A. Pots

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

In a cooking pot shop sales decreased dramatically. Marketing managers of the shop did a research and found out that the reason was frying pans. People stopped buying pots as pans are both cheaper and more compact during storage. The Board of Directors made a decision to extent assortment and start selling also pans. The first batch is already ordered.

Warehouse logistics department was given a task to find a place for the new goods. Now there are N pots in the warehouse. Every pot has a diameter D_i. There is the only way to save space — it is possible to embed into any pot another one of a smaller diameter, into which in turn other can be embedded.

Help the logistics specialist to find a minimal number of pots in the warehouse in which all other pots can be embedded.

Input

The first line contains a single number N ($1 \le N \le 1000$). The second line contains N integers D_i separated by spaces ($1 \le D_i \le 10\,000$).

Output

Output the obtained number.

Examples

input	output
5 7 5 2 5 2	2

B. Row and Column Swaps

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

Given an $n \times n$ matrix with integer entries, support the following three operations.

- `R a b`: swap all entries in row a and row b;
- `C a b`: swap all entries in column a and column b;
- `A a b`: query the entry in row a and column b.

Input

The first line consists of two integers n and k, where n is the size of the matrix and k is the total number of operations. Each of the following n lines contains n integers, which are entries in the in the initial matrix. Each of the following k lines contains an operation, as stated above.

$1 \leq n \leq 1000$, $1 \leq k \leq 500000$, all entries in the matrix are in $[0, 10^9]$.

Output

For each query, output a line which is the answer.

Example

input	output
3 5 1 2 3 4 5 6 7 8 9 A 3 2 R 3 2 C 2 3 A 2 2 A 3 2	8 9 6

C. Maximizing the Bitwise AND

time limit per test: 1 second
memory limit per test: 256 megabytes

Given an array with n non-negative integers $a_1, a_2, \ldots, a_n$, choose $1 \leq i < j \leq n$ to maximize the bitwise AND of a_i and a_j. Note that choosing i and j such that $a_i = a_j$ is valid, but we must have $i \neq j$.

Input

The first line contains a single integer n. $2 \leq n \leq 10^5$. The second line contains n integers which are $a_1, a_2, \ldots, a_n$, with $0 \leq a_i \leq 10^9$.

Output

A single line with a single integer which is the maximum bitwise value we can achieve.

Examples

input	output
3 1 2 1	1

input	output
5 1024 1023 1022 1021 1020	1022

Note

See https://en.wikipedia.org/wiki/Bitwise_operation#AND for a definition of bitwise AND.

D. Uranium Cubes

time limit per test: 1 second
memory limit per test: 256 megabytes

There are $1 \le n \le 10^5$ small cubes of uranium located on a line. The coordinate of the i-th cube is a_i. Several cubes can be located at the same place on the line. It is guaranteed that $1 \le a_i \le L$, where $L \le 10^9$ is specified. There's also a specified bound $0 \le B \le 10^{15}$, whose meaning is explained below.

A subset S of the cubes, and a location p along the line are selected. The selected cubes are moved to the location p. The cost of doing these moves is the sum over all the cubes in S of the distance that they move. The move cost must be at most B.

Your goal is to compute the maximum size of the set S so that this is possible. Obviously you're trying to figure out if you can get a critical mass of uranium together within budget, because you want to blow up the world. Duh.

Input

The first line contains three integers n, L and B. Each of the following n lines contains a single integer a_i.

Output

An integer, which is the maximum number of cubes we can get together within budget.

Examples

input	output
5 20 6 1 2 10 12 14	3

input	output
3 3 2 3 1 2	3

Note

In the first example, set $p = 11$ and move cubes with coordinates 10, 12 and 14. Doing so costs $1 + 1 + 3 = 5$ which is within our budget of 6.

In the second example, set $p = 2$ and collect all three cubes there for a cost of 2.

E. How Big is that Set?

time limit per test: 1 second
memory limit per test: 256 megabytes

You're given a number $N > 0$ with no leading zeros and at most 50 digits. Consider the set of numbers $S(N)$ defined as follows: A number $Z \in S(N)$ if and only if you can construct Z using the following procedure: (1) Start with N. (2) Delete all the zero digits. (3) Permute the remaining digits arbitrarily. (4) Insert any number of zeros arbitrarily into this string of digits, except at the beginning.

Your goal is to compute the number of elements of $S(N)$ that are less than N, that is compute $|S(N) \bigcap \{1, 2, \ldots, N-1\}|$ See the example below for clarification.

Input
A single line containing N.

Output
A single line with the answer. It's guaranteed that the answer is at most $2^{63} - 1$.

Examples

input	output
1020	7

input	output
20120	21

Note
In the first example, the set $S(1020) = \{12, 21, 102, 120, 201, 210, 1002, 1020, \ldots\}$, where the elements are sorted in increasing order. Thus the answer is 7.

F. Number of Components

time limit per test: 8 seconds

memory limit per test: 256 megabytes

Suppose that we have an array of n distinct numbers $a_1, a_2, \ldots, a_n$. Let's build a graph on n vertices as follows: for every pair of vertices $i < j$ let's connect i and j with an edge, if $a_i < a_j$. Let's define **weight** of the array to be the number of connected components in this graph. For example, weight of array $[1, 4, 2]$ is 1, weight of array $[5, 4, 3]$ is 3.

You have to perform q queries of the following form — change the value at some position of the array. After each operation, output the weight of the array. Updates are not independent (the change stays for the future).

Input

The first line contains two integers n and q ($1 \le n, q \le 5 \cdot 10^5$) — the size of the array and the number of queries.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^6$) — the initial array.

Each of the next q lines contains two integers pos and x ($1 \le pos \le n$, $1 \le x \le 10^6$, $x \neq a_{pos}$). It means that you have to make $a_{pos} = x$.

It's guaranteed that at every moment of time, all elements of the array are different.

Output

After each query, output the weight of the array.

Example

input	output
5 3 50 40 30 20 10 1 25 3 45 1 48	3 3 4

Note

After the first query array looks like $[25, 40, 30, 20, 10]$, the weight is equal to 3.

After the second query array looks like $[25, 40, 45, 20, 10]$, the weight is still equal to 3.

After the third query array looks like $[48, 40, 45, 20, 10]$, the weight is equal to 4.