A. Spelling Check

time limit per test: 2 seconds
memory limit per test: 256 megabytes

Petya has noticed that when he types using a keyboard, he often presses extra buttons and adds extra letters to the words. Of course, the spell-checking system underlines the words for him and he has to click every word and choose the right variant. Petya got fed up with correcting his mistakes himself, that's why he decided to invent the function that will correct the words itself. Petya started from analyzing the case that happens to him most of the time, when all one needs is to delete one letter for the word to match a word from the dictionary. Thus, Petya faces one mini-task: he has a printed word and a word from the dictionary, and he should delete one letter from the first word to get the second one. And now the very non-trivial question that Petya faces is: which letter should he delete?

Input

The input data contains two strings, consisting of lower-case Latin letters. The length of each string is from 1 to 10^6 symbols inclusive, the first string contains exactly 1 symbol more than the second one.

Output

In the first line output the number of positions of the symbols in the first string, after the deleting of which the first string becomes identical to the second one. In the second line output space-separated positions of these symbols in increasing order. The positions are numbered starting from 1. If it is impossible to make the first string identical to the second string by deleting one symbol, output one number 0.

Examples

input	output
abdrakadabra abrakadabra	1 3

input	output
aa a	2 1 2

input	output
competition codeforces	0

B. Test

time limit per test: 2 seconds
memory limit per test: 256 megabytes

Sometimes it is hard to prepare tests for programming problems. Now Bob is preparing tests to new problem about strings — input data to his problem is one string. Bob has 3 wrong solutions to this problem. The first gives the wrong answer if the input data contains the substring s_1, the second enters an infinite loop if the input data contains the substring s_2, and the third requires too much memory if the input data contains the substring s_3. Bob wants these solutions to fail single test. What is the minimal length of test, which couldn't be passed by all three Bob's solutions?

Input

There are exactly 3 lines in the input data. The i-th line contains string s_i. All the strings are non-empty, consists of lowercase Latin letters, the length of each string doesn't exceed 10^5.

Output

Output one number — what is minimal length of the string, containing s_1, s_2 and s_3 as substrings.

Examples

input	output
ab bc cd	4

input	output
abacaba abaaba x	11

C. Camp Schedule

time limit per test: 1 second
memory limit per test: 512 megabytes

The new camp by widely-known over the country Spring Programming Camp is going to start soon. Hence, all the team of friendly curators and teachers started composing the camp's schedule. After some continuous discussion, they came up with a schedule s, which can be represented as a binary string, in which the i-th symbol is '1' if students will write the contest in the i-th day and '0' if they will have a day off.

At the last moment Gleb said that the camp will be the most productive if it runs with the schedule t (which can be described in the same format as schedule s). Since the number of days in the current may be different from number of days in schedule t, Gleb required that the camp's schedule must be altered so that the number of occurrences of t in it as a substring is maximum possible. At the same time, **the number of contest days and days off shouldn't change**, only their order may change.

Could you rearrange the schedule in the best possible way?

Input

The first line contains string s ($1 \leqslant |s| \leqslant 500\,000$), denoting the current project of the camp's schedule.

The second line contains string t ($1 \leqslant |t| \leqslant 500\,000$), denoting the optimal schedule according to Gleb.

Strings s and t contain characters '0' and '1' only.

Output

In the only line print the schedule having the largest number of substrings equal to t. Printed schedule should consist of characters '0' and '1' only and the number of zeros should be equal to the number of zeros in s and the number of ones should be equal to the number of ones in s.

In case there multiple optimal schedules, print any of them.

Examples

input	output
101101 110	110110

input	output
10010110 100011	01100011

input	output
10 11100	01

Note

In the first example there are two occurrences, one starting from first position and one starting from fourth position.

In the second example there is only one occurrence, which starts from third position. Note, that the answer is not unique. For example, if we move the first day (which is a day off) to the last position, the number of occurrences of t wouldn't change.

In the third example it's impossible to make even a single occurrence.

D. Marbles

time limit per test: 2 seconds
memory limit per test: 256 megabytes

In the spirit of the holidays, Saitama has given Genos two grid paths of length n (a weird gift even by Saitama's standards). A grid path is an ordered sequence of neighbouring squares in an infinite grid. Two squares are neighbouring if they share a side.

One example of a grid path is $(0, 0) \rightarrow (0, 1) \rightarrow (0, 2) \rightarrow (1, 2) \rightarrow (1, 1) \rightarrow (0, 1) \rightarrow (-1, 1)$. Note that squares in this sequence might be repeated, i.e. path has self intersections.

Movement within a grid path is restricted to adjacent squares within the sequence. That is, from the i-th square, one can **only move** to the $(i - 1)$-th or $(i + 1)$-th squares of this path. Note that there is only a single valid move from the first and last squares of a grid path. Also note, that even if there is some j-th square of the path that coincides with the i-th square, only moves to $(i - 1)$-th and $(i + 1)$-th squares are available. For example, from the second square in the above sequence, one can only move to either the first or third squares.

