

**The 2024 China-ASEAN International Collegiate
Programming Competition &
The 7th Guangxi Collegiate Programming Contest**

Contest Session

May 19, 2024

Problem List

A	Ascending Random Number Sequence
B	Black Friday
C	Cyclic Isomorphism
D	Dune
E	Epsilon Estuary
F	Farewell GXCPC!
G	Grand XOR Counting Problem Challenge
H	Hashing Collision
I	Ice Skating
J	Judge System with Random
K	Kingdom Rush: Vengeance
L	Loong
M	Matching on a Tree

Do not open the problem set before the competition begins!

This problem set should contain 13 problems on 20 numbered pages. Please inform a runner immediately if something is missing from your problem set.

Prepared by Hangzhou Dianzi University.

Problem A. Ascending Random Number Sequence

Input file: **standard input**
Output file: **standard output**
Time limit: 1 second
Memory limit: 128 megabytes

Colin recently received a gift from Eva: - a mysterious ascending random number sequence generator! You only need to input a positive integer n , and the generator will return an ascending sequence to you! After repeated experiments, Colin figured out how the generator works.

Initially, the generator will create a new empty sequence and select a number k from $\{1, 2, \dots, n\}$ with equal probability.

Then, the generator will select a number x from $\{1, 2, \dots, n\}$ with equal probability as the first element of the sequence.

Next, the generator will repeat the following process:

- If the maximum element in the sequence is greater than or equal to k , the generator will stop running immediately and return the current sequence that has already been generated.
- Otherwise, if the maximum element in the sequence is y , then the generator will select a number x from $\{y + 1, y + 2, \dots, n\}$ with equal probability and append x to the end of the sequence.

Now Colin wants to know, for a given n , what the expected length of the sequence returned by the generator is?

Input

A single integer n ($1 \leq n \leq 10^7$).

Output

A single integer, representing the expected length modulo 998244353.

Example

standard input	standard output
3	55458021

Note

- If $k = 1$, the expected length is 1.
- If $k = 2$, the expected length is $1 \times \frac{2}{3} + 2 \times \frac{1}{3} = \frac{4}{3}$
- If $k = 3$, the expected length is $1 \times \frac{1}{3} + 2 \times (\frac{1}{3} + \frac{1}{3} \times \frac{1}{2}) + 3 \times \frac{1}{3} \times \frac{1}{2} = \frac{11}{6}$

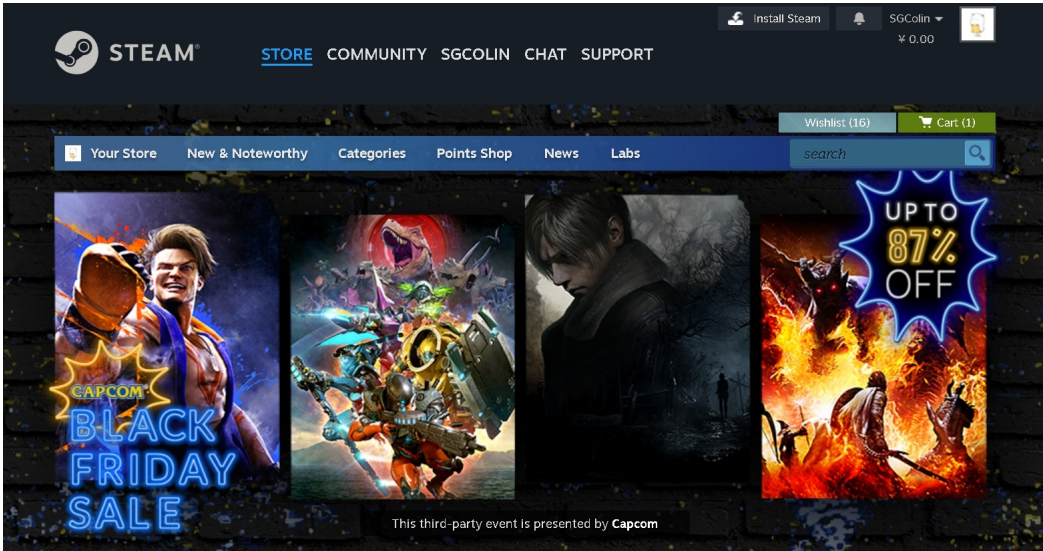
Thus, the answer is $(1 + \frac{4}{3} + \frac{11}{6})/3 = \frac{25}{18}$.

