Problem Jaguar. Participants entry

Input file:	`input.txt` or standard input
Output file:	`output.txt` or standard output
Time limit:	3 seconds
Memory limit:	512 megabytes

At the open programming olympiad for school students, there are n participants numbered from 1 to n. Participant number i is wearing a t-shirt of color a_i. The organizers of the competition plan to invite participants to the competition hall in turn. To make this process visually pleasing, they want to avoid situations where two consecutive participants enter wearing t-shirts of the same color. To achieve this, participants will be invited according to the following algorithm:

- The first participant to enter the competition hall is participant number 1.
- Then, each new participant that is invited must have a t-shirt color that differs from the color of the last participant who entered. If there are several such participants, the one with the smallest number is chosen.
- Finally, if all remaining participants have the same t-shirt color as the last participant who entered, all remaining participants are invited in increasing order of their numbers.

On the night before the olympiad, the organizers prepared a plan for the participants entry, but just before the opening, they noticed that participants with neighboring numbers sometimes swap t-shirts. Of course, this led to the previous plan no longer complying with the rules, and the organizers needed to develop a new one.

You are required to respond to two types of queries:

1. Two participants with numbers x_i and $(x_i + 1)$ swap t-shirts.
2. Find out the order in which participant number y_i will enter if participants start entering according to the algorithm described above, starting from the first one, taking into account all previous t-shirt swaps.

Input

The first line contains two integers n and q ($1 \le n \le 500\,000$, $1 \le q \le 500\,000$) — the number of participants in the competition and the number of queries.

The second line contains n integers a_1, a_2, …, a_n ($1 \le a_i \le n$) — the initial colors of the participants' t-shirts in the order of their numbering.

The following q lines describe the queries. In the i-th line, there is an integer t_i ($1 \le t_i \le 2$) — the type of the i-th query.

1. If $t_i = 1$, then the next line contains an integer x_i ($1 \le x_i < n$). In this case, in the i-th query, participants with numbers x_i and $(x_i + 1)$ swap t-shirts.
2. If $t_i = 2$, then the next line contains an integer y_i ($1 \le y_i \le n$). In this case, in the i-th query, you need to find out the order in which participant number y_i will enter if participants start entering according to the algorithm described above, starting from the first one, taking into account all previous t-shirt swaps.

Output

For each query of the second type, you should output a single integer on a separate line — the answer to the query.

It is guaranteed that there is at least one query of the second type.

Examples

input	output
10 10 3 1 1 2 2 1 1 2 2 2 2 2 2 3 2 4 2 10 1 1 2 2 2 3 2 4 2 5 2 10	2 4 3 10 2 3 4 6 10
10 10 1 2 2 2 3 4 5 6 7 8 2 1 1 1 2 2 1 2 2 3 1 3 2 4 1 4 2 3 2 5	1 2 2 2 5 4

Note

In the first example for the problem, in the initial configuration, participants enter the competition hall in the order:

$$1,\quad 2,\quad 4,\quad 3,\quad 5,\quad 6,\quad 8,\quad 7,\quad 9,\quad 10$$

That is, participant number 2 enters second, participant number 3 enters fourth, participant number 4 enters third, and participant number 10 enters tenth.

After participants with numbers 1 and 2 swap t-shirts, the t-shirt colors of the participants are as follows:

$$1,\quad 3,\quad 1,\quad 2,\quad 2,\quad 1,\quad 1,\quad 2,\quad 2,\quad 2$$

Therefore, after this change, participants will enter the competition hall in the order:

$$1,\quad 2,\quad 3,\quad 4,\quad 6,\quad 5,\quad 7,\quad 8,\quad 9,\quad 10$$

That is, participant number 2 will enter second, participant number 3 will enter third, participant number 4 will enter fourth, participant number 5 will enter sixth, and participant number 10 will enter tenth.

In the second example for the problem, in the initial configuration, participants enter the competition hall in the order:

$$1,\quad 2,\quad 5,\quad 3,\quad 6,\quad 4,\quad 7,\quad 8,\quad 9,\quad 10$$

That is, participant number 1 enters first.

After participants with numbers 1 and 2 swap t-shirts, the t-shirt colors of the participants are as follows:

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

Therefore, after this change, participants will enter in the order:

1, 2, 3, 5, 4, 6, 7, 8, 9, 10

That is, participant number 2 enters second.

After participants with numbers 2 and 3 swap t-shirts, the t-shirt colors of the participants are as follows:

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

Therefore, after this change, participants will enter the competition hall in the order:

1, 3, 2, 5, 4, 6, 7, 8, 9, 10

That is, participant number 3 enters second.

