Codeforces Round #265 (Div. 2)

A. inc ARG

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Sergey is testing a next-generation processor. Instead of bytes the processor works with memory cells consisting of n bits. These bits are numbered from 1 to n. An integer is stored in the cell in the following way: the least significant bit is stored in the first bit of the cell, the next significant bit is stored in the second bit, and so on; the most significant bit is stored in the n-th bit.

Now Sergey wants to test the following instruction: "add 1 to the value of the cell". As a result of the instruction, the integer that is written in the cell must be increased by one; if some of the most significant bits of the resulting number do not fit into the cell, they must be discarded.

Sergey wrote certain values of the bits in the cell and is going to add one to its value. How many bits of the cell will change after the operation?

Input

The first line contains a single integer n ($1 \le n \le 100$) — the number of bits in the cell.

The second line contains a string consisting of n characters — the initial state of the cell. The first character denotes the state of the first bit of the cell. The second character denotes the second least significant bit and so on. The last character denotes the state of the most significant bit.

Output

Print a single integer — the number of bits in the cell which change their state after we add 1 to the cell.

Sample test(s)

input
4 1100
output
3

input
4 1111
output
4

Note

In the first sample the cell ends up with value 0010, in the second sample — with 0000.

B. Inbox (100500)

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Over time, Alexey's mail box got littered with too many letters. Some of them are read, while others are unread.

Alexey's mail program can either show a list of all letters or show the content of a single letter. As soon as the program shows the content of an unread letter, it becomes read letter (if the program shows the content of a read letter nothing happens). In one click he can do any of the following operations:

- Move from the list of letters to the content of any single letter.
- Return to the list of letters from single letter viewing mode.
- In single letter viewing mode, move to the next or to the previous letter in the list. You cannot move from the first letter to the previous one or from the last letter to the next one.

The program cannot delete the letters from the list or rearrange them.

Alexey wants to read all the unread letters and go watch football. Now he is viewing the list of all letters and for each letter he can see if it is read or unread. What minimum number of operations does Alexey need to perform to read all unread letters?

Input

The first line contains a single integer n ($1 \le n \le 1000$) — the number of letters in the mailbox.

The second line contains n space-separated integers (zeros and ones) — the state of the letter list. The i-th number equals either 1, if the i-th number is unread, or 0, if the i-th letter is read.

Output

Print a single number — the minimum number of operations needed to make all the letters read.

Sample test(s)

input
5 0 1 0 1 0
output
3

input
5 1 1 0 0 1
output
4

input
2 0 0
output
0

Note

In the first sample Alexey needs three operations to cope with the task: open the second letter, move to the third one, move to the fourth one.

In the second sample the action plan: open the first letter, move to the second letter, return to the list, open the fifth letter.

In the third sample all letters are already read.

C. No to Palindromes!

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Paul *hates* palindromes. He assumes that string s is *tolerable* if each its character is one of the first p letters of the English alphabet and s doesn't contain any palindrome contiguous substring of length 2 or more.

Paul has found a tolerable string s of length n. Help him find the lexicographically next tolerable string of the same length or else state that such string does not exist.

Input

The first line contains two space-separated integers: n and p ($1 \le n \le 1000$; $1 \le p \le 26$). The second line contains string s, consisting of n small English letters. It is guaranteed that the string is tolerable (according to the above definition).

Output

If the lexicographically next tolerable string of the same length exists, print it. Otherwise, print "NO" (without the quotes).

Sample test(s)

input
3 3 cba
output
NO

input
3 4 cba
output
cbd

input
4 4 abcd
output
abda

Note

String s is *lexicographically larger* (or simply *larger*) than string t with the same length, if there is number i, such that $s_1 = t_1$, ..., $s_i = t_i$, $s_{i+1} > t_{i+1}$.

The lexicographically next tolerable string is the lexicographically minimum tolerable string which is larger than the given one.

A palindrome is a string that reads the same forward or reversed.

D. Restore Cube

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Peter had a cube with non-zero length of a side. He put the cube into three-dimensional space in such a way that its vertices lay at integer points (it is possible that the cube's sides are not parallel to the coordinate axes). Then he took a piece of paper and wrote down eight lines, each containing three integers — coordinates of cube's vertex (a single line contains coordinates of a single vertex, each vertex is written exactly once), put the paper on the table and left. While Peter was away, his little brother Nick decided to play with the numbers on the paper. In one operation Nick could swap some numbers **inside a single line** (Nick didn't swap numbers from distinct lines). Nick could have performed any number of such operations.

When Peter returned and found out about Nick's mischief, he started recollecting the original coordinates. Help Peter restore the original position of the points or else state that this is impossible and the numbers were initially recorded incorrectly.

Input

Each of the eight lines contains three space-separated integers — the numbers written on the piece of paper after Nick's mischief. All numbers do not exceed 10^6 in their absolute value.

Output

If there is a way to restore the cube, then print in the first line "`YES`". In each of the next eight lines print three integers — the restored coordinates of the points. The numbers in the i-th output line must be a permutation of the numbers in i-th input line. The numbers should represent the vertices of a cube with non-zero length of a side. If there are multiple possible ways, print any of them.

If there is no valid way, print "`NO`" (without the quotes) in the first line. Do not print anything else.

Sample test(s)

input
0 0 0 0 0 1 0 0 1 0 0 1 0 1 1 0 1 1 0 1 1 1 1 1
output
YES 0 0 0 0 0 1 0 1 0 1 0 0 0 1 1 1 0 1 1 1 0 1 1 1

input
0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1
output
NO

E. Substitutes in Number

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Andrew and Eugene are playing a game. Initially, Andrew has string s, consisting of digits. Eugene sends Andrew multiple queries of type "$d_i \to t_i$", that means "replace all digits d_i in string s with substrings equal to t_i". For example, if $s = 123123$, then query "$2 \to 00$" transforms s to 10031003, and query "$3 \to$ " ("replace 3 by an empty string") transforms it to $s = 1212$. After all the queries Eugene asks Andrew to find the remainder after division of number with decimal representation equal to s by 1000000007 $(10^9 + 7)$. When you represent s as a decimal number, please ignore the leading zeroes; also if s is an empty string, then it's assumed that the number equals to zero.

Andrew got tired of processing Eugene's requests manually and he asked you to write a program for that. Help him!

Input

The first line contains string s $(1 \le |s| \le 10^5)$, consisting of digits — the string before processing all the requests.

The second line contains a single integer n $(0 \le n \le 10^5)$ — the number of queries.

The next n lines contain the descriptions of the queries. The i-th query is described by string "d_i->t_i", where d_i is exactly one digit (from 0 to 9), t_i is a string consisting of digits (t_i can be an empty string). The sum of lengths of t_i for all queries doesn't exceed 10^5. The queries are written in the order in which they need to be performed.

Output

Print a single integer — remainder of division of the resulting number by 1000000007 $(10^9 + 7)$.

Sample test(s)

input
123123 1 2->00
output
10031003

input
123123 1 3->
output
1212

input
222 2 2->0 0->7
output
777

input
1000000008 0
output
1

Note

Note that the leading zeroes are not removed from string s after the replacement (you can see it in the third sample).