Problem A. Funky Numbers

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

As you very well know, this year's funkiest numbers are so called triangular numbers (that is, integers that are representable as $\frac{k(k+1)}{2}$, where k is some positive integer), and the coolest numbers are those that are representable as a sum of two triangular numbers.

A well-known hipster Andrew adores everything funky and cool but unfortunately, he isn't good at maths. Given number n, help him define whether this number can be represented by a sum of two triangular numbers (not necessarily different)!

Input

The first input line contains an integer n ($1 \leqslant n \leqslant 10^9$).

Output

Print "YES" (without the quotes), if n can be represented as a sum of two triangular numbers, otherwise print "NO" (without the quotes).

Examples

standard input	standard output
256	YES
512	NO

Note

In the first sample number $256 = \frac{2 \cdot 3}{2} + \frac{22 \cdot 23}{2}$.

In the second sample number 512 can not be represented as a sum of two triangular numbers.

Problem B. The Sum

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

You are given an array of n elements. You have to answer some queries of the array: change an element and report the sum of a subarray.

Input

The first line of input contains two numbers n and k ($1 \le n, k \le 100\,000$), the number of elements and the number of queries. Each of the next k lines contain queries of one of two kinds:

1. A i x — set i-th element of the array to x ($1 \le i \le n$, $0 \le x \le 10^9$)
2. Q l r — find the sum of the array elements between l and r, inclusive. ($1 \le l \le r \le n$)

Initially the array is filled with zeroes.

Output

For each Q-query print a single number: the corresponding sum.

Examples

standard input	standard output
5 9 A 2 2 A 3 1 A 4 2 Q 1 1 Q 2 2 Q 3 3 Q 4 4 Q 5 5 Q 1 5	0 2 1 2 0 5

Problem C. Dot Product

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	256 mebibytes

You have two arrays A and B of the same length. You must process three kinds of queries.

- "`*` l r x": add an integer x to all A_i where $l \leq i \leq r$.
- "`.` l r x": add an integer x to all B_i where $l \leq i \leq r$.
- "`?` l r": calculate the sum $A_l \cdot B_l + \ldots + A_r \cdot B_r$.

Arrays are 1-indexed. Initially, both arrays are filled with zeroes.

Input

The first line of input contains two space-separated integers n and m: the arrays' lengths and the number of queries, respectively ($1 \leq n, m \leq 100\,000$).

The next m lines contain queries in the format described above. In each query, $1 \leq l \leq r \leq n$ and $1 \leq x < 10^9 + 7$.

Output

For each query of the third type, print its result modulo $10^9 + 7$ on a separate line.

Examples

standard input	standard output
5 4 * 1 4 10 . 2 5 8 ? 1 3 ? 2 5	160 240

Problem D. Sum again...

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Implement a data structure which contains a set S of integers, with which you can do the following operations:

- $add(i)$ — add a number i into the set S (if it doesn't contain it);
- $sum(l, r)$ — print a sum of all elements x in S, which satisfy the inequality $l \le x \le r$.

Input

Initially set S is empty. First line of the input contains one integer n — the number of queries ($1 \le n \le 300\,000$). The following n lines contain queries. Each query has a type «+ i», or «? l r». Query of type «? l r» is a query $sum(l, r)$.

If query «+ i» goes at the beginning or after other query «+», then it's a query $add(i)$. If it goes after query «?», and the result of it was y, then it's a query $add((i + y) \bmod 10^9)$.

In each query arguments are in the range from 0 to 10^9.

Output

For each query of type «? l r» print a single integer — the answer for this query.

Example

standard input	standard output
6 + 1 + 3 + 3 ? 2 4 + 1 ? 2 4	3 7

Problem E. K-th maximum

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	64 mebibytes

Implement a data structure which can insert elements, erase elements and find the k-th maximum.

Input

The first line of input contains a single number n, the number of queries. Each of the next n lines contain one query each. Each query is described by two numbers c_i and k_i ($|k_i| \le 10^9$), the type and the argument of the query, respectively. You should support the following commands:

- $+1$ (or 1): insert an element with value k_i.
- 0: find and print k_i-th maximum.
- -1: remove an element with value k_i.