After participants with numbers 3 and 4 swap t-shirts, the t-shirt colors of the participants are as follows:

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

Therefore, after this change, participants will enter in the order:

1, 4, 2, 5, 3, 6, 7, 8, 9, 10

That is, participant number 4 enters second.

After participants with numbers 4 and 5 swap t-shirts, the t-shirt colors of the participants are as follows:

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

Therefore, after this change, participants will enter in the order:

1, 4, 2, 5, 3, 6, 7, 8, 9, 10

That is, participant number 3 enters fifth, and participant number 5 enters fourth.

Scoring

The tests for this problem consist of 12 groups. Points for each group are awarded only if all tests in the group and all tests in some of the previous groups are passed. Note that passing the tests from the statement is not required for some groups. **Offline verification** means that the results of testing your solution on this group will only be available after the competition ends. The final score for each group is equal to the maximum score obtained for this group of tests across all submitted solutions.

Group	Points	Additional constraints	Required groups	Comment
		n, q		
0	0	–	–	Tests from the statement.
1	7	$n, q \le 500$	0	
2	9	$n, q \le 5\,000$	0, 1	
3	5	$n, q \le 10\,000$	0 – 2	
4	10	$n, q \le 100\,000$	0 – 3	
5	8	$n, q \le 200\,000$	0 – 4	
6	7	$n, q \le 300\,000$	0 – 5	
7	9	–	–	$1 \le a_i \le 2$
8	9	–	7	$1 \le a_i \le 5$
9	11	–	–	For any $i \ne j$: $a_i = 1$ or $a_i \ne a_j$ If $t_i = 2$, then $y_i = \lceil \frac{9n}{10} \rceil$
10	8	–	9	For any $i \ne j$: $a_i = 1$ or $a_i \ne a_j$
11	9	–	9	If $t_i = 2$, then $y_i = \lceil \frac{9n}{10} \rceil$
12	8	–	0 – 11	**Offline testing.**

Problem Volvo. Permutations and Queries

Input file:	`input.txt` or standard input
Output file:	`output.txt` or standard output
Time limit:	1 second
Memory limit:	512 megabytes

You are given a permutation p of length n. A permutation of length n is an array consisting of n distinct integers from 1 to n in arbitrary order. We define the *cost* of the permutation as the sum over all i from 1 to n of the quantity $(p_i)^i$ (the i-th element of the permutation raised to the power of i). Thus, the *cost* of the permutation p is given by

$$\sum_{i=1}^{n} (p_i)^i$$

There are q queries of three types:

1. Reverse. After this, your permutation p is replaced by the permutation q, such that $q_i = p_{n-i+1}$ for all i from 1 to n.
2. Invert. After this, your permutation p is replaced by the permutation q, such that $q_i = n - p_i + 1$ for all i from 1 to n.
3. Take the inverse. After this, your permutation p is replaced by the permutation q, such that $q_{p_i} = i$ for all i from 1 to n.

Note that after each operation, p remains a permutation.

After each query, you need to output the *cost* of the permutation.

Input

The first line contains two integers n and q ($1 \le n, q \le 100\,000$) — the length of the permutation and the number of queries.

The second line contains n positive integers $p_1, p_2, \ldots, p_n$ ($1 \le p_i \le n$) — the elements of the permutation. It is guaranteed that all p_i are distinct.

The third line contains q positive integers $b_1, b_2, \ldots, b_q$ ($1 \le b_i \le 3$) — the description of the queries. The number b_i means that the i-th modification query to be applied to the permutation is of type b_i.

Output

Output q numbers, the i-th of which is the remainder of the *cost* of the permutation modulo 998 244 353, after applying the first i queries.

Examples

input	output
5 5 1 2 3 4 5 1 2 3 1 2	65 3413 3413 65 3413
5 6 5 3 1 4 2 3 3 1 2 3 1	293 303 3225 215 317 3209

Note

Let's analyze the second example.

Initially, $p = [5, 3, 1, 4, 2]$.

The first query is of type 3, meaning take the inverse. The permutation after this query becomes $[3, 5, 2, 4, 1]$. The *cost* of this permutation is $3^1 + 5^2 + 2^3 + 4^4 + 1^5 = 3 + 25 + 8 + 256 + 1 = 293$.

The second query is of type 3, meaning take the inverse. The permutation after this query becomes $[5, 3, 1, 4, 2]$. The *cost* of this permutation is $5^1 + 3^2 + 1^3 + 4^4 + 2^5 = 5 + 9 + 1 + 256 + 32 = 303$.

