Codeforces Round #791 (Div. 2)

A. AvtoBus

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Spring has come, and the management of the AvtoBus bus fleet has given the order to replace winter tires with summer tires on all buses.

You own a small bus service business and you have just received an order to replace n tires. You know that the bus fleet owns two types of buses: with two axles (these buses have 4 wheels) and with three axles (these buses have 6 wheels).

You don't know how many buses of which type the AvtoBus bus fleet owns, so you wonder how many buses the fleet might have. You have to determine the minimum and the maximum number of buses that can be in the fleet if you know that the total number of wheels for all buses is n.

Input

The first line contains an integer t ($1 \le t \le 1\,000$) — the number of test cases. The following lines contain description of test cases.

The only line of each test case contains one integer n ($1 \le n \le 10^{18}$) — the total number of wheels for all buses.

Output

For each test case print the answer in a single line using the following format.

Print two integers x and y ($1 \le x \le y$) — the minimum and the maximum possible number of buses that can be in the bus fleet.

If there is no suitable number of buses for the given n, print the number -1 as the answer.

Example

input
4 4 7 24 998244353998244352
output
1 1 -1 4 6 166374058999707392 249561088499561088

Note

In the first test case the total number of wheels is 4. It means that there is the only one bus with two axles in the bus fleet.

In the second test case it's easy to show that there is no suitable number of buses with 7 wheels in total.

In the third test case the total number of wheels is 24. The following options are possible:

- Four buses with three axles.
- Three buses with two axles and two buses with three axles.
- Six buses with two axles.

So the minimum number of buses is 4 and the maximum number of buses is 6.

B. Stone Age Problem

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

Once upon a time Mike and Mike decided to come up with an outstanding problem for some stage of ROI (rare olympiad in informatics). One of them came up with a problem prototype but another stole the idea and proposed that problem for another stage of the same olympiad. Since then the first Mike has been waiting for an opportunity to propose the original idea for some other contest... Mike waited until this moment!

You are given an array a of n integers. You are also given q queries of two types:

- Replace i-th element in the array with integer x.
- Replace each element in the array with integer x.

After performing each query you have to calculate the sum of all elements in the array.

Input
The first line contains two integers n and q ($1 \le n, q \le 2 \cdot 10^5$) — the number of elements in the array and the number of queries, respectively.

The second line contains n integers $a_1, \ldots, a_n$ ($1 \le a_i \le 10^9$) — elements of the array a.

Each of the following q lines contains a description of the corresponding query. Description begins with integer t ($t \in \{1, 2\}$) which denotes a type of the query:

- If $t = 1$, then two integers i and x are following ($1 \le i \le n$, $1 \le x \le 10^9$) — position of replaced element and it's new value.
- If $t = 2$, then integer x is following ($1 \le x \le 10^9$) — new value of each element in the array.

Output
Print q integers, each on a separate line. In the i-th line print the sum of all elements in the array after performing the first i queries.

Example

input
5 5 1 2 3 4 5 1 1 5 2 10 1 5 11 1 4 1 2 1
output
19 50 51 42 5

Note
Consider array from the example and the result of performing each query:

1. Initial array is $[1, 2, 3, 4, 5]$.
2. After performing the first query, array equals to $[5, 2, 3, 4, 5]$. The sum of all elements is 19.
3. After performing the second query, array equals to $[10, 10, 10, 10, 10]$. The sum of all elements is 50.
4. After performing the third query, array equals to $[10, 10, 10, 10, 11]$. The sum of all elements is 51.
5. After performing the fourth query, array equals to $[10, 10, 10, 1, 11]$. The sum of all elements is 42.
6. After performing the fifth query, array equals to $[1, 1, 1, 1, 1]$. The sum of all elements is 5.

C. Rooks Defenders

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You have a square chessboard of size $n \times n$. Rows are numbered from top to bottom with numbers from 1 to n, and columns — from left to right with numbers from 1 to n. So, each cell is denoted with pair of integers (x, y) ($1 \le x, y \le n$), where x is a row number and y is a column number.