To ensure that movement is not ambiguous, the two grid paths will not have an alternating sequence of three squares. For example, a contiguous subsequence $(0, 0) \rightarrow (0, 1) \rightarrow (0, 0)$ **cannot occur** in a valid grid path.

One marble is placed on the first square of each grid path. Genos wants to get both marbles to the last square of each grid path. However, there is a catch. Whenever he moves one marble, the other marble will copy its movement if possible. For instance, if one marble moves east, then the other marble will *try* and move east as well. By *try*, we mean if moving east is a valid move, then the marble will move east.

Moving north increases the second coordinate by 1, while moving south decreases it by 1. Similarly, moving east increases first coordinate by 1, while moving west decreases it.

Given these two valid grid paths, Genos wants to know if it is possible to move both marbles to the ends of their respective paths. That is, if it is possible to move the marbles such that both marbles rest on the last square of their respective paths.

Input

The first line of the input contains a single integer n ($2 \le n \le 1\,000\,000$) — the length of the paths.

The second line of the input contains a string consisting of $n - 1$ characters (each of which is either 'N', 'E', 'S', or 'W') — the first grid path. The characters can be thought of as the sequence of moves needed to traverse the grid path. For example, the example path in the problem statement can be expressed by the string "NNESWW".

The third line of the input contains a string of $n - 1$ characters (each of which is either 'N', 'E', 'S', or 'W') — the second grid path.

Output

Print "YES" (without quotes) if it is possible for both marbles to be at the end position at the same time. Print "NO" (without quotes) otherwise. In both cases, the answer is case-insensitive.

Examples

input	output
7 NNESWW SWSWSW	YES

input	output
3 NN SS	NO

Note

In the first sample, the first grid path is the one described in the statement. Moreover, the following sequence of moves will get both marbles to the end: NNESWWSWSW.

In the second sample, no sequence of moves can get both marbles to the end.

E. Isomorphic Strings

time limit per test: 3 seconds
memory limit per test: 256 megabytes

You are given a string s of length n consisting of lowercase English letters.

For two given strings s and t, say S is the set of distinct characters of s and T is the set of distinct characters of t. The strings s and t are *isomorphic* if their lengths are equal and there is a one-to-one mapping (bijection) f between S and T for which $f(s_i) = t_i$. Formally:

1. $f(s_i) = t_i$ for any index i,
2. for any character $x \in S$ there is exactly one character $y \in T$ that $f(x) = y$,
3. for any character $y \in T$ there is exactly one character $x \in S$ that $f(x) = y$.

For example, the strings "aababc" and "bbcbcz" are isomorphic. Also the strings "aaaww" and "wwwaa" are isomorphic. The following pairs of strings are not isomorphic: "aab" and "bbb", "test" and "best".

You have to handle m queries characterized by three integers x, y, len $(1 \le x, y \le n - len + 1)$. For each query check if two substrings $s[x... x + len - 1]$ and $s[y... y + len - 1]$ are isomorphic.

Input

The first line contains two space-separated integers n and m $(1 \le n \le 2 \cdot 10^5, 1 \le m \le 2 \cdot 10^5)$ — the length of the string s and the number of queries.

The second line contains string s consisting of n lowercase English letters.

The following m lines contain a single query on each line: x_i, y_i and len_i $(1 \le x_i, y_i \le n, 1 \le len_i \le n - max(x_i, y_i) + 1)$ — the description of the pair of the substrings to check.

Output

For each query in a separate line print "YES" if substrings $s[x_i... x_i + len_i - 1]$ and $s[y_i... y_i + len_i - 1]$ are isomorphic and "NO" otherwise.

Example

input	output
7 4 abacaba 1 1 1 1 4 2 2 1 3 2 4 3	YES YES NO YES

Note

The queries in the example are following:

1. substrings "a" and "a" are isomorphic: $f(a) = a$;
2. substrings "ab" and "ca" are isomorphic: $f(a) = c$, $f(b) = a$;
3. substrings "bac" and "aba" are not isomorphic since $f(b)$ and $f(c)$ must be equal to a at same time;
4. substrings "bac" and "cab" are isomorphic: $f(b) = c$, $f(a) = a$, $f(c) = b$.

F. Palindromes with One Mistake

time limit per test: 5 seconds
memory limit per test: 256 megabytes

As you certainly know, a palindrome is a string that is the same when reversed. A palindrome with one mistake is a string with the property that you can change one character of it to make it a palindrome. An example is "dodad".

Your program should take as input a string of lower and upper case letters. The output is the longest (contiguous) substring of the input string that is a palindrome or a palindrome with one mistake (whichever is longer). If there are two that are equally long, output the leftmost one.

Input
The input is a single line of lower or upper case letters. The length is at most 10^6.

Output
Output the longest (and leftmost in case of ties) palindrome or palindrome with one mistake occurring in the input.

Examples

input	output
dogaaaaTaacat	aaaaTaaca

input	output
dogaaaTaaacat	gaaaTaaac