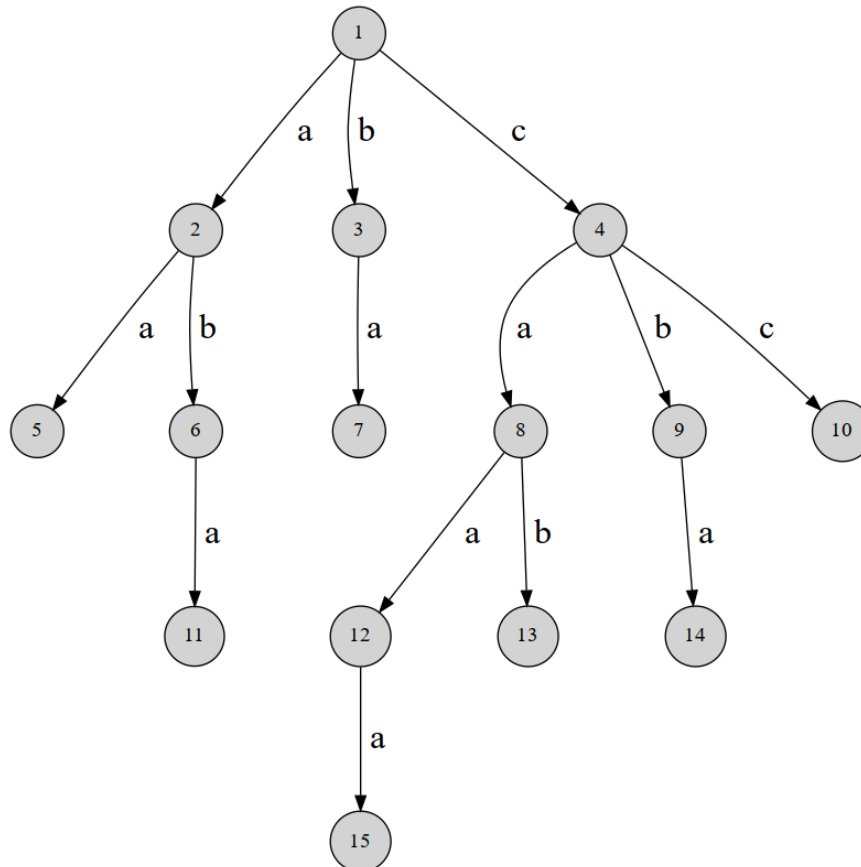


Problem A. Assembling Strings from a Trie

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

A trie (prefix tree) is a tree-like data structure in which edges are labeled with characters. From any node, all outgoing edges have distinct labels, and the sequence of labels on the path from the root to a node represents a string.

Take the picture below as an example. The path $1 \rightarrow 4 \rightarrow 8 \rightarrow 12$ represents the string “caa”.



In this problem, the alphabet consists of **all positive integers**, rather than a fixed set of characters.

There are n non-empty strings $S_1, S_2, \dots, S_n$ that form a trie T with m nodes. Node 1 is the root, while the indices of the remaining nodes have been arbitrarily permuted.

Given the structure of the trie, your task is to reconstruct the original strings.

- If multiple valid reconstructions exist, output the one with the minimum n .
- If there are still multiple valid solutions, output the lexicographically smallest sequence $[S_1, S_2, \dots, S_n]$.

The lexicographic order of string sequences is defined as follows: $[A_1, A_2, \dots, A_n]$ is smaller than $[B_1, B_2, \dots, B_n]$ if there exists an index k such that $A_i = B_i$ for all $1 \leq i < k$ and $A_k < B_k$.

The lexicographic order of two integer strings is defined as follows: a string C is lexicographically smaller than a string D if and only if either C is a prefix of D , or there exists an index k ($1 \leq k \leq \min(|C|, |D|)$) such that $C(i) = D(i)$ for all $1 \leq i < k$ and $C(k) < D(k)$, where $C(k)$ denotes the k -th element of string C .

Input

The first line contains an integer T ($1 \leq T \leq 500$), the number of test cases.

For each test case:

- One line contains an integer m ($2 \leq m \leq 200000$), the number of nodes in the trie.
- One line contains $m - 1$ integers $f_2, f_3, \dots, f_m$ ($1 \leq f_i < i$), where f_i denotes the parent of node i .

The total number of nodes over all test cases does not exceed 2,000,000.

Output

For each test case:

Output one integer n , the number of reconstructed strings.

Then output n lines. The strings themselves may be extremely long, so only their hash values are required.