The third query is of type 1, meaning reverse. The permutation after this query becomes $[2, 4, 1, 3, 5]$. The *cost* of this permutation is $2^1 + 4^2 + 1^3 + 3^4 + 5^5 = 3225$.

The fourth query is of type 2, meaning invert. The permutation after this query becomes $[4, 2, 5, 3, 1]$. The *cost* of this permutation is $4^1 + 2^2 + 5^3 + 3^4 + 1^5 = 215$.

The fifth query is of type 3, meaning take the inverse. The permutation after this query becomes $[5, 2, 4, 1, 3]$. The *cost* of this permutation is $5^1 + 2^2 + 4^3 + 1^4 + 3^5 = 317$.

The last query is of type 1, meaning reverse. The permutation after this query becomes $[3, 1, 4, 2, 5]$. The *cost* of this permutation is $3^1 + 1^2 + 4^3 + 2^4 + 5^5 = 3209$.

Scoring

The tests for this problem consist of five groups. Points for each group are awarded only if all tests in the group and all tests in some of the previous groups are passed. Note that passing the tests from the statement is not required for some groups. The final score for each group is the maximum score obtained for that group of tests across all submitted solutions.

Group	Points	Additional constraints		Required groups	Comment
		n	q		
0	0	–	–	–	Tests from the statement.
1	15	$n \le 1000$	$q \le 1000$	0	
2	22	–	–	–	$b_i = b_j$ for all $1 \le i, j \le q$
3	26	–	–	–	$b_i \le 2$ for all $1 \le i \le q$
4	16	–	–	–	$p_i = i$ for all $1 \le i \le n$
5	21	–	–	0 – 4	

Problem Toyota. Tasks from Sasha

Input file:	`input.txt` or standard input
Output file:	`output.txt` or standard output
Time limit:	3 seconds
Memory limit:	512 megabytes

Sasha recently moved into a multi-story building. The building has a total of n floors, numbered from 1 to n. There is exactly one resident living on each floor. Between the floors, there are $n-1$ staircases, and these staircases are not necessarily built between adjacent floors. It is known that for every floor except the first, there is exactly one staircase leading to a lower floor. For the i-th floor ($2 \le i \le n$), this staircase leads to the floor numbered p_i.

Sasha plans to solve k tasks numbered from 1 to k. For the task with number i, Sasha has calculated that it is most optimal to solve it on the floor numbered x_i. Since the tasks are different from each other, all values of x_i are distinct.

It is boring to solve tasks alone, so for each task, Sasha wants to invite at least one resident to solve it together. However, the residents of the building really dislike climbing stairs and are only willing to descend to the floors where the tasks will be solved. Therefore, Sasha can invite a resident from floor j to solve task i only if there is a way to reach floor x_i from floor j, using several, possibly zero, staircases that each time lead to a floor with a lower number. Thus, a resident from floor j can solve task i only if $j = x_i$ or $p_j = x_i$, or $p_{p_j} = x_i$, and so on.

Residents really dislike descending the stairs unnecessarily. Therefore, if Sasha invites a certain set of people to solve a task, they will only be willing to gather to solve the task on the **highest floor that they all can descend to**. For example, if there is a staircase from floor 3 to floor 2, then Sasha cannot invite residents from floors 2 and 3 to solve the task on floor 1, as all residents can gather on a higher floor.

Sasha does not like to appear intrusive, so he will invite a resident from each floor to solve no more than one task. At the same time, he may choose not to invite residents from some floors to solve tasks.

Sasha has one more favorite task that he will not share with anyone except you. But for him to tell you about it, you need to help him count the number of different ways to invite residents to solve tasks with him, ensuring that all restrictions are met. Two ways are considered different if at least one task is solved by a different set of residents.

Input

The first line contains two integers n and k ($3 \le n \le 10^6$, $1 \le k \le \min(n, 2000)$) — the number of floors in the building and the number of tasks.

The second line contains k integers $x_1, x_2, \ldots, x_k$ ($1 \le x_i \le n$) — the floors on which Sasha will solve the tasks. It is guaranteed that all x_i are distinct.

The third line contains $n-1$ integers $p_2, p_3, \ldots, p_n$ ($1 \le p_i < i$), where p_i describes the number of the floor to which the staircase leads down from floor i.

Output

Output one number — the number of different ways to invite residents of the building to solve tasks together with Sasha, modulo 998 244 353.