It is guaranteed that the data structure will never contain two elements with the same value. Also it is guaranteed that there is at least k elements in the data structure when the k-th maximum is requested.

Output

For each command of type 0 print a single number: the k_i-th maximum.

Example

standard input	standard output
11 +1 5 +1 3 +1 7 0 1 0 2 0 3 -1 5 +1 10 0 1 0 2 0 3	7 5 3 10 7 3

Problem F. Permutations

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	64 mebibytes

Vasya has written down all numbers from 1 to n in some order, writing each number exactly once. There are too many numbers, so he cannot look at all of them at once. However, he still needs to understand this sequence, so he wrote a program which can answer the queries — how many numbers on positions from x to y, inclusive, have the value from k to l, inclusive.

You are asked to write the same program.

Input

The first line of input contains two numbers n, m ($1 \le n, m \le 100\,000$), the number of elements and the number of queries. The next line contains n distinct numbers from 1 to n, the numbers written by Vasya. Next m lines contain queries. Each of them contains four numbers x, y, k, l ($1 \le x \le y \le n$, $1 \le k \le l \le n$).

Output

Print m lines, each of them containing the answer to Vasya's query.

Example

standard input	standard output

4 2
1 2 3 4
1 2 2 3
1 3 1 3
	1
	3

Problem G. Permutations strike back

Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	256 mebibytes

Vasya has written down all numbers from 1 to n in some order, writing each number exactly once. Sometimes he erases some number and writes another one at its position. There are too many numbers, so Vasya cannot look at all of them at once. However, he still needs to understand this sequence, so he wrote a program which can answer the queries — how many numbers on positions from x to y, inclusive, have the value from k to l, inclusive.

You are asked to write the same program.

Input

The first line of input contains two numbers n, m ($1 \le n, m \le 100\,000$), the number of elements and the number of queries. The next line contains n distinct numbers from 1 to n, the numbers written by Vasya. Next m lines contain queries.

If Vasya wants to change the number at some position, the query starts with the word `SET` and looks like `SET a b` ($1 \le a, b \le n$). That means that Vasya changed the number on position a to b.

If Vasya wants to ask his data structure, the query starts with the word `GET` and looks like `SET x y k l` ($1 \le x \le y \le n$, $1 \le k \le l \le n$).

Output

Print a single number for each Vasya's query — the answer to his problem.

Example

standard input	standard output
4 4 1 2 3 4 GET 1 2 2 3 GET 1 3 1 3 SET 1 4 GET 1 3 1 3	1 3 2

Problem H. Important Guests

Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	512 megabytes

You are probably familiar with the famous dancing-master, professor Padegras, who is now preparing students of Mvodsk State University for their graduation ball. Tomorrow, the head of the University has important foreign guests who asked to show them all aspects of the students' life. That is why Padegras was asked to choose a small subset of his students for a promo-dance.

Professor has N boys and M girls in his class, and he knows all K mutual sympathy relations between them. For the planned dance, Padegras needs to choose exactly two boys and exactly two girls, and since the chosen dance is known to be very energetic and passionate, only the pairs with mutual sympathy are able to participate. Moreover, the pairs are going to mix many times during the dance, so all four possible mutual sympathy relations between each of the chosen boys and each of the chosen girls are required.

One more problem is that this event will occur tomorrow, meaning Padegras cannot be sure who of his students will be able to come. Cheer him up and calculate how many different subsets of exactly two boys and exactly two girls meet all the given requirements.

Input

The first line of input contains three integer numbers N, M and K: the number of boys, the number of girls and the number of mutual sympathy relations ($1 \le N, M \le 100\,000$, $1 \le K \le 200\,000$).

Each of next K lines describe one mutual sympathy relation by two integer numbers v_i ($1 \le v_i \le N$) and u_i ($1 \le u_i \le M$), denoting the boy and the girl in this relation. It is guaranteed that each possible pair occurs at most once.

Output

Print one number: the answer to the problem.