The i -th line should contain the hash value of string S_i , defined as

$$\left(\sum_{j=1}^{|S_i|} 998244353^{|S_i|-j} \cdot S_i(j) \right) \bmod 2^{64}.$$

Example

standard input	standard output
4	1
2	1
1	2
3	1
1 1	2
4	2
1 2 1	1
6	1996488707
1 2 1 4 5	2
	998244354
	1992983577591021572

Note

For the example,

- **testcase 1:** The solution is $S_1 = [1]$.
- **testcase 2:** The solution is $S_1 = [1]$, $S_2 = [2]$.
- **testcase 3:** The solution is $S_1 = [1]$, $S_2 = [2, 1]$.
- **testcase 4:** The solution is $S_1 = [1, 1]$, $S_2 = [2, 1, 1]$.

Problem B. Budgeted Broadcasts

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

There are n cities in Byteland connected by $n - 1$ bidirectional roads, forming a tree, i.e., for any two cities, there is exactly one simple path between them.

Colin, the leader of Byteland, plans to establish the broadcasting system by establishing communication stations in some cities, with at most one station per city. For any two stations, if **there is no other station** on the path between them, they must develop a direct connection. The cost of such a connection equals the number of roads between them, i.e., their distance in the tree.

For a construction plan S (a subset of cities with stations), define:

- $|S|$: the number of stations built.
- $val(S)$: the total cost of all connections that must be established.

The allocated budget for plan S is $budget(S) = |S| \times val(S)$.

Colin wants to compute the **sum of budgets** over **all possible** 2^n construction plans. Two plans are different if and only if there exists at least one city where the presence of a station differs.

Since the answer can be large, output it modulo 998244353.

Input

The first line contains a single integer n ($2 \leq n \leq 10^6$), representing the number of cities.

For the following $n - 1$ lines, each line contains two positive integers u, v ($1 \leq u, v \leq n, u \neq v$) representing a road in Byteland. It is guaranteed that all the roads connect n cities into a tree.

Output

Output a single integer, representing the answer.

Examples

standard input	standard output
3 1 2 1 3	14
4 1 2 1 3 1 4	66

Note

For the first example,

When there are fewer than two stations, the budget needed is 0;

When the stations are built at $\{1, 2\}$ or $\{1, 3\}$, the budget needed is 2;

When the stations are built at $\{2, 3\}$, the budget needed is 4;

When the stations are built at $\{1, 2, 3\}$, the budget needed is 6.

This page is intentionally left blank

Problem C. Call for Papers

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 256 megabytes

Colin is a first-year Ph.D. student, working hard with many beautiful but unrealistic academic fantasies. He aims to design clever algorithms for essential problems and publish them in top conferences.

There are n conferences Colin is interested in. The i -th one starts receiving submissions at 8 a.m. on the day r_i , and the deadline to submit papers is 8 p.m. on the day d_i . After the submitted paper has been reviewed by the experts, the authors will be notified of the result at 2 p.m. on the day t_i . The result is either **Accepted** or **Rejected**. In addition, there is an impact factor w_i for the i -th conference, which represents the reputation of the conference. To simplify the problem, we guarantee that the impact factors of conferences are **pairwise distinct**.

But recently, he sadly found that his papers were rejected again and again. By summarizing the past data, Colin raised an assumption: the probability of acceptance for a paper is equal to the number of times it has been submitted over the impact factor of the conference. Formally, if Colin submits a paper to the i -th conference now, and it has already been submitted for k times (including this time), then the probability of acceptance this time is equal to

$$\min\left(1, \frac{k}{w_i}\right)$$

Now Colin has m papers he would like to submit, and the i -th paper will be finished at 9 a.m. on the day f_i . We say a conference is available at a time if it has already started receiving papers, but its deadline hasn't passed. Once Colin finishes each paper, if there does not exist any available conference at that time, then he will think this paper is not welcome and won't submit it anymore; if there exists any available conference at that time, he will instantly submit it to one of the available conferences with the highest impact factor. Then Colin will repeat as follows:

On the day of receiving the notification of the result, if the paper is **Rejected**:

- If any available conference exists **at that time**, Colin will instantly submit the paper to one of the available conferences with the highest impact factor and wait for the result.
- Otherwise, Colin will be pretty upset about this paper and won't submit it anymore, even if there are some further conferences that haven't started receiving papers.

If the paper is **Accepted**, Colin will earn reputation points, which are equal to the impact factor of the conference that accepts the paper. He will not submit the paper to other conferences.

Colin would like to ask, if his assumption is valid, what is the expected value of reputation points he can earn from each paper? If a paper is not accepted in the end, then Colin will earn 0 reputation points from it. Please find the answer modulo 998244353.

Notice that all the times mentioned are in AOE time, and there is no need to consider any issues related to time zones.

