Problem A. Z-function to prefx-function

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You are given z-function of some (unknown for you) string s.

Write prefix-function of the string s.

Input

The first line contains one integer: n ($1 \le n \le 2 \cdot 10^5$) — the length of string s.

The second line contains z-function — n space-separated integers.

Output

Write n space-separated numbers — prefix-function of the string s.

Example

standard input	standard output
8 0 0 1 0 3 0 1 1	0 0 1 0 1 2 3 1

Note

In the example s is equal to "`ABACABAA`".

Problem B. Prefix-function to z-function

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You are given prefix-function of some (unknown for you) string s.

Write z-function of the string s.

Input

The first line contains one integer: n ($1 \le n \le 2 \cdot 10^5$) — the length of string s.

The second line contains prefix-function — n space-separated integers.

Output

Write n space-separated numbers — z-function of the string s.

Example

standard input	standard output
8 0 0 1 0 1 2 3 1	0 0 1 0 3 0 1 1

Note

In the example s is equal to "`ABACABAA`".

Problem C. Prefix-function

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Write program which finds prefix-function for given string s.

Input

The first line of the input contains string s. The length of string is between 1 and 10^6. It contains only latin letters and digits.

Output

Write required prefix-function corresponding to given string s.

Example

standard input	standard output
abacaba	0 0 1 0 1 2 3

Problem D. Z-function

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Write program which finds z-function for given string s.

Input

The first line of the input contains string s. The length of string is between 1 and 10^6. It contains only latin letters and digits.

Output

Write required z-function corresponding to given string s.

Example

standard input	standard output
abacaba	0 0 1 0 3 0 1

Problem E. Prefix-palindromes

Input file:	standard input
Output file:	standard output
Time limit:	0.5 seconds
Memory limit:	256 megabytes

You problem is to find the length of the longest prefix which is a palindrome at the same time.

Input

The only line contains s, consisting of lowercase Latin letters. The length doesn't exceed 10^5.

Output

Print the only number — the length of the longest such prefix that is a palindrome. If the whole string is palindrome you should print its length.

Examples

standard input	standard output
aaab	3
abab	3

Problem F. Two Strings

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You are given two strings a and b. Find shortest string which being repeated infinitely contains the both strings. I.e. find such shortest s that infinite string $ss \ldots s \ldots$ contains a and contains b.

Input

The first line contains number of test cases (between 1 and 100 inclusive). Each testcase consists of two lines: string a and string b. Each of them is not longer than 10000. Strings contain characters with ASCII codes from 33 to 126 inclusive.

Output

For each test case print two lines: length of the answer and answer itself. Separate answers for test cases with empty line.

Example

standard input	standard output
2 abcabc bcabca toolkit kitten	3 abc 10 toolkitten

Problem G. Cubes

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.5 seconds
Memory limit:	256 megabytes

A ghost Petya plays with his toy cubes. He places his cubes in a row and fondly regards his creation. His friends decided to play a prank a play on Petya, and brought a mirror in his playroom. Everyone knows that you can not see a ghost's reflection a mirror! But you can still see reflections of the cubes.

Now Petya sees n colored cubes in front of him, but doesn't know which ones are real, and which are only mirror reflections.

Help Petya to determine how many cubes he may have. Petya sees all cubes reflections, and some of the actual cubes in front of him. Note that some of the cubes can be behind Petya, hence he won't see these cubes.

Input

The first contains an the number of cubes n ($1 \le n \le 100000$) and the number m of different cube colors ($1 \le m \le 100000$). The next line contains n integers that are colors of the cubes seen by Petya.