Examples

input	output
3 1 1 1 1	5
6 2 2 5 1 2 3 4 5	12
7 3 2 7 1 1 1 2 2 3 3	62

Note

In the first example, Sasha has five ways to invite residents to solve tasks:

- only with the resident on floor 1;
- with residents on floors 1 and 2;
- with residents on floors 1 and 3;
- with residents on floors 1, 2, and 3;
- with residents on floors 2 and 3;

Sasha cannot invite the resident from floor 2 alone to solve the task, as then the highest floor where all residents willing to solve the task could gather would be floor 2, while Sasha wants to solve the task on floor 1.

In the second example, two different suitable ways to invite residents to solve tasks could be as follows:

- Invite residents from floors 2 and 6 to solve the first task, and the resident from floor 5 to solve the second task.
- Invite the resident from floor 2 to solve the first task, and residents from floors 5 and 6 to solve the second task.

Scoring

The tests for this problem consist of nine groups. Points for each group are awarded only if all tests in the group and all tests in some of the previous groups are passed. Note that passing the tests from the statement is not required for some groups. **Offline verification** means that the results of testing your solution on this group will only be available after the competition ends. The final score for each group is equal to the maximum score obtained for that group of tests across all submitted solutions.

Group	Points	Additional constraints		Required groups	Comment
		n	k		
0	0	–	–	–	Tests from the statement.
1	12	$n \le 10$	$k \le 10$	0	
2	13	$n \le 500$	$k \le 500$	0, 1	
3	9	–	$k = 1$	–	
4	10	–	–	–	$p_i = i - 1$
5	13	–	–	4	Each floor is connected to no more than two floors with a higher number
6	14	$n \le 200\,000$	$k \le 500$	0 – 2	
7	11	–	$k \le 500$	0 – 3, 6	
8	10	–	$k \le 1000$	0 – 3, 6, 7	
9	8	–	–	0 – 8	**Offline verification**

Problem Mercedes. Anxiety Before the Olympiad

Input file:	`input.txt` or standard input
Output file:	`output.txt` or standard output
Time limit:	1 second
Memory limit:	512 megabytes

Before entering a closed olympiad in informatics, there are n participants waiting for entrance, numbered from 1 to n. It is known that every minute, one participant will be invited into the competition hall in the order of their numbers. In one minute, the first participant will enter the hall, in two minutes the second participant will enter, and so on. Thus, the i-th participant will enter the hall i minutes after the start of the entrance process.

Each participant has a certain level of anxiety before the olympiad. The anxiety level of each participant is given by some integer (which may be negative). Before the start of the entrance process, the anxiety level of participant i is equal to a_i. Every minute, the anxiety level of the participant will change by b_i. Therefore, after x minutes from the start of the entrance process, the anxiety level of participant i will be equal to $a_i + x \cdot b_i$.

Alexander is an experienced psychologist who decided to calm the participants waiting to enter the olympiad. To do this, Alexander talks to the participants and calms them down. He can talk to each participant no more than once. After talking to Alexander, the anxiety level of the participant becomes 0 and does not change afterward. The *effectiveness* of Alexander's work with participant i is considered to be the level of anxiety of the participant at the moment of communication with Alexander. Thus, if Alexander talks to participant i after t_i minutes from the start of the entrance process into the olympiad, the *effectiveness* will be equal to $a_i + t_i \cdot b_i$. Note that if the anxiety level of the participant was negative, then the *effectiveness* will also be negative.

Alexander will work with the participants in the order of their numbers. However, Alexander does not have to talk to all participants, meaning it is possible that he will not communicate with the last participants in line. Note that Alexander will talk to each participant no more than once and this must happen before the participant enters the competition hall. At the same time, Alexander can communicate with several participants at once every minute. More formally, Alexander's work process can be described as follows:

- In total, Alexander will talk to the first k participants in line, where k is chosen by him.
- For each of the first k participants, we will fix a non-negative integer t_i — the time of communication with Alexander. Note that t_i can be equal to zero, which means that Alexander talked to participant i before the first participant was allowed into the competition hall.
- For each i from 1 to k, $t_i < i$, as Alexander must talk to the participants before they enter the competition hall.
- For each i from 1 to $k-1$, $t_i \le t_{i+1}$, as Alexander talks to the participants in the order of their numbers.
- The *total effectiveness* of Alexander's communication with the participants will be described by the following formula:

$$\sum_{i=1}^{k} (a_i + t_i \cdot b_i)$$

Alexander has a predetermined plan for working with the participants. The work plan is given by a sequence of n integers $p_1, p_2, \ldots, p_n$. For each i, if $p_i \neq -1$, then by the time the first i participants are allowed into the competition hall (after i minutes from the start of the entrance process), Alexander must talk to the first p_i participants and no one else. In this case, it is also guaranteed that $p_i \ge i$. If $p_i = -1$, it means that there are no restrictions on the number of participants with whom Alexander must work after i minutes from the start of the entrance process.