Input

The first line contains two integers n, m ($1 \leq n \leq 5 \times 10^3$, $1 \leq m \leq 10^5$), representing the number of conferences Colin is interested in, and the number of papers Colin would like to submit.

For the following n lines, each line contains four integers r_i, d_i, t_i, w_i ($1 \leq r_i < d_i < t_i \leq 10^9$, $1 \leq w_i \leq 10^8$), representing the information for the i -th conference: it starts receiving submissions from 8 a.m. on day r_i , the deadline is 8 p.m. on day d_i , and authors will be notified of the results at 2 p.m. on day t_i whether their submissions are accepted. It's guaranteed that all the w_i are pairwise distinct.

For the following m lines, each line contains a single integer f_i ($1 \leq f_i \leq 10^9$), representing that Colin will finish the i -th paper at 9 a.m. on the day f_i .

Output

For each paper, print a line with a single integer, representing the expected value of reputation points Colin could earn from it, modulo 998244353.

Example

standard input	standard output
5 3	1
1 2 3 1	249561091
2 3 4 2	332748120
3 4 5 3	
4 5 6 4	
5 6 7 5	
1	
2	
3	

Note

For the first paper, Colin will submit it to the first conference, and according to the assumption, the paper will definitely be accepted. Thus, the expected value of reputation he could earn is equal to the impact factor of the first conference, which is 1.

For the second paper, Colin will first submit it to the second conference with 50% probability of being accepted. If it is rejected, he will receive the notification on the day 4, then he will submit it to the fourth conference, with $\frac{2}{4} = 50\%$ probability of being accepted. If it is still rejected, then he will receive the notification on day 6, then he will submit it to the fifth conference, with $\frac{3}{5} = 60\%$ probability of being accepted. If it is still rejected, then there is no more chance to submit it. Thus, the expected value is $2 \times 50\% + 4 \times 50\% \times 50\% + 5 \times 50\% \times 50\% \times 60\% = \frac{11}{4}$, which equals 249561091 modulo 998244353.

Problem D. Dune

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 256 megabytes

Colin is a big fan of the *Dune* series movies and was deeply impressed by the scene in which Paul Atreides summons and rides a giant sandworm!



Colin noticed that the timing of summoning sandworms is very important. There are always so many mysteries hidden under the calm dunes! Therefore, he modeled “Summoning Sandworms” and designed a problem for you:

To simplify the problem, we describe the topography of the dunes with a one-dimensional numerical axis. There are n sandworms moving parallel to the number axis. Specifically, at the initial time $t = 0$, the body of the i -th sandworm can be described by the segment $[l_i, r_i]$ (thus, the length of the i -th sandworm is $r_i - l_i$), and it will move in the positive direction of the number axis at a speed of v_i per second. That is, at time t , the body of the i -th worm can be described by the segment $[l_i + v_i \times t, r_i + v_i \times t]$.

Your position is at coordinate x , and you can choose **any nonnegative real number** time t' to summon sandworms passing through your position (i.e., at time t' , the coordinate $x \in [l_i + v_i \times t', r_i + v_i \times t']$). However, there may be more than one sandworm passing through your position at this moment. In this case, you can only summon **the shortest one** among these sandworms.

There's nothing more exciting than summoning a giant sandworm! So Colin wants to know what is the longest length of sandworm you can possibly summon.

Input

The first line contains two integers n, x ($1 \leq n \leq 10^5, 0 \leq x \leq 10^9$), representing the number of sandworms and your position.

In the following n lines, each line contains three integers l_i, r_i, v_i ($-10^9 \leq l_i \leq r_i \leq 10^9, 1 \leq v_i \leq 10^9$), describing the i -th sandworm.

It's guaranteed that at least one sandworm will pass by your position.

Output

Output a single integer representing the longest length of sandworm you can possibly summon.

Example

standard input	standard output
4 3 4 1000000000 1 2 4 1 1 1000000000 1000000000 1 2 3	2

Problem E. Expected Inversions

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 256 megabytes

Quark the corgi has a bin full of balls, each of which has an integer written on it. The integers form a **multiset** $S = \{b_1, b_2, \dots, b_n\}$, and he wants to use it to generate a random sequence $A = a_1, a_2, \dots, a_n$, that he will later use to decide in what order to fetch the balls.



Initially, the sequence A is empty. Quark performs the following operation exactly n rounds:

For the i -th round, let S_i be the multiset of the remaining numbers after previous rounds ($S_i = S$ if $i = 1$). Let sum_i be the sum of all elements in S_i . Quark will randomly select an element from S_i with probability proportional to its value. Specifically, the probability of selecting $b_j \in S_i$ is $\frac{b_j}{sum_i}$. Then Quark will append the selected number to the end of the sequence A and remove it from S_i .

