Codeforces Round #366 (Div. 2)

A. Hulk

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

Dr. Bruce Banner hates his enemies (like others don't). As we all know, he can barely talk when he turns into the incredible Hulk. That's why he asked you to help him to express his feelings.

Hulk likes the Inception so much, and like that his feelings are complicated. They have n layers. The first layer is hate, second one is love, third one is hate and so on...

For example if $n = 1$, then his feeling is "`I hate it`" or if $n = 2$ it's "`I hate that I love it`", and if $n = 3$ it's "`I hate that I love that I hate it`" and so on.

Please help Dr. Banner.

Input
The only line of the input contains a single integer n $(1 \le n \le 100)$ — the number of layers of love and hate.

Output
Print Dr.Banner's feeling in one line.

Examples

input
1
output
I hate it

input
2
output
I hate that I love it

input
3
output
I hate that I love that I hate it

B. Spider Man

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

Peter Parker wants to play a game with Dr. Octopus. The game is about cycles. *Cycle* is a sequence of vertices, such that first one is connected with the second, second is connected with third and so on, while the last one is connected with the first one again. Cycle may consist of a single isolated vertex.

Initially there are k cycles, i-th of them consisting of exactly v_i vertices. Players play alternatively. Peter goes first. On each turn a player must choose a cycle with at least 2 vertices (for example, x vertices) among all available cycles and replace it by two cycles with p and $x - p$ vertices where $1 \le p < x$ is chosen by the player. The player who cannot make a move loses the game (and his life!).

Peter wants to test some configurations of initial cycle sets before he actually plays with Dr. Octopus. Initially he has an empty set. In the i-th test he adds a cycle with a_i vertices to the set (this is actually a multiset because it can contain two or more identical cycles). After each test, Peter wants to know that if the players begin the game with the current set of cycles, who wins?

Peter is pretty good at math, but now he asks you to help.

Input

The first line of the input contains a single integer n ($1 \le n \le 100\,000$) — the number of tests Peter is about to make.

The second line contains n space separated integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 10^9$), i-th of them stands for the number of vertices in the cycle added before the i-th test.

Output

Print the result of all tests in order they are performed. Print `1` if the player who moves first wins or `2` otherwise.

Examples

input
3 1 2 3
output
2 1 1

input
5 1 1 5 1 1
output
2 2 2 2 2

Note

In the first sample test:

In Peter's first test, there's only one cycle with 1 vertex. First player cannot make a move and loses.

In his second test, there's one cycle with 1 vertex and one with 2. No one can make a move on the cycle with 1 vertex. First player can replace the second cycle with two cycles of 1 vertex and second player can't make any move and loses.

In his third test, cycles have 1, 2 and 3 vertices. Like last test, no one can make a move on the first cycle. First player can replace the third cycle with one cycle with size 1 and one with size 2. Now cycles have 1, 1, 2, 2 vertices. Second player's only move is to replace a cycle of size 2 with 2 cycles of size 1. And cycles are 1, 1, 1, 1, 2. First player replaces the last cycle with 2 cycles with size 1 and wins.

In the second sample test:

Having cycles of size 1 is like not having them (because no one can make a move on them).

In Peter's third test: There a cycle of size 5 (others don't matter). First player has two options: replace it with cycles of sizes 1 and 4 or 2 and 3.

- If he replaces it with cycles of sizes 1 and 4: Only second cycle matters. Second player will replace it with 2 cycles of sizes 2. First player's only option to replace one of them with two cycles of size 1. Second player does the same thing with the other cycle. First player can't make any move and loses.
- If he replaces it with cycles of sizes 2 and 3: Second player will replace the cycle of size 3 with two of sizes 1 and 2. Now only cycles with more than one vertex are two cycles of size 2. As shown in previous case, with 2 cycles of size 2 second player wins.

So, either way first player loses.

C. Thor

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

Thor is getting used to the Earth. As a gift Loki gave him a smartphone. There are n applications on this phone. Thor is fascinated by this phone. He has only one minor issue: he can't count the number of unread notifications generated by those applications (maybe Loki put a curse on it so he can't).

q events are about to happen (in chronological order). They are of three types:

1. Application x generates a notification (this new notification is unread).
2. Thor reads all notifications generated so far by application x (he may re-read some notifications).
3. Thor reads the first t notifications generated by phone applications (notifications generated in first t events of the first type). It's guaranteed that there were at least t events of the first type before this event. Please note that he doesn't read first t unread notifications, he just reads the very first t notifications generated on his phone and he may re-read some of them in this operation.

Please help Thor and tell him the number of unread notifications after each event. You may assume that initially there are no notifications in the phone.

Input

The first line of input contains two integers n and q ($1 \le n, q \le 300\,000$) — the number of applications and the number of events to happen.

The next q lines contain the events. The i-th of these lines starts with an integer $type_i$ — type of the i-th event. If $type_i = 1$ or $type_i = 2$ then it is followed by an integer x_i. Otherwise it is followed by an integer t_i ($1 \le type_i \le 3$, $1 \le x_i \le n$, $1 \le t_i \le q$).

Output

Print the number of unread notifications after each event.

Examples

input

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

output

```
1
2
3
2
```

input

```
4 6
1 2
1 4
1 2
3 3
1 3
1 3
```

output

```
1
2
3
0
1
2
```

Note

In the first sample:

1. Application 3 generates a notification (there is 1 unread notification).
2. Application 1 generates a notification (there are 2 unread notifications).
3. Application 2 generates a notification (there are 3 unread notifications).
4. Thor reads the notification generated by application 3, there are 2 unread notifications left.

In the second sample test:

1. Application 2 generates a notification (there is 1 unread notification).
2. Application 4 generates a notification (there are 2 unread notifications).
3. Application 2 generates a notification (there are 3 unread notifications).
4. Thor reads first three notifications and since there are only three of them so far, there will be no unread notification left.