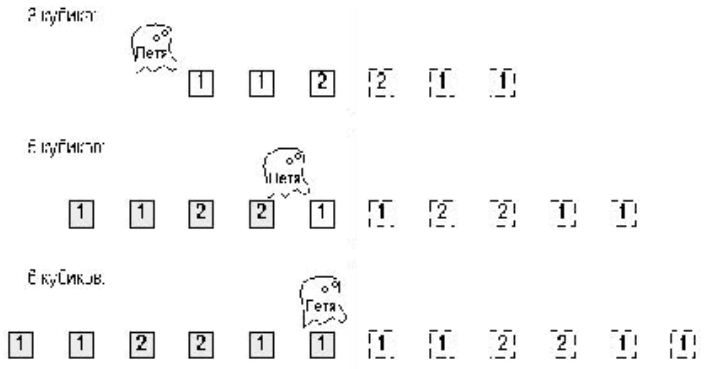
Output

Print all k such that Petya can possible have k actual cubes.

Example

standard input	standard output
6 2 1 1 2 2 1 1	3 5 6

Note



Problem H. Words

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.5 seconds
Memory limit:	256 megabytes

For given string $s = s_1 s_2 \dots s_n$ the allowed operation consists of the following steps:

- choose integer value k between 0 and n, inclusive;
- remove first k characters, but append them in the reverse order: s becomes $s_{k+1} \dots s_n \; s_k \; s_{k-1} \; \dots s_1$.

For two given strings s and t you are to check if exists such operation that the result equals to t. You can apply an operation only once.

Input

The input contains two lines: s and t. They have lengths between 1 and 50 000, inclusive. They can have different lengths.

Both strings contain only uppercase and lowercase English letters and digits.

Output

Print "`Yes`" if the required operation exists, and "`No`" if not. In case of positive answer print possible value of k. If there many valid values of k, use the minimal.

Example

standard input	standard output
wpwdwpw wdwpwpw	Yes 2

Problem I. Minimal String Period

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

In this problem p is minimal period of string s if p being repeated infinitely contains s as a prefix. I.e. if $pp\ldots p\ldots$ starts with s.

For example, for $s = abcab$ the minimal period is $p = abc$.

Input

Input contains multiple test cases. For each line of input find its minimal period. Each line is non-empty and contains lowercase and uppercase English letters. The sum of lengths of all lines doesn't exceed 8000.

Output

For each line print its minimal period.

Example

standard input	standard output
abcab aaaaaa	abc a

Problem J. Minimal String Period (Hard)

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

In this problem p is minimal period of string s if p being repeated infinitely contains s as a prefix. I.e. if $pp\ldots p\ldots$ starts with s.

For example, for $s = abcab$ the minimal period is $p = abc$.

Input

Input contains multiple test cases. For each line of input find its minimal period. Each line is non-empty and contains lowercase and uppercase English letters. The sum of lengths of all lines doesn't exceed 10^6.

Input does not contain empty lines.

Output

For each line print its minimal period.

Example

standard input	standard output
abcab aaaaaa	abc a

Problem K. Inaccurate Search

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Given text $t = t_1\ t_2\ \ldots\ t_{|t|}$ and pattern $p = p_1\ p2\ \ldots\ p_{|p|}$. Also given the parameter k.

For position d if $t[d\ \ldots\ d + |p| - 1] = p$ then there is an exact matching from the position d.

For position d if in substring $t[d\ \ldots\ d + |p| - 1]$ there is a window (segment) of f consecutive characters which can be changed to make the substring be equal to p and $f \leq k$ then it is inaccurate matching. Obviously, an exact matching is special case of an inaccurate matching.

For example, if $k = 3$, t="`abacabadabacaba`" and p="`baxayad`", there is one inaccurate matchings from the position 2 (indices are 1-based).

Your task is to find all inaccurate matchings of p for given text t.

Input

The first line contains the text t. The second line contains the pattern p. Both of them contain only Latin letters. Their lengths are between 1 and 10^6, inclusive. The third line contains k ($0 \leq k \leq 10^6$).

Output

Print the number of inaccurate matchings and the corresponding positions in increasing order.