After all n rounds, Quark obtains a random sequence A . He is interested in the expected number of inversions in A , i.e., the expected number of index pairs (i, j) such that $i < j$ and $a_i > a_j$.

Since corgis are known to dislike fractions, you only need to output the answer modulo 998244353.

Input

The first line contains a single integer n ($2 \leq n \leq 10^6$), representing the size of the multiset S .

The second line contains n integers $b_1, b_2, \dots, b_n$ ($1 \leq b_i \leq 10^5$), representing the elements in S .

Output

A single integer, representing the expected number of inversions in the random sequence A , modulo 998244353.

Examples

standard input	standard output
3 1 2 3	715408455
3 2 3 3	598946613

Note

For the first example, the table below shows the number of inversions and the corresponding probability to generate it for each possible sequence A .

A	Number of inversions	Probability
1, 2, 3	0	$\frac{1}{15}$
1, 3, 2	1	$\frac{1}{10}$
2, 1, 3	1	$\frac{1}{12}$
2, 3, 1	2	$\frac{1}{4}$
3, 1, 2	2	$\frac{1}{6}$
3, 2, 1	3	$\frac{1}{3}$

Thus, the expected number of inversions equals $\frac{1}{10} + \frac{1}{12} + 2 \times \frac{1}{4} + 2 \times \frac{1}{6} + 3 \times \frac{1}{3} = \frac{121}{60}$, which equals 715408455 modulo 998244353.

Problem F. Finding Cyclic Isomorphism

Input file: `standard input`
Output file: `standard output`
Time limit: 4 seconds
Memory limit: 1024 megabytes

Colin is learning isomorphism theory recently and tries to apply it to string problems!

Define two strings S, T to be **cyclically isomorphic** if and only if they satisfy all the following conditions:

- their lengths are equal, i.e., $|S| = |T|$;
- there exists an integer $i \in [0, |S|)$ such that if we cyclically right-shift S for i times, the resulting string $S' = T$.

For example, strings **abcb** and **babc** are cyclically isomorphic since we can cyclically right-shift the first string once to get the second string.

Now you are given n non-empty pattern strings $S_1, S_2, \dots, S_n$, and another non-empty text string T .

Colin raises q questions; each question is described by an interval $[l_i, r_i]$ ($1 \leq l_i < r_i \leq |T|$). Colin wants to know how many integers x satisfy all the following conditions:

- $x \in [l_i, \min(|T| + l_i - r_i + 1, r_i - 1)]$
- $T[l_i, x - 1] = T[r_i, x + r_i - l_i - 1]$
- $T[x, r_i - 1]$ is cyclically isomorphic with at least one of $S_1, S_2, \dots, S_n$.

Recall that $|T|$ represents the length of string T , and $T[a, b]$ ($1 \leq a \leq b \leq |T|$) represents the substring of T from the a -th character to the b -th character (including both boundaries). $T[a, b]$ ($a > b$) represents an empty string, and empty strings are always equal.

Input

The first line contains two integers n, q ($1 \leq n \leq 5 \times 10^5, 1 \leq q \leq 10^6$), representing the number of pattern strings and the number of questions.

For the following n lines, each line contains a non-empty string S_i consisting of lowercase letters, representing the i -th pattern string. It's guaranteed that $\sum_{i=1}^n |S_i| \leq 5 \times 10^5$.

The following line contains a non-empty string T ($2 \leq |T| \leq 10^6$) consisting of lowercase letters.

For the following q lines, each line contains two integers l_i, r_i ($1 \leq l_i < r_i \leq |T|$), describing the i -th question.

Output

Output q lines; the i -th line contains a single integer, representing the answer for the i -th question.

Example

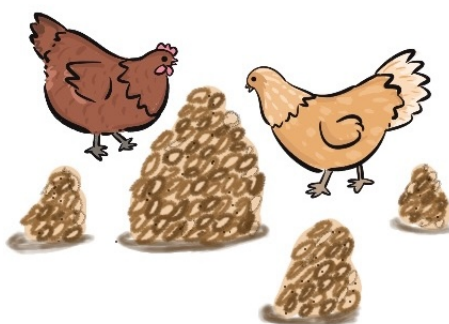
standard input	standard output
5 5	1
baab	0
aa	1
aaab	1
ab	0
abaaa	
bbaabababb	
7 9	
1 4	
3 7	
7 9	
5 10	

Problem G. Game of Quantum Computing Chickens

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

Colin is studying quantum game theory by watching his two quantum computing chickens, Chi and Ken, play a peculiar seed-picking game. Initially, there are several piles of seeds. The chickens take turns, with Chi moving first. The rules are:

- If all piles are empty, the current chicken loses immediately.
- Otherwise, the current chicken must take seeds from the **largest pile**. She must take at least one seed and may take all seeds from that pile.



