Codeforces Round #138 (Div. 2)

A. Parallelepiped

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

You've got a rectangular parallelepiped with integer edge lengths. You know the areas of its three faces that have a common vertex. Your task is to find the sum of lengths of all 12 edges of this parallelepiped.

Input

The first and the single line contains three space-separated integers — the areas of the parallelepiped's faces. The area's values are positive (> 0) and do not exceed 10^4. It is guaranteed that there exists at least one parallelepiped that satisfies the problem statement.

Output

Print a single number — the sum of all edges of the parallelepiped.

Examples

input
1 1 1
output
12

input
4 6 6
output
28

Note

In the first sample the parallelepiped has sizes $1 \times 1 \times 1$, in the second one — $2 \times 2 \times 3$.

B. Array

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

You've got an array a, consisting of n integers: $a_1, a_2, \ldots, a_n$. Your task is to find a minimal by inclusion segment $[l, r]$ $(1 \le l \le r \le n)$ such, that among numbers $a_l, a_{l+1}, \ldots, a_r$ there are exactly k distinct numbers.

Segment $[l, r]$ $(1 \le l \le r \le n; l, r$ are integers) of length $m = r - l + 1$, satisfying the given property, is called *minimal by inclusion*, if there is no segment $[x, y]$ satisfying the property and less then m in length, such that $1 \le l \le x \le y \le r \le n$. Note that the segment $[l, r]$ doesn't have to be minimal in length among all segments, satisfying the given property.

Input

The first line contains two space-separated integers: n and k $(1 \le n, k \le 10^5)$. The second line contains n space-separated integers $a_1, a_2, \ldots, a_n$ — elements of the array a $(1 \le a_i \le 10^5)$.

Output

Print a space-separated pair of integers l and r $(1 \le l \le r \le n)$ such, that the segment $[l, r]$ is the answer to the problem. If the sought segment does not exist, print "-1 -1" without the quotes. If there are multiple correct answers, print any of them.

Examples

input
4 2 1 2 2 3
output
1 2

input
8 3 1 1 2 2 3 3 4 5
output
2 5

input
7 4 4 7 7 4 7 4 7
output
-1 -1

Note

In the first sample among numbers a_1 and a_2 there are exactly two distinct numbers.

In the second sample segment $[2, 5]$ is a minimal by inclusion segment with three distinct numbers, but it is not minimal in length among such segments.

In the third sample there is no segment with four distinct numbers.

C. Bracket Sequence

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

A *bracket sequence* is a string, containing only characters "(", ")", "[" and "]".

A *correct bracket sequence* is a bracket sequence that can be transformed into a correct arithmetic expression by inserting characters "1" and "+" between the original characters of the sequence. For example, bracket sequences "()[]", "([])" are correct (the resulting expressions are: "(1)+[1]", "([1+1]+1)"), and "](" and "[" are not. **The empty string is a correct bracket sequence by definition.**

A *substring* $s[l \ldots r]$ $(1 \le l \le r \le |s|)$ of string $s = s_1 s_2 \ldots s_{|s|}$ (where $|s|$ is the length of string s) is the string $s_l s_{l+1} \ldots s_r$. **The empty string is a substring of any string by definition**.

You are given a bracket sequence, not necessarily correct. Find its substring which is a correct bracket sequence and contains as many opening square brackets «[» as possible.

Input

The first and the only line contains the bracket sequence as a string, consisting only of characters "(", ")", "[" and "]". It is guaranteed that the string is non-empty and its length doesn't exceed 10^5 characters.

Output

In the first line print a single integer — the number of brackets «[» in the required bracket sequence. In the second line print the optimal sequence. If there are more than one optimal solutions print any of them.

Examples

input
([])
output
1 ([])

input
(((
output
0

D. Two Strings

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

A *subsequence* of length $|x|$ of string $s = s_1 s_2 \ldots s_{|s|}$ (where $|s|$ is the length of string s) is a string $x = s_{k_1} s_{k_2} \ldots s_{k_{|x|}}$ $(1 \le k_1 < k_2 < \ldots < k_{|x|} \le |s|)$.

You've got two strings — s and t. Let's consider all subsequences of string s, coinciding with string t. Is it true that each character of string s occurs in at least one of these subsequences? In other words, is it true that for all i $(1 \le i \le |s|)$, there is such subsequence $x = s_{k_1} s_{k_2} \ldots s_{k_{|x|}}$ of string s, that $x = t$ and for some j $(1 \le j \le |x|)$ $k_j = i$.

Input

The first line contains string s, the second line contains string t. Each line consists only of lowercase English letters. The given strings are non-empty, the length of each string does not exceed $2 \cdot 10^5$.

Output

Print "Yes" (without the quotes), if each character of the string s occurs in at least one of the described subsequences, or "No" (without the quotes) otherwise.

Examples

input
abab ab
output
Yes

input
abacaba aba
output
No

input
abc ba
output
No

Note

In the first sample string t can occur in the string s as a subsequence in three ways: **ab**ab, **a**ba**b** and ab**ab**. In these occurrences each character of string s occurs at least once.

In the second sample the 4-th character of the string s doesn't occur in any occurrence of string t.

In the third sample there is no occurrence of string t in string s.

E. Partial Sums

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You've got an array a, consisting of n integers. The array elements are indexed from 1 to n. Let's determine a two step operation like that:

1. First we build by the array a an array s of partial sums, consisting of n elements. Element number i ($1 \le i \le n$) of array s equals $s_i = \left(\sum_{j=1}^{i} a_j\right) \bmod (10^9 + 7)$. The operation $x \bmod y$ means that we take the remainder of the division of number x by number y.
2. Then we write the contents of the array s to the array a. Element number i ($1 \le i \le n$) of the array s becomes the i-th element of the array a ($a_i = s_i$).

You task is to find array a after exactly k described operations are applied.

Input

The first line contains two space-separated integers n and k ($1 \le n \le 2000$, $0 \le k \le 10^9$). The next line contains n space-separated integers $a_1, a_2, \ldots, a_n$ — elements of the array a ($0 \le a_i \le 10^9$).

Output

Print n integers — elements of the array a after the operations are applied to it. Print the elements in the order of increasing of their indexes in the array a. Separate the printed numbers by spaces.

Examples

input
3 1 1 2 3
output
1 3 6

input
5 0 3 14 15 92 6
output
3 14 15 92 6

Codeforces (c) Copyright 2010-2016 Mike Mirzayanov
The only programming contests Web 2.0 platform