You have to perform q queries of three types:

- Put a new rook in cell (x, y).
- Remove a rook from cell (x, y). It's guaranteed that the rook was put in this cell before.
- Check if each cell of *subrectangle* $(x_1, y_1) - (x_2, y_2)$ of the board is attacked by at least one rook.

Subrectangle is a set of cells (x, y) such that for each cell two conditions are satisfied: $x_1 \le x \le x_2$ and $y_1 \le y \le y_2$.

Recall that cell (a, b) is attacked by a rook placed in cell (c, d) if either $a = c$ or $b = d$. In particular, the cell containing a rook is attacked by this rook.

Input
The first line contains two integers n and q ($1 \le n \le 10^5$, $1 \le q \le 2 \cdot 10^5$) — the size of the chessboard and the number of queries, respectively.

Each of the following q lines contains description of a query. Description begins with integer t ($t \in \{1, 2, 3\}$) which denotes type of a query:

- If $t = 1$, two integers x and y follows ($1 \le x, y \le n$) — coordinated of the cell where the new rook should be put in. It's

guaranteed that there is no rook in the cell (x, y) at the moment of the given query.

- If $t = 2$, two integers x and y follows ($1 \le x, y \le n$) — coordinates of the cell to remove a rook from. It's guaranteed that there is a rook in the cell (x, y) at the moment of the given query.
- If $t = 3$, four integers x_1, y_1, x_2 and y_2 follows ($1 \le x_1 \le x_2 \le n$, $1 \le y_1 \le y_2 \le n$) — subrectangle to check if each cell of it is attacked by at least one rook.

It's guaranteed that among q queries there is at least one query of the third type.

Output

Print the answer for each query of the third type in a separate line. Print "Yes" (without quotes) if each cell of the subrectangle is attacked by at least one rook.

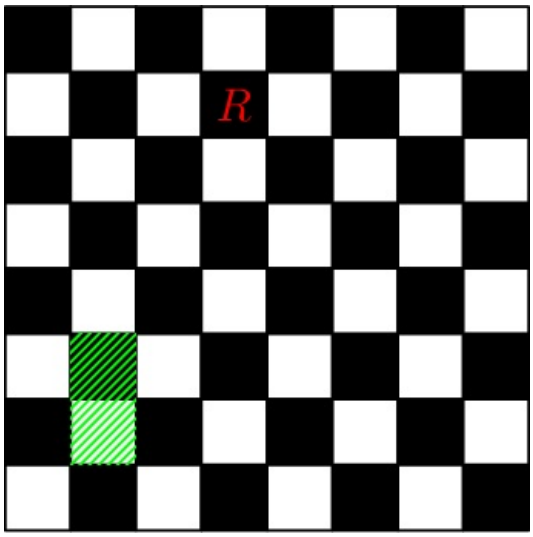
Otherwise print "No" (without quotes).

Example

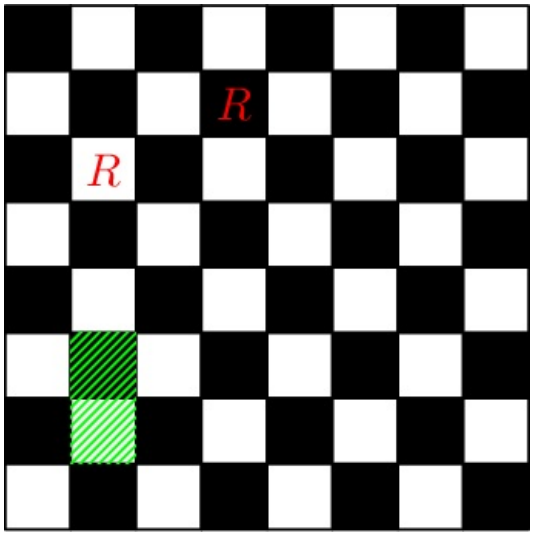
input
8 10 1 2 4 3 6 2 7 2 1 3 2 3 6 2 7 2 1 4 3 3 2 6 4 8 2 4 3 3 2 6 4 8 1 4 8 3 2 6 4 8
output
No Yes Yes No Yes