Due to quantum entanglement, the initial pile sizes are uncertain. Colin models this using a **multiset** S , initially empty. He asks you to support two operations:

- **1 x y**: Add y copies of number x to the multiset S .
- **2 l r**: Generate a new multiset S' as follows: for each $x \in S$, if $l \leq x \leq r$, include x in S' with probability 0.5; otherwise, do nothing. You then need to compute the probability that Chi has a winning strategy when the piles are exactly the numbers in S' , modulo 998244353. S' will be discarded afterwards and doesn't change S .

Note that S' may be empty, in which case the first chicken (Chi) loses.

Input

The first line contains an integer n ($1 \leq n \leq 5 \times 10^5$), the number of operations.

Each of the next n lines contains three positive integers:

- For type 1: **1 x y** ($1 \leq x, y \leq 10^9$).
- For type 2: **2 l r** ($1 \leq l \leq r \leq 10^9$).

Output

For each type 2 operation, output a line with a single integer: the probability that Chi has a winning strategy, modulo 998244353.

Example

standard input	standard output
6	499122177
1 1 1	499122177
2 1 1	374341633
1 1 1	
2 1 1	
1 2 2	
2 1 2	

Note

For the example:

- First query: $S = \{1\}$. S' is empty with 0.5 probability, or $\{1\}$ with 0.5 probability. Chi wins only if $S' = \{1\}$.
- Second query: $S = \{1, 1\}$. Possible S' : empty (25%), $\{1\}$ (50%), $\{1, 1\}$ (25%). Chi wins only if $S' = \{1\}$.
- Third query: $S = \{1, 1, 2, 2\}$. Probability calculation yields $374341633 \equiv \frac{5}{8} \pmod{998244353}$.

Problem H. Heaviest Convex Set

Input file: standard input
Output file: standard output
Time limit: 6 seconds
Memory limit: 512 megabytes

A convex polygon is a simple polygon whose interior forms a convex set. Specifically, if any side of the polygon is extended indefinitely in both directions to form a straight line, all other sides lie entirely on the same side of this line. All interior angles of a convex polygon are less than 180° , and the line segment between any two vertices lies inside the polygon or on its boundary. A valid convex polygon must contain at least three vertices, and no three vertices are collinear.

Colin has n points on a plane, the i -th point has coordinates (x_i, y_i) and a weight w_i .

Now Colin wants you to delete at most k points so that the remaining points form exactly the vertex set of a convex polygon; under this condition, you need to maximize the sum of the weights of the remaining points.

Input

The first line contains two integers n and k ($3 \leq n \leq 5 \times 10^4$, $0 \leq k \leq 20$, $n - k \geq 3$), representing the number of points and the maximum number of points that can be deleted.

The i -th of the following n lines contains three integers x_i, y_i, w_i ($1 \leq x_i, y_i \leq 10^9$, $|w_i| \leq 10^9$), describing the coordinates and weight of each point.

It is guaranteed that all coordinates are distinct.

Output

Output a single line containing an integer: the maximum possible sum of weights of the remaining points after deleting at most k points, such that they form exactly the vertex set of a convex polygon.

If no valid solution exists, output “impossible”.

Examples

standard input	standard output
4 1 1 1 5 10 1 8 10 10 -16 1 10 3	16
9 6 1 1 1 1 2 1 1 3 1 2 1 1 2 2 1 2 3 1 3 1 1 3 2 1 3 3 1	6
5 0 1 1 1 1 3 1 3 1 1 3 3 1 2 2 1	impossible

Problem I. Ice Skating

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

Maru the golden retriever is passionate about ice skating—he lives for the thrill of gliding at breakneck speed across the rink. Unfortunately, his enthusiasm far exceeds his skill. Maru's control is... questionable at best (possibly because he normally uses all four of his legs). More often than not, he ends up crashing into obstacles or launching himself spectacularly off the edge of the rink.



The rink can be described as a grid of n rows and m columns, and we use (r, c) to denote the cell on the r -th row and the c -th column. Thus $(1, 1)$ is the upper-left corner, and (n, m) is the bottom-right corner. Initially, there are k obstacles on the rink where the i -th obstacle is on the cell (x_i, y_i) , and all the remaining $n \times m - k$ cells are empty.

If Maru is currently at (a, b) , he has two options to move:

- Skate right: Keep moving right until meeting an obstacle or the boundary of the rink. That is, if $(a, b + 1)$ is within the rink and there is no obstacle on it, Maru will move to $(a, b + 1)$ and repeat checking this condition; otherwise, he stops moving.
- Skate down: Keep moving down until meeting an obstacle or the boundary of the rink. That is, if $(a + 1, b)$ is within the rink and there is no obstacle on it, Maru will move to $(a + 1, b)$ and repeat checking this condition; otherwise, he stops moving.