The inverse of 18 modulo 998244353 is 720954255, so the output result is $25 \times 720954255 \bmod 998244353 = 55458021$.

Problem B. Black Friday

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 128 megabytes

Steam’s Black Friday sales are coming! Many of the games that Colin has already put on his wish list are discounted, and he is ready to clear his shopping cart! However, then he remembered that there was no money left in his account. But he couldn’t miss such a good opportunity, so poor Colin could only borrow money from Eva to buy the game.



There are n games on Colin’s wish list. The original price of the i -th game is p_i yuan, and the discounted price is q_i yuan. We define the discount rate of a game as $\frac{p_i - q_i}{p_i}$. Eva naturally cannot let Colin spend money as he pleases, so she only agrees to Colin buying games with discount rates greater than or equal to k . Now Colin wants to know how much money he can borrow at most from Eva.

Input

The first line contains three integers n, a, b ($1 \leq n \leq 10^4, 1 \leq a \leq b \leq 10^4$), representing the number of games on Colin’s wish list, and the lowest discount rate Eva can accept $k = \frac{a}{b}$.
For the following n lines, each line contains two integers p_i, q_i ($1 \leq q_i \leq p_i \leq 10^4$), representing the original price and the discounted price of the i -th game.

Output

A single integer, representing the maximum amount of money Colin can borrow from Eva.

Example

standard input	standard output
3 7 10 42 13 31 9 20 6	15

Note

The second and third games meet Eva’s requirements, so Colin can borrow at most $9 + 6 = 15$ yuan from Eva.

Problem C. Finding Cyclic Isomorphism

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

Colin has been 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 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. Epsilon Estuary

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 128 megabytes

Recently, Colin got addicted to playing a tactics puzzle RPG: *Super Algebrawl*. In the game, Colin transforms into a boar king with magical mathematical powers! He is on a quest for glory, slaying the horrible monsters that plague the animal realm. Every encounter is a puzzle to be solved with logic and simple algebra!

Just now, Colin came to the next battlefield: Epsilon Estuary. The slimes are the rulers here, and they have a disgusting splitting skill - "SPLASH": once it is injured, assuming its remaining health is x , it will immediately split into two slimes with health $\lfloor x/2 \rfloor$ and $\lceil x/2 \rceil$ respectively, and the new slimes can continue to use this skill if they are injured in the future. A slime with health 0 will die immediately.

However, our challenger Colin happens to have the best weapon against them: the "SHOCK" magic scroll! Specifically, every time Colin uses the shock magic scroll, it will inflict y damage to every monster. If a monster's health does not exceed y before Colin uses the magic scroll, it will die immediately after Colin uses the magic scroll.



The image above shows an example: assuming $y = 5$, and there are two slimes on the field with health values $x_1 = 25$, $x_2 = 8$. After Colin used the scroll once, their health became $x'_1 = 20$, $x'_2 = 3$ respectively. Then they split immediately. The first slime split into two slimes with health 10 and 10 respectively; the second slime split into two slimes with health 2 and 1 respectively.

Colin's magic scroll can be used any number of times, so this is destined to be a one-way killing. But clicking the mouse repeatedly is too tiring, so Colin would like you to calculate, given the initial status of the slimes on the field, how many times he needs to use the scroll at least to ensure that all monsters on the field are killed?

Input

The first line contains a single integer T ($1 \leq T \leq 10^5$), representing the number of test cases.

For each test case:

- The first line contains two integers n, y ($1 \leq n \leq 10^5, 1 \leq y \leq 10^{18}$), representing the number of slimes on the battlefield and the damage that can be dealt to each slime by using the magic scroll once.
- The second line contains n integers $x_1, x_2, \dots, x_n$ ($1 \leq x_i \leq 10^{18}$), representing the initial health of each slime on the battlefield.

It's guaranteed that the sum of n over all the test cases will not exceed 10^6 .

Output

Output T lines, the i -th line contains a single integer, representing the answer for the i -th test case.

Example

standard input	standard output
2	3
2 5	7
25 8	
1 1	
100	

Note

For the first test case, the changes in health of the slimes on the field are:

$$\{25, 8\} \rightarrow \{10, 10, 2, 1\} \rightarrow \{3, 2, 3, 2\} \rightarrow \emptyset$$

Problem F. Farewell GXPC!