Note

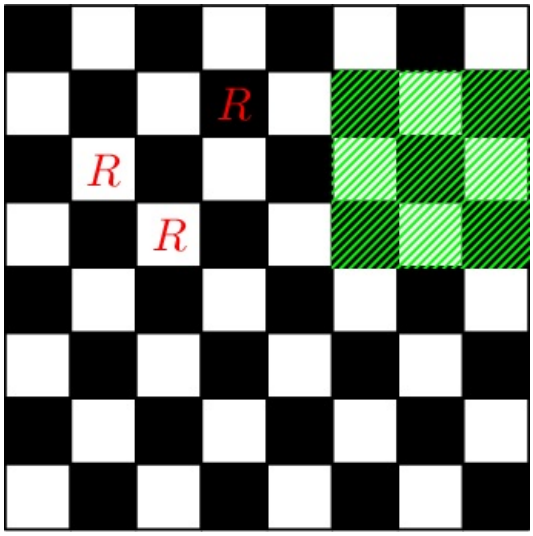
Consider example. After the first two queries the board will look like the following picture (the letter R denotes cells in which rooks are located, the subrectangle of the query of the third type is highlighted in green):



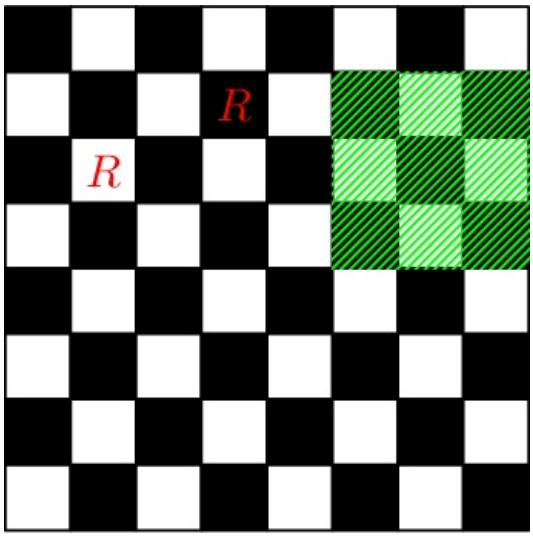
Chessboard after performing the third and the fourth queries:



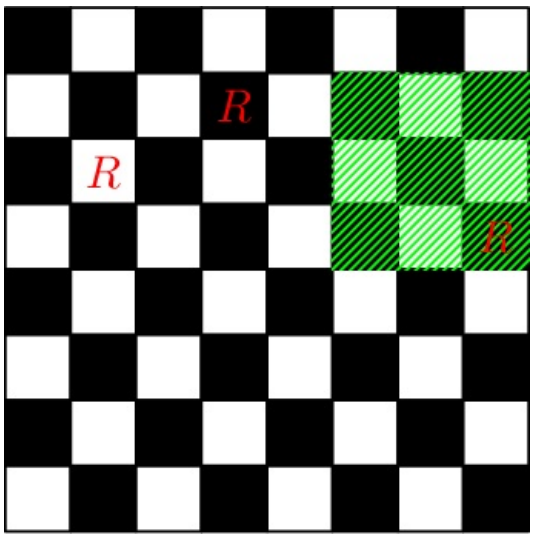
Chessboard after performing the fifth and the sixth queries:



Chessboard after performing the seventh and the eighth queries:



Chessboard after performing the last two queries:



D. Toss a Coin to Your Graph...

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

One day Masha was walking in the park and found a graph under a tree... Surprised? Did you think that this problem would have some logical and reasoned story? No way! So, the problem...

Masha has an oriented graph which i-th vertex contains some positive integer a_i. Initially Masha can put a coin at some vertex. In one operation she can move a coin placed in some vertex u to any other vertex v such that there is an oriented edge $u \to v$ in the graph. Each time when the coin is placed in some vertex i, Masha write down an integer a_i in her notebook (in particular, when Masha initially puts a coin at some vertex, she writes an integer written at this vertex in her notebook). Masha wants to make exactly $k - 1$ operations in such way that the maximum number written in her notebook is as small as possible.