Examples

standard input	standard output
abacaaa aaa 1	4 1 3 4 5
ababa aaaaa 3	1 1

Problem L. Olympiad string

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

On one very famous olympiad programming site you can enter some additional information about your team. You can enter the full name of your team, the name of your university, city and so on. Entering this information is not very easy, because the authors of the site have added some JavaScript to the corresponding web-page, so during entering the information you are limited with next two operations:

1. Enter one symbol to the beginning of the line, or
2. Copy any part of already typed string to the beginning of the line.

First you wanted to find such algorithm to type the name of your team that will take the minimal time. But then you realized that this algorithm will probably be not so good. Really, copying a part of a string takes more time than just typing a character, but when you are typing a character you can do a mistake (you can press a wrong key), but when you are copying a part of a string, if the algorithm is correct, you won't do any mistake. So you decided not to think about time, but find such algorithm to enter the name of your team, which will require the minimal possible number of operations. Write a program, which will find such a way for any string.

Input

Input will consist of several tests. For each test one string of lowercase and uppercase latin letters will be given — the string, for which you have to find the optimal algorithm. The length of each string will not exceed 4 000.

Output

For each test, write to output the algorithm, which will require the minimal number of operations. Start describing each algorithm with string "Typing this string will require K operations" (without quotes), where K is the number of operations in your algorithm. Then output one line with text "These operations are the following:" without quotes. Then output K lines describing the operations. For each operation, output it type (`1` for typing a character, or `2` for copying a string). If the type is `1`, output then one space and then the character itself. If the type is `2`, output then two numbers — the positions of the first and the last characters of copied part in the string before performing this operation.

If several solutions exist, output any.

Example

standard input	standard output
dabfabc a	Typing this string will require 6 operations These operations are the following: 1 c 1 b 1 a 1 f 2 2 3 1 d Typing this string will require 0 operations These operations are the following: Typing this string will require 1 operations These operations are the following: 1 a

Problem M. Retrostring

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.5 seconds
Memory limit:	256 megabytes

A string S is a sequence of characters $S_1, S_2, \ldots, S_n$, where $|S| = n$ is the *length* of the string S.

For each k ($1 \le k \le |S|$) the *k-th prefix* of the string S is a string $S_1, S_1, \ldots, S_k$ of length k. If $k < |S|$, then the prefix is *proper*.

Similarly, for each k ($1 \le k \le |S|$) the *k-th suffix* of S is a string $S_{|S|-k+1}, S_{|S|-k+2}, \ldots, S_{|S|}$ of length k. If $k < |S|$, then the suffix is *proper*.

Let the *repeatability number* of a string S be equal to the number of its proper suffices which are equal to the respective prefix of the same length. For example, repeatability number of s="`abacaba`" is 2 (the corresponding suffices are "`aba`" and "`a`"), repeatability number of s="`aaaaaa`" is 5.

A string is called a *retrostring*, if its repeatability number is strictly larger than repeatability numbers of all its proper prefixes.

You are given a string S. Find its largest prefix (not necessarily proper) which is a retrostring.

Input

The first line contains a string S ($1 \le |S| \le 1000000$). The string contains only the ASCII characters with codes from 33 to 126.

Output

In the only string print the largest prefix of S which is a retrostring.

Examples

standard input	standard output
z	z
aabaabaaabaabaaabaaba	aabaabaaabaabaa
abacabaaaa	abacaba

Problem N. Prefix-function of Gray Strings

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.5 seconds
Memory limit:	256 megabytes

For string S the prefix-function is $\pi[i] = \max\{k|0 \le k < i, S[1..k] = S[i-k+1..i]\}$ (here the indices are from 1 to $|S|$). For example, for $S = abacabaa$ the prefix-function is: $[0, 0, 1, 0, 1, 2, 3, 1]$.

The k-th Gray string is: $G_k = G_{k-1} + c(k) + G_{k-1}$, where $c(k)$ is the k-th character from the alphabet, and G_0 is empty string. For example, $G_4 = abacabadabacaba$.

