A. 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).

Examples

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.

B. 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.

Examples

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
```

C. 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 $(10^9 + 7)$.

Examples

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).

D. World of Darkraft - 2

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Roma found a new character in the game "World of Darkraft - 2". In this game the character fights monsters, finds the more and more advanced stuff that lets him fight stronger monsters.

The character can equip himself with k distinct types of items. Power of each item depends on its level (positive integer number). Initially the character has one 1-level item of each of the k types.

After the victory over the monster the character finds exactly one new randomly generated item. The generation process looks as follows. Firstly the type of the item is defined; each of the k types has the same probability. Then the level of the new item is defined. Let's assume that the level of player's item of the chosen type is equal to t at the moment. Level of the new item will be chosen uniformly among integers from segment $[1; t+1]$.

From the new item and the current player's item of the same type Roma chooses the best one (i.e. the one with greater level) and equips it (if both of them has the same level Roma choses any). The remaining item is sold for coins. Roma sells an item of level x of any type for x coins.

Help Roma determine the expected number of earned coins after the victory over n monsters.

Input

The first line contains two integers, n and k ($1 \le n \le 10^5$; $1 \le k \le 100$).

Output

Print a real number — expected number of earned coins after victory over n monsters. The answer is considered correct if its relative or absolute error doesn't exceed 10^{-9}.

Examples

input

```
1 3
```

output

```
1.0000000000
```

input

```
2 1
```

output

```
2.3333333333
```

input

```
10 2
```

output

```
15.9380768924
```

E. The Classic Problem

time limit per test: 5 seconds
memory limit per test: 768 megabytes
input: standard input
output: standard output

You are given a weighted undirected graph on n vertices and m edges. Find the shortest path from vertex s to vertex t or else state that such path doesn't exist.

Input

The first line of the input contains two space-separated integers — n and m ($1 \le n \le 10^5$; $0 \le m \le 10^5$).

Next m lines contain the description of the graph edges. The i-th line contains three space-separated integers — u_i, v_i, x_i ($1 \le u_i, v_i \le n$; $0 \le x_i \le 10^5$). **That means that vertices with numbers u_i and v_i are connected by edge of length 2^{x_i} (2 to the power of x_i).**

The last line contains two space-separated integers — the numbers of vertices s and t.

The vertices are numbered from 1 to n. The graph contains no multiple edges and self-loops.

Output

In the first line print the remainder after dividing the length of the shortest path by 1000000007 $(10^9 + 7)$ if the path exists, and -1 if the path doesn't exist.

If the path exists print in the second line integer k — the number of vertices in the shortest path from vertex s to vertex t; in the third line print k space-separated integers — the vertices of the shortest path in the visiting order. The first vertex should be vertex s, the last vertex should be vertex t. If there are multiple shortest paths, print any of them.

Examples

input

```
4 4
1 4 2
1 2 0
2 3 0
3 4 0
1 4
```

output

```
3
4
1 2 3 4
```

input

```
4 3
1 2 4
2 3 5
3 4 6
1 4
```

output

```
112
4
1 2 3 4
```

input

```
4 2
1 2 0
3 4 1
1 4
```

output

```
-1
```

Note

A *path* from vertex s to vertex t is a sequence $v_0, \ldots, v_k$, such that $v_0 = s$, $v_k = t$, and for any i from 0 to $k - 1$ vertices v_i and v_{i+1} are connected by an edge.

The *length* of the path is the sum of weights of edges between v_i and v_{i+1} for all i from 0 to $k - 1$.

The *shortest path* from s to t is the path which length is minimum among all possible paths from s to t.