5. Application 3 generates a notification (there is 1 unread notification).
6. Application 3 generates a notification (there are 2 unread notifications).

D. Ant Man

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

Scott Lang is at war with Darren Cross. There are n chairs in a hall where they are, numbered with $1, 2, ..., n$ from left to right. The i-th chair is located at coordinate x_i. Scott is on chair number s and Cross is on chair number e. Scott can jump to all other chairs (not only neighboring chairs). He wants to start at his position (chair number s), visit each chair **exactly once** and end up on chair number e with Cross.

As we all know, Scott can shrink or grow big (grow big only to his normal size), so at any moment of time he can be either small or large (normal). The thing is, he can only shrink or grow big while being on a chair (not in the air while jumping to another chair). Jumping takes time, but shrinking and growing big takes no time. Jumping from chair number i to chair number j takes $|x_i - x_j|$ seconds. Also, jumping off a chair and landing on a chair takes extra amount of time.

If Scott wants to jump to a chair on his left, he can only be small, and if he wants to jump to a chair on his right he should be large.

Jumping off the i-th chair takes:

- c_i extra seconds if he's small.
- d_i extra seconds otherwise (he's large).

Also, landing on i-th chair takes:

- b_i extra seconds if he's small.
- a_i extra seconds otherwise (he's large).

In simpler words, jumping from i-th chair to j-th chair takes exactly:

- $|x_i - x_j| + c_i + b_j$ seconds if $j < i$.
- $|x_i - x_j| + d_i + a_j$ seconds otherwise ($j > i$).

Given values of x, a, b, c, d find the minimum time Scott can get to Cross, assuming he wants to visit each chair exactly once.

Input

The first line of the input contains three integers n, s and e ($2 \le n \le 5000, 1 \le s, e \le n, s \ne e$) — the total number of chairs, starting and ending positions of Scott.

The second line contains n integers $x_1, x_2, ..., x_n$ ($1 \le x_1 < x_2 < ... < x_n \le 10^9$).

The third line contains n integers $a_1, a_2, ..., a_n$ ($1 \le a_1, a_2, ..., a_n \le 10^9$).

The fourth line contains n integers $b_1, b_2, ..., b_n$ ($1 \le b_1, b_2, ..., b_n \le 10^9$).

The fifth line contains n integers $c_1, c_2, ..., c_n$ ($1 \le c_1, c_2, ..., c_n \le 10^9$).

The sixth line contains n integers $d_1, d_2, ..., d_n$ ($1 \le d_1, d_2, ..., d_n \le 10^9$).

Output

Print the minimum amount of time Scott needs to get to the Cross while visiting each chair exactly once.

Example

input
7 4 3 8 11 12 16 17 18 20 17 16 20 2 20 5 13 17 8 8 16 12 15 13 12 4 16 4 15 7 6 8 14 2 11 17 12 8
output
139

Note

In the sample testcase, an optimal solution would be . Spent time would be $17 + 24 + 23 + 20 + 33 + 22 = 139$.

E. Black Widow

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

Natalia Romanova is trying to test something on the new gun S.H.I.E.L.D gave her. In order to determine the result of the test, she needs to find the number of answers to a certain equation. The equation is of form:

Where represents logical OR and represents logical exclusive OR (XOR), and $v_{i,j}$ are some boolean variables or their negations. Natalia calls the left side of the equation a XNF formula. Each statement in brackets is called a clause, and $v_{i,j}$ are called literals.

In the equation Natalia has, the left side is actually a 2-XNF-2 containing variables $x_1, x_2, ..., x_m$ and their negations. An XNF formula is 2-XNF-2 if:

1. For each $1 \le i \le n$, $k_i \le 2$, i.e. the size of each clause doesn't exceed two.
2. Each variable occurs **in the formula at most two times** (with negation and without negation in total). Please note that it's possible that a variable occurs twice but its negation doesn't occur in any clause (or vice versa).

Natalia is given a formula of m variables, consisting of n clauses. Please, make sure to check the samples in order to properly understand how the formula looks like.

Natalia is more into fight than theory, so she asked you to tell her the number of answers to this equation. More precisely, you need to find the number of ways to set $x_1, ..., x_m$ with $true$ and $false$ (out of total of 2^m ways) so that the equation is satisfied. Since this number can be extremely large, you need to print the answer modulo $10^9 + 7$.

Please, note that some variable may appear twice in one clause, or not appear in the equation at all (but still, setting it to $false$ or $true$ gives different ways to set variables).

Input

The first line of input contains two integers n and m ($1 \le n, m \le 100\,000$) — the number of clauses and the number of variables respectively.

The next n lines contain the formula. The i-th of them starts with an integer k_i — the number of literals in the i-th clause. It is followed by k_i non-zero integers $a_{i,1}, ..., a_{i,k_i}$. If $a_{i,j} > 0$ then $v_{i,j}$ is $x_{a_{i,j}}$ otherwise it's negation of $x_{-a_{i,j}}$ ($1 \le k_i \le 2$, $-m \le a_{i,j} \le m$, $a_{i,j} \neq 0$).

Output

Print the answer modulo $1\,000\,000\,007$ ($10^9 + 7$) in one line.

Examples

input
6 7 2 4 -2 2 6 3 2 -7 1 2 -5 1 2 3 6 2 -2 -5

output
48

input
8 10 1 -5 2 4 -6 2 -2 -6 2 -7 9 2 10 -1 2 3 -1 2 -8 9 2 5 8

output
544

input
2 3 2 1 1 2 -3 3

output
4

Note

The equation in the first sample is:

The equation in the second sample is:

The equation in the third sample is:
