Codeforces Round #136 (Div. 1)

A. Little Elephant and Problem

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

The Little Elephant has got a problem — somebody has been touching his sorted by non-decreasing array a of length n and possibly swapped some elements of the array.

The Little Elephant doesn't want to call the police until he understands if he could have accidentally changed the array himself. He thinks that he could have accidentally changed array a, only if array a can be sorted in no more than one operation of swapping elements (not necessarily adjacent). That is, the Little Elephant could have accidentally swapped some two elements.

Help the Little Elephant, determine if he could have accidentally changed the array a, sorted by non-decreasing, himself.

Input

The first line contains a single integer n ($2 \le n \le 10^5$) — the size of array a. The next line contains n positive integers, separated by single spaces and not exceeding 10^9, — array a.

Note that the elements of the array are not necessarily distinct numbers.

Output

In a single line print "YES" (without the quotes) if the Little Elephant could have accidentally changed the array himself, and "NO" (without the quotes) otherwise.

Examples

input
2 1 2
output
YES

input
3 3 2 1
output
YES

input
4 4 3 2 1
output
NO

Note

In the first sample the array has already been sorted, so to sort it, we need 0 swap operations, that is not more than 1. Thus, the answer is "YES".

In the second sample we can sort the array if we swap elements 1 and 3, so we need 1 swap operation to sort the array. Thus, the answer is "YES".

In the third sample we can't sort the array in more than one swap operation, so the answer is "NO".

B. Little Elephant and Array

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

The Little Elephant loves playing with arrays. He has array a, consisting of n positive integers, indexed from 1 to n. Let's denote the number with index i as a_i.

Additionally the Little Elephant has m queries to the array, each query is characterised by a pair of integers l_j and r_j $(1 \le l_j \le r_j \le n)$. For each query l_j, r_j the Little Elephant has to count, how many numbers x exist, such that number x occurs exactly x times among numbers $a_{l_j}, a_{l_j+1}, \ldots, a_{r_j}$.

Help the Little Elephant to count the answers to all queries.

Input

The first line contains two space-separated integers n and m $(1 \le n, m \le 10^5)$ — the size of array a and the number of queries to it. The next line contains n space-separated positive integers $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le 10^9)$. Next m lines contain descriptions of queries, one per line. The j-th of these lines contains the description of the j-th query as two space-separated integers l_j and r_j $(1 \le l_j \le r_j \le n)$.

Output

In m lines print m integers — the answers to the queries. The j-th line should contain the answer to the j-th query.

Examples

input
7 2 3 1 2 2 3 3 7 1 7 3 4

output
3 1

C. Little Elephant and Shifts

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

The Little Elephant has two permutations a and b of length n, consisting of numbers from 1 to n, inclusive. Let's denote the i-th $(1 \le i \le n)$ element of the permutation a as a_i, the j-th $(1 \le j \le n)$ element of the permutation b — as b_j.

The *distance* between permutations a and b is the minimum absolute value of the difference between the positions of the occurrences of some number in a and in b. More formally, it's such minimum $|i - j|$, that $a_i = b_j$.

A *cyclic shift* number i $(1 \le i \le n)$ of permutation b consisting from n elements is a permutation $b_i b_{i+1} \dots b_n b_1 b_2 \dots b_{i-1}$. Overall a permutation has n cyclic shifts.

The Little Elephant wonders, for all cyclic shifts of permutation b, what is the distance between the cyclic shift and permutation a?

Input

The first line contains a single integer n $(1 \le n \le 10^5)$ — the size of the permutations. The second line contains permutation a as n distinct numbers from 1 to n, inclusive. The numbers are separated with single spaces. The third line contains permutation b in the same format.

Output

In n lines print n integers — the answers for cyclic shifts. Print the answers to the shifts in the order of the shifts' numeration in permutation b, that is, first for the 1-st cyclic shift, then for the 2-nd, and so on.

Examples

input
2 1 2 2 1
output
1 0

input
4 2 1 3 4 3 4 2 1
output
2 1 0 1

D. Little Elephant and Triangle

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The Little Elephant is playing with the Cartesian coordinates' system. Most of all he likes playing with integer points. The Little Elephant defines an integer point as a pair of integers $(x; y)$, such that $0 \le x \le w$ and $0 \le y \le h$. Thus, the Little Elephant knows only $(w + 1)\cdot(h + 1)$ distinct integer points.

The Little Elephant wants to paint a triangle with vertexes at integer points, the triangle's area must be a positive integer. For that, he needs to find the number of groups of three points that form such triangle. At that, the order of points in a group matters, that is, the group of three points $(0;0)$, $(0;2)$, $(2;2)$ isn't equal to the group $(0;2)$, $(0;0)$, $(2;2)$.

Help the Little Elephant to find the number of groups of three integer points that form a nondegenerate triangle with integer area.

Input

A single line contains two integers w and h $(1 \le w, h \le 4000)$.

Output

In a single output line print an integer — the remainder of dividing the answer to the problem by 1000000007 $(10^9 + 7)$.

Examples

input
2 1
output
36

input
2 2
output
240

E. Little Elephant and Inversions

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

The Little Elephant has array a, consisting of n positive integers, indexed from 1 to n. Let's denote the number with index i as a_i.

The Little Elephant wants to count, how many pairs of integers l and r are there, such that $1 \le l < r \le n$ and sequence $b = a_1 a_2 \ldots a_l a_r a_{r+1} \ldots a_n$ has no more than k inversions.

An *inversion* in sequence b is a pair of elements of the sequence b, that change their relative order after a stable sorting of the sequence. In other words, an inversion is a pair of integers i and j, such that $1 \le i < j \le |b|$ and $b_i > b_j$, where $|b|$ is the length of sequence b, and b_j is its j-th element.

Help the Little Elephant and count the number of the described pairs.

Input

The first line contains two integers n and k ($2 \le n \le 10^5$, $0 \le k \le 10^{18}$) — the size of array a and the maximum allowed number of inversions respectively. The next line contains n positive integers, separated by single spaces, $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) — elements of array a.

Please, do not use the %lld specifier to read or write 64-bit integers in C++. It is preferred to use cin, cout streams or the %I64d specifier.

Output

In a single line print a single number — the answer to the problem.

Examples

input
3 1 1 3 2
output
3

input
5 2 1 3 2 1 7
output
6