Notice that, regardless of how far Maru moves in one option, it is considered one move.

Maru doesn't have much energy left, because his humans refused to give him a second breakfast; he wants to know the minimum number of moves to reach a "stable position". A "stable position" is a position where he cannot move to another position, no matter how he chooses to move next.

You need to support q operations, each of which belongs to one of the following two types:

- 1 a b : A query for the minimum number of moves to reach a "stable position", if Maru starts his skating at (a, b) . It's guaranteed that there is no obstacle on cell (a, b) .
- 2 a b : Put an obstacle on cell (a, b) . It's guaranteed that there is no obstacle on cell (a, b) before this operation.

Input

The first line contains four integers n, m, k, q ($1 \leq n, m, k, q \leq 10^5$), representing the size of the rink, the number of obstacles initially, and the number of operations.

In the following k lines, each line contains two integers x_i, y_i ($1 \leq x_i \leq n, 1 \leq y_i \leq m$), representing the position of the i -th obstacle. It's guaranteed that the positions of obstacles are pairwise distinct.

In the following q lines, each line contains three integers op_i, a_i, b_i ($op_i \in \{1, 2\}, 1 \leq a_i \leq n, 1 \leq b_i \leq m$), representing the i -th operation.

Output

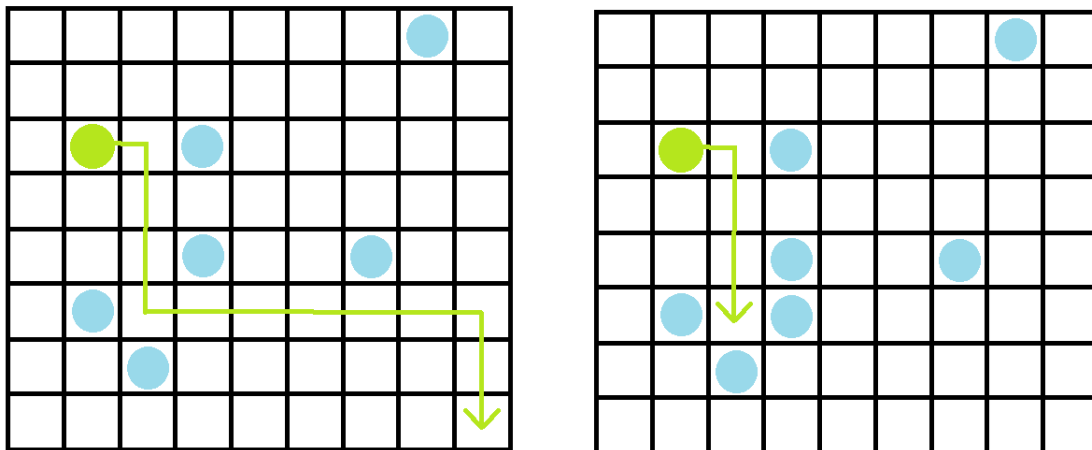
For each query, output a single line with a single integer, representing the answer.

Example

standard input	standard output
8 9 6 3	4
3 4	2
5 4	
7 3	
5 7	
1 8	
6 2	
1 3 2	
2 6 4	
1 3 2	

Note

The image below shows one possible answer to each query in the example.



Problem J. Jamming of Intelligence

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

Narwhy, an underdog underwater undercover agent of Byteland, is executing a signal jamming operation against enemy intelligence networks. His objective is to introduce inconsistencies into the enemy's surveillance data to disrupt their decryption efforts.



The enemy's current intelligence database contains:

- n submarines, numbered $1, 2, \dots, n$, each positioned at coordinates (x_i, y_i) . Their firing ranges r_i are unknown positive real values.
- **a signal overlap reports**: each specifying submarines u and v that can jointly engage a target, forming a coverage constraint:

$$r_u + r_v \geq \text{dist}(u, v) = \sqrt{(x_u - x_v)^2 + (y_u - y_v)^2}.$$

- **b maximum range signatures**: each indicating that between submarines u and v , the stronger emitter has range at least w :

$$\max(r_u, r_v) \geq w.$$

- **c minimum range signatures**: each indicating that between submarines u and v , the weaker emitter has range at most w :

$$\min(r_u, r_v) \leq w.$$

Narwhy's jamming strategy is to make the entire dataset **inconsistent**—ensuring no assignment of positive values $r_1, \dots, r_n$ can satisfy all constraints. This would force the enemy into time-consuming signal verification.