Your task is for each given pair k, p find the value $\pi[p]$ for $S = G_k$.

Input

The input contains one or more lines. Each line contains a pair of integers k, p ($1 \le k \le 26$, $0 \le p < 2^k - 1$). There are at most 10 pairs.

Output

For each pair print the answer on a separate line.

Example

standard input	standard output
4 0	0
4 1	0
4 2	1
4 3	0
4 4	1
4 5	2
4 6	3
4 7	0
4 8	1
4 9	2

Problem O. Electronic Display

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.5 seconds
Memory limit:	256 megabytes

New netbook lock is a puzzle not only for the robbers, but also for their owners. At every moment of time some sequence of zeroes and once is shown on a display. Lock will open only if some fixed sequence is shown. To get the desired sequence you can press two buttons: 0 and 1.

When you press button 0 the current sequence is shift one position right (rightmost digit disappears) and 0 is added at the left. When you press 1 the same thing happen, the only difference is that the digit added at the left is 1.

You know the current combination being shows and the combination that opens the lock. The goal is to determine the minimum number of times one needs to press the button in order to open the lock (get the desired combination from the current).

Input

The first line contain the current sequence of digits while the second line contain a sequence that opens the lock. It's guaranteed that both sequences are non-empty, have equal length not exceeding 10^5 and consist of only 0 and 1. Digits are given without any whitespaces.

Output

Print one integer — the minimum number of button pressings required to open the lock.

Examples

standard input	standard output
1101 1011	2
0000 1111	4

Problem P. Fibonacci Strings

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

Memory limit is 1024 megabytes.

The sequence $f_0\ f_1\ \ldots$ is Fibonacci Strings if:

- f_0 = “”
- f_1 = “a”
- f_2 = “b”
- f_3 = “ab”
- f_4 = “bab”
- ...
- $f_i = f_{i-2} + f_{i-1}$

Given string s containing only letters ‘a’ and ‘b’ and nonnegative integer k.

Find the number of occurrences string s in the string f_k modulo $1\,000\,000\,009$ $(10^9 + 9)$.

Input

The first line contains non-empty s, the length of s is not greater than 10^3. It contains only ‘a’ and ‘b’.

The second line contains integer k $(0 \le k \le 10^4)$.

Output

Print the number of occurrences modulo $1\,000\,000\,009$ $(10^9 + 9)$.

Examples

standard input	standard output
b 4	2
ab 5	2

Problem Q. Epigraph

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Young writer has just finished the text of his first novel. As an epigraph he has chosen the sequence obtained as a concatenation of all distinct substrings of the novel name.

Substring of a given string is a sequence of consecutive characters. You are asked to find the length of the resulting epigraph.

Input

The only line contains the name of the novel consisting of lowercase and uppercase latin letters. Its length doesn't exceed 8000 characters.

Output

Print one integer — the length of the epigraph.

Example

standard input	standard output
BOBR	19

Note

In the first sample, epigraph is obtained using substrings "B", "O", "R", "BO", "OB", "BR", "BOB", "OBR", and "BOBR". Total length is 19.

Problem R. Maximal XOR

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Given n integers $a_1, a_2, \ldots, a_n$. Over all pairs of indices $1 \le i, j \le n$, $i \ne j$ find maximal value of a_i `xor` a_j.

The operation `xor` is logical exclusive OR operation on each pair of corresponding bits (denoted as `^` in most modern programming languages). For example, 5 `xor` $3 = 6$.

Input

The first line contains integer n ($2 \le n \le 2 \cdot 10^5$) — number of elements in a. The second line contains n integers a_i ($0 \le a_i \le 4 \cdot 10^{18}$).

Output

Print the maximal value of a_i `xor` a_j.

Examples

standard input	standard output
3 1 2 3	3
3 2 2 0	2
5 1 3 4 5 7	7