Input

The first line contains three integers n, m and k ($1 \le n \le 2 \cdot 10^5$, $0 \le m \le 2 \cdot 10^5$, $1 \le k \le 10^{18}$) — the number of vertices and edges in the graph, and the number of operation that Masha should make.

The second line contains n integers a_i ($1 \le a_i \le 10^9$) — the numbers written in graph vertices.

Each of the following m lines contains two integers u and v ($1 \le u \ne v \le n$) — it means that there is an edge $u \to v$ in the graph.

It's guaranteed that graph doesn't contain loops and multi-edges.

Output

Print one integer — the minimum value of the maximum number that Masha wrote in her notebook during optimal coin movements.

If Masha won't be able to perform $k - 1$ operations, print -1.

Examples

input
6 7 4 1 10 2 3 4 5 1 2 1 3 3 4 4 5 5 6 6 2 2 5
output
4

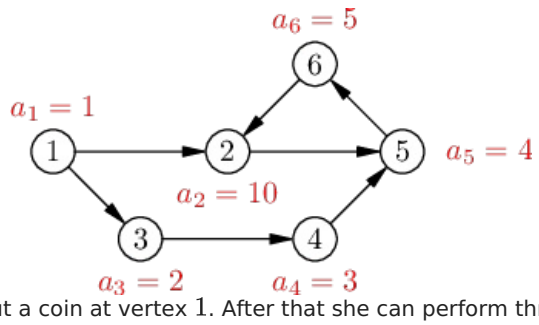
input
6 7 100 1 10 2 3 4 5 1 2 1 3 3 4 4 5 5 6 6 2 2 5
output
10

input
2 1 5 1 1 1 2
output
-1

input
1 0 1 1000000000
output
1000000000

Note

Graph described in the first and the second examples is illustrated below.



In the first example Masha can initially put a coin at vertex 1. After that she can perform three operations: $1 \to 3$, $3 \to 4$ and $4 \to 5$. Integers $1, 2, 3$ and 4 will be written in the notepad.

In the second example Masha can initially put a coin at vertex 2. After that she can perform 99 operations: $2 \to 5$, $5 \to 6$, $6 \to 2$, $2 \to 5$, and so on. Integers $10, 4, 5, 10, 4, 5, \ldots, 10, 4, 5, 10$ will be written in the notepad.

In the third example Masha won't be able to perform 4 operations.

E. Typical Party in Dorm

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

Today is a holiday in the residence hall — Oleh arrived, in honor of which the girls gave him a string. Oleh liked the gift a lot, so he immediately thought up and offered you, his best friend, the following problem.

You are given a string s of length n, which consists of the first 17 lowercase Latin letters $\{a, b, c, \ldots, p, q\}$ and question marks. And q queries. Each query is defined by a set of pairwise distinct lowercase first 17 letters of the Latin alphabet, which can be used to replace the question marks in the string s.

The answer to the query is the sum of the number of distinct substrings that are palindromes over all strings that can be obtained from the original string s by replacing question marks with available characters. The answer must be calculated modulo $998\,244\,353$.

Pay attention! Two substrings are different when their start and end positions in the string are different. So, the number of different substrings that are palindromes for the string aba will be 4: a, b, a, aba.

Consider examples of replacing question marks with letters. For example, from the string aba??ee when querying $\{a, b\}$ you can get the strings ababaee or abaaaee but you cannot get the strings pizza, abaee, abacaba, aba?fee, aba47ee, or abatree.

Recall that a palindrome is a string that reads the same from left to right as from right to left.

Input

The first line contains a single integer n ($1 \le n \le 1\,000$) — the length of the string s.

The second line contains the string s, which consists of n lowercase Latin letters and question marks. It is guaranteed that all letters in the string belong to the set $\{a, b, c, \ldots, p, q\}$.

The third line contains a single integer q ($1 \le q \le 2 \cdot 10^5$) — the number of queries.