If the existing intelligence is already inconsistent, no action is needed. Otherwise, Narwhy can inject some of the given d **range suppression signals** into the network. The i -th such signal imposes an upper bound w_i on submarine u_i 's range, but each injection carries an operational cost p_i .

Since Narwhy's main expertise is actually in narwhal tusk operations, he needs your help! Determine the minimum total cost to jam the intelligence system by selecting suppression signals to create an unsatisfiable constraint set.

Input

The first line contains an integer T ($1 \leq T \leq 500$), representing the number of test cases.

For each test case:

- The first line contains five integers n, a, b, c, d ($2 \leq n \leq 5 \times 10^4, 0 \leq a, b, c \leq 5 \times 10^4, 0 \leq d \leq 250$), representing the number of submarines, signal overlap reports, maximum range signatures, minimum range signatures, and available suppression signals respectively.
- For the next n lines, each line contains two integers x_i, y_i ($1 \leq x_i, y_i \leq n$), representing the coordinates of submarine i .
- For the next a lines, each line contains two integers u, v ($1 \leq u < v \leq n$), representing a signal overlap report. It is guaranteed that no unordered pair (u, v) appears more than once among all the signal overlap reports.
- For the next b lines, each line contains three integers u, v, w ($1 \leq u < v \leq n, 1 \leq w \leq n$), representing a maximum range signature. It is guaranteed that no unordered pair (u, v) appears more than once among all the maximum range signatures.
- For the next c lines, each line contains three integers u, v, w ($1 \leq u < v \leq n, 1 \leq w \leq n$), representing a minimum range signature. It is guaranteed that no unordered pair (u, v) appears more than once among all the minimum range signatures.
- For the next d lines, each line contains three integers u_i, w_i, p_i ($1 \leq u_i, w_i \leq n, 1 \leq p_i \leq 10^9$), representing an available suppression signal. It is guaranteed that no submarine u appears more than once among all the available suppression signals.

It is guaranteed that for all test cases, $\sum n \leq 4 \times 10^5, \sum a \leq 4 \times 10^5, \sum b \leq 4 \times 10^5, \sum c \leq 4 \times 10^5$.

Output

For each test case, output a single integer:

- “0” if no insertion is needed;
- otherwise “-1” if it is impossible;
- otherwise the minimum total cost required.

Example

standard input	standard output
3	-1
2 0 0 0 1	0
1 1	5
1 1	
2 2 100	
4 3 0 3 0	
1 1	
4 4	
1 4	
2 3	
1 2	
1 3	
2 3	
1 2 1	
1 3 1	
2 3 1	
3 1 1 1 2	
1 1	
1 2	
3 3	
1 3	
2 3 2	
1 2 1	
3 1 5	
2 1 7	

This page is intentionally left blank

Problem K. Keeping Cactus Alive

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 256 megabytes

An undirected connected graph is called a cactus if and only if each of its edges belongs to at most one simple cycle[†]. Note that the cactus graph contains no self-loops, but **may have multiple edges** connecting the same pair of nodes.

There is an aging circuit network with n nodes and m undirected edges, and the network is guaranteed to be a cactus. Each edge (u_i, v_i) has a durability d_i , representing the length of time during which it can transport electricity.

Your task is to choose an activation time s_i , ($s_i \geq 0$) for each edge (u_i, v_i) . Once activated, the edge can transport electricity only in the time interval $[s_i, s_i + d_i]$.

However, this fragile network must remain connected at all times. First, the edges that can transport electricity at time 0 must connect all n nodes. After that, if at any moment the network becomes disconnected, the entire power system immediately collapses.

The question is: how long can you keep the network running?

Formally, if your answer is A , then there must exist a choice of activation times $\{s_i\}$ such that for every real number $t \in [0, A]$, all n nodes are connected by the edges that can transport electricity at time t .

[†] : In this problem, a simple cycle is a pair of a sequence of vertices $(v_0, \dots, v_{L-1})$ and a sequence of edges $(e_0, \dots, e_{L-1})$ satisfying the following conditions.

- $L \geq 1$
- $i \neq j \implies v_i \neq v_j, e_i \neq e_j$
- For $0 \leq i < L - 1$, the edge e_i connects v_i and v_{i+1}
- The edge e_{L-1} connects v_{L-1} and v_0

Input

The first line of the input contains an integer T , indicating the number of the test cases.

For each test case:

The first line contains two integers n, m ($2 \leq n \leq 10^5, 1 \leq m \leq 2 \times 10^5$), indicating the number of the nodes and the edges in the circuit network.

Then the following m lines, each line contains three integers u_i, v_i, d_i ($1 \leq u_i, v_i \leq n, u_i \neq v_i, 1 \leq d_i \leq 10^9$), indicating an edge in the given graph. It's guaranteed that the graph is a cactus.