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 128 megabytes

Perhaps this is the last time Colin is responsible for setting GXPC problems. Looking back on the past four years, setting GXPC problems has always been with Colin in his competitive programming career and is full of rich memories and moving moments. To preserve this wonderful memory, Colin decided to select some representative problems from the problems he had given to the previous GXPCs to form a new contest: the Grand Prix of Guangxi!

Colin has set a total of n problems before, and each problem can be described by two parameters: difficulty d_i and the fun level f_i . Colin will choose some of them to form the contest GP of Guangxi. To make the contest perfect, Colin wants the chosen problems satisfying:

- The sum of their difficulty $\in [L, R]$
- The sum of their fun level is maximized.

Can you help Colin choose which problems should be included in the contest? Since Colin has not determined the specific situation of the final L and R , he will ask for the answer for q pairs of (L_i, R_i) . Please calculate the optimal result for each query.

Input

The first line contains two integers n, q ($1 \leq n \leq 5000, 1 \leq q \leq 10^6$), representing the number of problems and the number of queries.

For the following n lines, the i -th line contains two integers d_i, f_i ($1 \leq d_i \leq 10^5, 1 \leq f_i \leq 10^5$), representing the difficulty and the fun level of the i -th problem.

For the following q lines, each line contains two integers L_i, R_i ($1 \leq L_i \leq R_i \leq 10^5$), representing a query.

Output

Output q lines; the i -th line contains a single integer, representing the maximum possible sum of the fun level. If there is no way to choose problems that meet all requirements, output '-1'.

Example

standard input	standard output
3 3	21
10 17	11
20 4	-1
30 7	
1 30	
41 52	
23 25	

Note

For the first query, Colin can choose the first problem and the second problem, the sum of their difficulty $= 10 + 20 = 30 \in [1, 30]$, and the sum of their fun level $= 17 + 4 = 21$.

For the second query, Colin can choose the second problem and the third problem, the sum of their difficulty $= 20 + 30 = 50 \in [41, 52]$, and the sum of their fun level $= 4 + 7 = 11$.

For the third query, there is no way to choose problems that meet all requirements.

Problem G. Grand XOR Counting Problem Challenge

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

The annual GXPC (Grand XOR Counting Problem Challenge) begins! This year, Judge Colin has prepared a super difficult counting problem for the contestants! Can you slay this ultimate counting problem and win the championship?

Colin has a sequence of n non-negative integers $a_1, a_2, \dots, a_n$.

He wants to know, for each integer $b \in [0, m)$, how many ways there are to choose k distinct indices $1 \leq i_1 < i_2 < i_3 < \dots < i_k \leq n$ satisfying $a_{i_1} \oplus a_{i_2} \oplus \dots \oplus a_{i_k} = b$, where $\oplus$ represents the bitwise XOR operation.

You only need to output the answers modulo 998244353.

Input

The first line contains three integers n, k, m ($1 \leq n \leq 10^6, 1 \leq k \leq n, 1 \leq m \leq 2^{20}$).

The second line contains n integers $a_1, a_2, \dots, a_n$ ($0 \leq a_i < m$), representing the integer sequence.

Output

One line with m integers, where the i -th integer represents the answer modulo 998244353 for $b = i - 1$.

Example

standard input	standard output
4 2 4 0 1 2 3	0 2 2 2

Problem H. Hashing Collision

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

Recently, Colin learned how the string hashing algorithm works. Generally speaking, it is used to convert a string into an integer.

The good and widely used way to define the hash of a string s of length n (indexed from 1) is

$$\text{hash}(s) = \left(s[1] + s[2] \cdot p + s[3] \cdot p^2 + \dots + s[n] \cdot p^{n-1} \right) \bmod m = \left(\sum_{i=0}^{n-1} s[i+1] \cdot p^i \right) \bmod m$$

where p and m are some chosen, positive numbers. It is called a polynomial rolling hash function.

But Colin doesn't understand how to choose appropriate p and m , so he often encounters hashing collision problems. Consider two substrings s_1, s_2 of the string s such that $s_1 \neq s_2$ but $\text{hash}(s_1) = \text{hash}(s_2)$ holds; then we say that there is a hashing collision between s_1 and s_2 .

Now given a string s of length n , and two integers p and m chosen by Colin. He wants to know how many ways there are to choose integers l_1, r_1, l_2, r_2 satisfying $1 \leq l_1 \leq r_1 \leq n, 1 \leq l_2 \leq r_2 \leq n$, and there is a hashing collision between $s[l_1, r_1]$ and $s[l_2, r_2]$

Input

The first line contains three integers n, p, m ($1 \leq n \leq 3000, 1 \leq p, m \leq 2 \times 10^9$).