This is followed by q lines, each containing a single line t — a set of characters that can replace question marks ($1 \le |t| \le 17$). It is guaranteed that all letters in the string belong to the set $\{a, b, c, \ldots, p, q\}$ and occur at most once.

Output

For each query print one number — the total numbers of palindromic substrings in all strings that can be obtained from the string s, modulo $998\,244\,353$.

Examples

input
7 ab??aba 8 a b ab abc abcd abcde abcdef abcdefg
output
14 13 55 105 171 253 351 465

input
11 ??????????? 6 abcdefghijklmnopq ecpnkhbmlidgfjao olehfan codef glhf q
output
900057460 712815817 839861037 756843750 70840320 66

Note

Consider the first example and the first query in it. We can get only one string as a result of replacing the question marks — abaaaba. It has the following palindrome substrings:

1. a — substring $[1; 1]$.
2. b — substring $[2; 2]$.
3. a — substring $[3; 3]$.
4. a — substring $[4; 4]$.
5. a — substring $[5; 5]$.
6. b — substring $[6; 6]$.
7. a — substring $[7; 7]$.
8. aa — substring $[3; 4]$.
9. aa — substring $[4; 5]$.
10. aba — substring $[1; 3]$.
11. aaa — substring $[3; 5]$.
12. aba — substring $[5; 7]$.
13. baaab — substring $[2; 6]$.
14. abaaaba — substring $[1; 7]$.

In the third request, we get 4 lines: abaaaba, abababa, abbaaba, abbbaba.

F. Formalism for Formalism

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

Yura is a mathematician, and his cognition of the world is so absolute as if he have been solving formal problems a hundred of trillions of billions of years. This problem is just that!

Consider all non-negative integers from the interval $[0, 10^n)$. For convenience we complement all numbers with leading zeros in such way that each number from the given interval consists of exactly n decimal digits.

You are given a set of pairs (u_i, v_i), where u_i and v_i are distinct decimal digits from 0 to 9.

Consider a number x consisting of n digits. We will enumerate all digits from left to right and denote them as $d_1, d_2, \ldots, d_n$. In one operation you can swap digits d_i and d_{i+1} if and only if there is a pair (u_j, v_j) in the set such that at least one of the following conditions is satisfied:

1. $d_i = u_j$ and $d_{i+1} = v_j$,
2. $d_i = v_j$ and $d_{i+1} = u_j$.

We will call the numbers x and y, consisting of n digits, *equivalent* if the number x can be transformed into the number y using some number of operations described above. In particular, every number is considered equivalent to itself.

You are given an integer n and a set of m pairs of digits (u_i, v_i). You have to find the maximum integer k such that there exists a set of integers $x_1, x_2, \ldots, x_k$ ($0 \le x_i < 10^n$) such that for each $1 \le i < j \le k$ the number x_i is **not** equivalent to the number x_j.

Input

The first line contains an integer n ($1 \le n \le 50\,000$) — the number of digits in considered numbers.

The second line contains an integer m ($0 \le m \le 45$) — the number of pairs of digits in the set.

Each of the following m lines contains two digits u_i and v_i, separated with a space ($0 \le u_i < v_i \le 9$).

It's guaranteed that all described pairs are pairwise distinct.

Output

Print one integer — the maximum value k such that there exists a set of integers $x_1, x_2, \ldots, x_k$ ($0 \le x_i < 10^n$) such that for each $1 \le i < j \le k$ the number x_i is **not** equivalent to the number x_j.

As the answer can be big enough, print the number k modulo $998\,244\,353$.

Examples

input
1 0
output
10

input
2

1
0 1

output

99

input

2
9
0 1
1 2
2 3
3 4
4 5
5 6
6 7
7 8
8 9

output

91

Note

In the first example we can construct a set that contains all integers from 0 to 9. It's easy to see that there are no two equivalent numbers in the set.

In the second example there exists a unique pair of equivalent numbers: 01 and 10. We can construct a set that contains all integers from 0 to 99 despite number 1.