More formally, if $p_i \neq -1$, then:

- $p_i \geq i$;
- $t_{p_i} < i$;
- For any j, such that $p_i < j \leq k$, it holds that $t_j \geq i$.

Help Alexander determine what the maximum possible *total effectiveness* of his work with the participants can be while satisfying all the constraints. It is guaranteed that the answer always exists.

Input

The first line contains a single integer n ($1 \leq n \leq 10^6$) — the number of participants waiting in line to enter the competition hall of the olympiad.

The next n lines contain two integers a_i and b_i ($-10^9 \leq a_i \leq 10^9$, $-10^6 \leq b_i \leq 10^6$) — the anxiety parameters of the i-th participant.

The following line contains n integers $p_1, p_2, \ldots, p_n$ ($i \leq p_i \leq n$ or $p_i = -1$) — the description of the work plan for Alexander.

It is guaranteed that for any pair $1 \leq i < j \leq n$, it holds that $p_i \leq p_j$, if $p_i \neq -1$ and $p_j \neq -1$.

Output

Output a single number — the maximum possible *total effectiveness* of Alexander's work.

It can be shown that Alexander will always be able to perform his work while adhering to all additional constraints and the work plan.

Examples

input	output
4 3 -6 4 -10 -7 -3 -3 6 3 3 -1 -1	15
4 -6 -1 -5 14 0 10 -30 2 2 3 -1 -1	-1
4 -6 -1 -5 14 0 10 -30 2 -1 -1 -1 -1	23

Note

In the first test example, it is optimal to choose $k = 4$ and $t = \{0, 0, 0, 3\}$. Then the *total effectiveness* will be:

$$(3 + t_0 \cdot (-6)) + (4 + t_1 \cdot (-3)) + (-7 + t_2 \cdot (-3)) + (-3 + t_4 \cdot 6) =$$

$$= (3 + 0 \cdot (-6)) + (4 + 0 \cdot (-3)) + (-7 + 0 \cdot (-3)) + (-3 + 3 \cdot 6) = 15$$

In the second test example, it is optimal to choose $k = 3$ and $t = \{0, 0, 1\}$. Then the *total effectiveness* will be:

$$(-6 + t_0 \cdot (-1)) + (-5 + t_1 \cdot 14) + (0 + t_2 \cdot 10) = (-6 + 0 \cdot (-1)) + (-5 + 0 \cdot 14) + (0 + 1 \cdot 10) = -1$$

In the third test example, it is optimal to choose $k = 3$ and $t = \{0, 1, 2\}$. Then the *total effectiveness* will be:

$$(-6 + t_0 \cdot (-1)) + (-5 + t_1 \cdot 14) + (0 + t_2 \cdot 10) = (-6 + 0 \cdot (-1)) + (-5 + 1 \cdot 14) + (0 + 2 \cdot 10) = 23$$

Scoring

The tests for this problem consist of fourteen groups. Points for each group are awarded only if all tests of the group and all tests of some previous groups are passed. Note that passing the tests from the statement is not required for some groups. **Offline testing** means that the results of testing your solution on this group will only be available after the end of the competition. The final score for each group is equal to the maximum score obtained for that group of tests across all submissions.

Group	Points	Constraints			Necessary groups	Comment
		n	b_i	p_i		
0	0	–	–	–	–	Tests from the statement.
1	6	$n \le 100$	–	$p_i = -1$	–	
2	6			–	0 – 1	
3	7	$n \le 5000$	–	$p_i = -1$	1	
4	6			–	0 – 3	
5	7	–	$b_i \le 0$	$p_i = -1$	–	
6	5			–	5	
7	7	–	$b_i \ge 0$	$p_i = -1$	–	
8	5			–	7	
9	9	–	$b_i \le b_{i+1}$	$p_i = -1$	–	
10	8			–	9	
11	10	$n \le 100\,000$	–	$p_i = -1$	–	Number $b_i > 0$ no more than 10
12	7			–	11	
13	9	–	–	$p_i = -1$	1, 3, 5, 7, 9, 11	**Offline testing**
14	8			–	0 – 13	