The second line contains n integers $s[1], s[2], \dots, s[n]$ ($0 \leq s[i] \leq 2 \times 10^9$), the i -th integer represents the value of the i -th character of string s .

Output

A single integer represents the answer.

Examples

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

Note

For the first example, the hashing results of each substring:

$$\text{hash}(s[1, 1]) = 1, \text{hash}(s[2, 2]) = 2$$

$$\text{hash}(s[3, 3]) = 1, \text{hash}(s[4, 4]) = 2$$

$$\text{hash}(s[1, 2]) = (1 + 2 \cdot 2) \bmod 6 = 5$$

$$\text{hash}(s[2, 3]) = (2 + 1 \cdot 2) \bmod 6 = 4$$

$$\text{hash}(s[3, 4]) = (1 + 2 \cdot 2) \bmod 6 = 5$$

$$\text{hash}(s[1, 3]) = (1 + 2 \cdot 2 + 1 \cdot 2^2) \bmod 6 = 3$$

$$\text{hash}(s[2, 4]) = (2 + 1 \cdot 2 + 2 \cdot 2^2) \bmod 6 = 0$$

$$\text{hash}(s[1, 4]) = (1 + 2 \cdot 2 + 1 \cdot 2^2 + 2 \cdot 2^3) \bmod 6 = 1$$

The ways to select parameters in the answer:

- $l_1 = 1, r_1 = 1, l_2 = 1, r_2 = 4$
- $l_1 = 3, r_1 = 3, l_2 = 1, r_2 = 4$
- $l_1 = 1, r_1 = 4, l_2 = 1, r_2 = 1$
- $l_1 = 1, r_1 = 4, l_2 = 3, r_2 = 3$

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

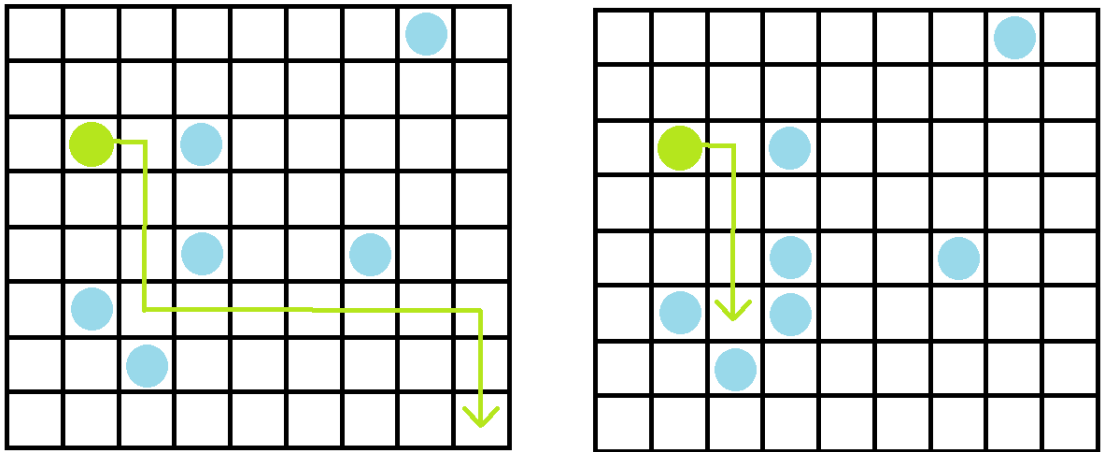
For each query, output a single line with a single integer, representing the answer.

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. Judge System with Random

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 128 megabytes

(The following content is just the background of the problem and has no relationship with the real competition judge system)

Welcome to the GX CPC (Gambling x Competitive Programming Contest)! As a brand new event that combines competitive programming contests and gambling, it has introduced a new judge system - CRJS (Completely Random Judge System).

There is only one problem in this contest. Participants can click the submit button at any time, and the judge system will generate results from {accepted, rejected} randomly. The results will be hidden from all participants (including the participant who made the submission) until the end of the game. One can compete for the championship if at least one result of his submissions is 'accepted'. The penalty for a participant equals **(the number of submissions before his first submission with the result 'accepted') $\times 20$ + the submission time of his first submission with the result 'accepted'**. Among all those who can compete for the championship, the one with the least penalty will be crowned champion. If more than one person's penalty is equal to the minimum penalty of all, then those people will share the honor of champion.