It is guaranteed that for all test cases, $\sum n \leq 5 \times 10^5, \sum m \leq 10^6$.

Output

For each test case: print one line containing a single integer, indicating the maximum length of time that you can keep this network running.

Example

standard input	standard output
2	2
3 3	5
1 2 1	
2 3 1	
3 1 2	
5 6	
1 2 3	
1 2 2	
2 3 3	
2 4 5	
2 5 5	
3 4 2	

Note

For the first example, you can set $s_1 = s_3 = 0, s_2 = 1$ so all the nodes are connected by edge 1 and edge 3 in the time interval $[0, 1]$, and by edge 2 and edge 3 in the time interval $[1, 2]$.

Problem L. Love Choice

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 256 megabytes

It's the beginning of a new relationship. Will this one result in joy or sorrow? In LoveChoice, you choose the actions that determine how the relationship unfolds. Pick your responses to each of the situations. These endearing and bittersweet stories are surprisingly realistic and easy to relate to.



To win Eva's heart, Colin studies hard by playing LoveChoice on Steam. To simplify the problem, we describe it as follows:

There are n multiple-choice questions; the i -th question has a_i normal choices. Additionally, some questions have an alternative hidden choice. There is only one correct choice for each question, and if a hidden choice exists, it must be the correct choice.

In a round of the game, you must answer all the questions in order. **After answering all the questions**, the system will tell you if **each question** is answered correctly.

Colin is calm. For each question, he will only choose one of the choices that may not be the wrong choice, with equal probability each time. However, Colin's observation ability is relatively ordinary, so if a question i has a hidden choice, Colin will not be aware of its existence before knowing that all the a_i normal choices for this question are wrong.

Colin is hardworking. His goal is to answer all questions correctly in a round. Please calculate the expected number of rounds for him to **answer all the questions correctly** for the first time.

Input

The first line has two integers n, m ($1 \leq m \leq n \leq 5 \times 10^3$), representing that there are n questions in the game and that there are m questions that have a hidden choice.

The second line has n integers $a_1, a_2, \dots, a_n$ ($1 \leq a_i \leq 10^9$), representing that there are a_i normal choices for the i -th question.

The third line has m distinct integers $b_1, b_2, \dots, b_m$ ($1 \leq b_i \leq n$), representing that the b_i -th question has a hidden choice.

Output

Output a single integer, representing the expected number of rounds for him to **answer all the questions correctly** for the first time, modulo $10^9 + 7$.

Example

standard input	standard output
2 1 1 3 1	333333338

Note

In the example, there are 2 questions with 1 choice and 3 choices respectively, and question 1 has a hidden choice. Thus Colin cannot answer the first question correctly in the first round as he does not know the existence of the hidden choice for the first round.

The probability of answering all the questions correctly for the first time in the second round is $\frac{2}{3}$.

The probability of answering all the questions correctly for the first time in the third round is $\frac{1}{3}$.

So the answer of the example is $\frac{2}{3} \times 2 + \frac{1}{3} \times 3 = \frac{7}{3}$, which modulo $10^9 + 7$ is 333333338.

Problem M. Metronome for Corgi

Input file: **standard input**
Output file: **standard output**
Time limit: 1 second
Memory limit: 256 megabytes

Azora the corgi loves music but cannot catch the rhythm well, especially when counting beats! After hard research on music theory, Azora found a hidden theorem: There won't be a note crossing the boundary between the beats for modern songs (or for most barking sessions for that matter).



To simplify the problem, we assume that a modern song consists of several sequential notes whose lengths are $l_1, l_2, \dots, l_n$ milliseconds. The notes will be played one by one, which means the song will start playing the first note at the time 0, and the second note at the beginning of the $l_1 + 1$ milliseconds, and the third note at the beginning of the $l_1 + l_2 + 1$ milliseconds, and so on.

If the length of the beat is L milliseconds, then there should not exist any note that is playing but not starting to play at the beginning of the kL milliseconds, for any non-negative integer k . Formally, there should not exist any note i such that $\sum_{j=1}^{i-1} l_j < kL$ but $\sum_{j=1}^i l_j > kL$ for any non-negative integer k .

Now Azora has received a new song from her humans, and she wants to practice counting beats before she barks the song back to them. To challenge herself, Azora wants to find the smallest possible length of a beat, such that it satisfies the theorem. Can you help her?

Input

The first line contains a single integer n ($1 \leq n \leq 2 \times 10^5$), representing the number of notes in the song. The second line contains n integers $l_1, l_2, \dots, l_n$ ($1 \leq l_i \leq 10^9$), representing the length of each note.

Output

Output a single integer, representing the minimum possible length of a beat.

Examples

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

This page is intentionally left blank