2 1 4 2 4 100 100 4 100 8 1 2 3 3 3 2 6 6	2 1 2

Problem F. Registration System

Time limit	5000 ms
Mem limit	65536 kB
Input file	stdin
Output file	stdout

A new e-mail service "Berlandesk" is going to be opened in Berland in the near future. The site administration wants to launch their project as soon as possible, that's why they ask you to help. You're suggested to implement the prototype of site registration system. The system should work on the following principle.

Each time a new user wants to register, he sends to the system a request with his `name`. If such a `name` does not exist in the system database, it is inserted into the database, and the user gets the response `OK`, confirming the successful registration. If the `name` already exists in the system database, the system makes up a new user name, sends it to the user as a prompt and *also inserts the prompt into the database.* The new name is formed by the following rule. Numbers, starting with 1, are appended one after another to `name` (`name1`, `name2`, ...), among these numbers the least i is found so that `name`i does not yet exist in the database.

Input

The first line contains number n ($1 \le n \le 10^5$). The following n lines contain the requests to the system. Each request is a non-empty line, and consists of not more than 32 characters, which are all lowercase Latin letters.

Output

Print n lines, which are system responses to the requests: `OK` in case of successful registration, or a prompt with a new name, if the requested name is already taken.

Examples

Input	Output
4 abacaba acaba abacaba acab	OK OK abacaba1 OK

Input	Output
6 first first second second third third	OK first1 OK second1 OK third1

Problem G. Keyboard

Time limit	2000 ms
Mem limit	262144 kB
Input file	stdin
Output file	stdout

Our good friend Mole is trying to code a big message. He is typing on an unusual keyboard with characters arranged in following way:

```
qwertyuiop
asdfghjkl;
zxcvbnm,./
```

Unfortunately Mole is blind, so sometimes it is problem for him to put his hands accurately. He accidentally moved both his hands with one position to the left or to the right. That means that now he presses not a button he wants, but one neighboring button (left or right, as specified in input).

We have a sequence of characters he has typed and we want to find the original message.

Input

First line of the input contains one letter describing direction of shifting (`'L'` or `'R'` respectively for left or right).

Second line contains a sequence of characters written by Mole. The size of this sequence will be no more than 100. Sequence contains only symbols that appear on Mole's keyboard. It doesn't contain spaces as there is no space on Mole's keyboard.

It is guaranteed that even though Mole hands are moved, he is still pressing buttons on keyboard and not hitting outside it.

Output

Print a line that contains the original message.

Examples

Input	Output
R s;;upimrrfod;pbr	allyouneedislove

Problem H. Chat Order

Time limit 3000 ms
Mem limit 262144 kB

Polycarp is a big lover of killing time in social networks. A page with a chatlist in his favourite network is made so that when a message is sent to some friend, his friend's chat rises to the very top of the page. The relative order of the other chats doesn't change. If there was no chat with this friend before, then a new chat is simply inserted to the top of the list.

Assuming that the chat list is initially empty, given the sequence of Polycaprus' messages make a list of chats after all of his messages are processed. Assume that no friend wrote any message to Polycarpus.

Input

The first line contains integer n ($1 \le n \le 200\,000$) — the number of Polycarpus' messages. Next n lines enlist the message recipients in the order in which the messages were sent. The name of each participant is a non-empty sequence of lowercase English letters of length at most 10.

Output

Print all the recipients to who Polycarp talked to in the order of chats with them, from top to bottom.

Examples

Input	Output
4 alex ivan roman ivan	ivan roman alex

Input	Output
8 alina maria ekaterina darya darya ekaterina maria alina	alina maria ekaterina darya

Note

In the first test case Polycarpus first writes to friend by name "alex", and the list looks as follows:

1. alex

Then Polycarpus writes to friend by name "ivan" and the list looks as follows:

1. ivan
2. alex

Polycarpus writes the third message to friend by name "roman" and the list looks as follows:

1. roman
2. ivan
3. alex

Polycarpus writes the fourth message to friend by name "ivan", to who he has already sent a message, so the list of chats changes as follows:

1. ivan
2. roman
3. alex

Problem I. Mike and Feet

Time limit 1000 ms
Mem limit 262144 kB

Mike is the president of country What-The-Fatherland. There are n bears living in this country besides Mike. All of them are standing in a line and they are numbered from 1 to n from left to right. i-th bear is exactly a_i feet high.

A group of bears is a non-empty contiguous segment of the line. The *size* of a group is the number of bears in that group. The *strength* of a group is the minimum height of the bear in that group.

Mike is a curious to know for each x such that $1 \le x \le n$ the maximum strength among all groups of size x.

Input

The first line of input contains integer n ($1 \le n \le 2 \times 10^5$), the number of bears.

The second line contains n integers separated by space, $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$), heights of bears.

Output

Print n integers in one line. For each x from 1 to n, print the maximum strength among all groups of size x.

Examples

Input	Output
10 1 2 3 4 5 4 3 2 1 6	6 4 4 3 3 2 2 1 1 1