Examples

standard input	standard output

3 4 9
1 1
1 2
1 4
2 1
2 3
3 1
3 2
3 3
3 4
	4

Problem I. Combinations Strike Back

Input file:	*standard input*
Output file:	*standard output*
Time limit:	7 seconds
Memory limit:	512 megabytes

Given a set containing n *different* elements, one can easily find the number of distinct k-element subsets of it. This value is the well-known binomial coefficient $\binom{n}{k} = \frac{n!}{k!\cdot(n-k)!}$.

However, this problem is a bit harder. You are a given a multiset with n (possibly equal) elements and you would like to answer several queries of the form "if we insert an additional element x into the multiset, how many distinct k-element submultisets it will have?". Note that, after each query, the given multiset remains **unchanged**.

Recall that a multiset is a generalization of the notion of set in which elements are allowed to appear more than once. A k-element submultiset of a multiset is obtained by selecting exactly k elements of the multiset (it is allowed to select equal elements). Two multisets are considered distinct if there is an element which occurs in one of them more times than in the other.

As the answers may be very large, you need to find each of them modulo 1 051 721 729.

Input

The first line of input contains an integer number n, the size of the multiset ($1 \le n \le 120\,000$). The next line contains n integer numbers a_i separated by spaces: the elements of the multiset ($1 \le a_i \le n$). Note that the elements are not necessarily pairwise distinct. The following line contains an integer number q, the number of queries ($1 \le q \le 120\,000$). The next q lines describe queries, each description consists of two integer numbers x and k ($1 \le x \le n$, $1 \le k \le n+1$).

Output

For each query, output the required number of different submultisets, taken modulo 1 051 721 729.

Examples

standard input	standard output

6
1 2 2 3 3 3
4
1 2
2 3
3 4
4 5
	6
	7
	6
	8

Note

For the first query (if we add element `1`), the multiset will look like `1 2 2 3 3 3 1`, so it will have six different two-element submultisets: `(1 1)`, `(1 2)`, `(1 3)`, `(2 2)`, `(2 3)` and `(3 3)`.

Problem J. Lunch Queue

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 megabytes

It is lunch time! That's why n employees are coming to the canteen to have lunch. It is known that i-th employee works in the team c_i and has impudence a_i.

Suppose there are currently l people in the queue. Positions in the queue are numbered from 1 to l from the beginning of the queue. When a person comes to the canteen, he tries to get as close to the beginning of the queue as possible by staying next to one of his teammates. Formally, if a newcomer gets into the queue at the position k ($1 \le k \le l+1$), he shifts the k-th person and everybody behind him one position back and gets the position k. In particular, $k = 1$ corresponds to the beginning of the queue and $k = l+1$ corresponds to the end of the queue.

An employee i can stay at the position k only if two conditions are held:

- After getting to the position k, at least one of the neighbors of newcomer is from the same team c_i;
- A newcomer is not shifting too much people for his level of impudence, namely: $k \ge l + 1 - a_i$.

Among all k satisfying the conditions above the minimum possible is chosen. If there are no suitable values of k, a newcomer simply takes place after the last person in the queue.

For example, if there are five people in the queue working in teams 1, 3, 4, 1, 3 respectively (starting from the beginning of the queue) and the newcomer works in team 1 and has impudence 3, he stays at the position 4 of the queue. After that, the sequence of the employee teams becomes 1, 3, 4, **1**, 1, 3. Note that impudence value of 3 also allowed him to get to the position 3, but there would be no teammates next to him in that case, so the first condition is violated.

Knowing the order in which employees arrive and their values of c_i and a_i, find out how the queue will look like in the end.

Input

The first line contains an integer n ($1 \le n \le 400\,000$), the number of employees.

Each of the next n lines contains two integers c_i, a_i ($1 \le c_i \le n$, $0 \le a_i \le 400\,000$), the team and the impudence of the i-th employee.

Output

Output n distinct integers from 1 to n which are the indices of people in the queue from beginning of the queue to the end after all employees get to the canteen. Employees are indexed from 1 to n in order of their appearance in the input data.

Example

standard input	standard output

8
1 0
2 1
2 2
1 1
3 2
1 3
2 3
2 5
	1 3 8 2 7 6 4 5