Due to the ridiculous mechanism of the contest and the fact that this was the first year the contest was held, only Colin and Eva were attracted as participants. Colin makes n submissions during the contest, the i th is at the x_i th minute. Eva makes m submissions during the contest, the i th is at the y_i th minute.

Now Colin wants to know, among all 2^{n+m} possible results, how many of them will make Colin win the championship?

Input

The first line contains two integers n, m ($1 \leq n, m \leq 10^5$), representing the number of submissions made by Colin and Eva respectively.

The second line contains n integers $x_1, x_2, \dots, x_n$ ($1 \leq x_1 \leq x_2 \leq \dots \leq x_n \leq 10^9$), representing the time of each submission made by Colin.

The third line contains m integers $y_1, y_2, \dots, y_m$ ($1 \leq y_1 \leq y_2 \leq \dots \leq y_m \leq 10^9$), representing the time of each submission made by Eva.

Output

A single integer, representing the answer modulo 998244353.

Example

standard input	standard output
1 2 40 10 20	2

Note

The table below shows all the possibilities (C for Colin and E for Eva):

The 7th Guangxi Collegiate Programming Contest
Guilin, Guangxi, China, May, 19, 2024

Case	C's Submission	E's first Submission	E's second Submission	C's Penalty	E's Penalty	Champion
1	rejected	rejected	rejected	-	-	-
2	rejected	rejected	accepted	-	40	E
3	rejected	accepted	rejected	-	10	E
4	rejected	accepted	accepted	-	10	E
5	accepted	rejected	rejected	40	-	C
6	accepted	rejected	accepted	40	40	C and E
7	accepted	accepted	rejected	40	10	E
8	accepted	accepted	accepted	40	10	E

Problem K. Kingdom Rush: Vengeance

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

Colin is a huge fan of the Kingdom Rush franchise tower defense games. In the latest generation of Kingdom Rush: Vengeance, Colin's favorite tower is the rotten forest, which can continuously inflict damage to all enemies within its control range.



Colin liked this convenient AOE(area-of-effect) damage tower so much that he decided to only use the rotten forest in the game! In order to practice the strategy of arranging towers, Colin modeled the game content and conducted several numerical experiments.

To simplify the problem, Colin assumes that the path of the map is a number axis. Enemies will be generated from some points and move in the positive direction of the number axis at a constant speed of 1 unit per second. There are n rotten forest towers along the path, where the i -th tower controls the range $[L_i, R_i]$ and will inflict w_i damage to **each enemy within its control range at the end of each second**. Note that the ranges controlled by different towers may overlap. In this case, the damage received by enemies in the overlapping area per second is the sum of the damage of all towers controlling this area. For example, assume that there is a tower that controls the range $[3, 4]$ and another tower controls the range $[4, 5]$. If an enemy is generated at the point 0.5 at time $t = 0.5s$, then it will receive the damage at the end of the third, fourth, and fifth seconds; and at the end of the fourth second, it will receive the damage from both towers.

In this battle m enemies are generated at $t = 0.5s$, and the i -th one is generated at the point $x_i + 0.5$ with health h_i . Once the total damage received by an enemy is greater or equal to its health, it will die immediately. Colin wants to know the exact death time of each enemy, or if the existing towers are not enough to kill this enemy.

Input

The first line contains two integers n, m ($1 \leq n, m \leq 10^5$), representing the number of towers and the number of enemies.

For the following n lines, each line contains three integers l_i, r_i, w_i ($1 \leq l_i \leq r_i \leq 10^9, 1 \leq w_i \leq 10^4$), representing the range controlled by the i -th rotten forest tower and the damage it could make per second.

For the following m lines, each line contains two integers x_i, h_i ($0 \leq x_i \leq 10^9, 1 \leq h_i \leq 10^{18}$), representing the i -th enemy is generated at the point $x_i + 0.5$ with health h_i .

Output

Output m lines, the i -th line contains a single integer, representing the death time of the i -th enemy. If the existing towers are not enough to kill this enemy, output -1.

Example

standard input	standard output
6 6	6
1 3 1	7
2 4 1	2
3 5 1	6
4 6 1	-1
5 7 1	1
6 8 1	
0 15	
0 17	
1 3	
1 15	
5 7	
6 2	

Problem L. Loong

Input file: `standard input`
Output file: `standard output`
Time limit: 1 second
Memory limit: 128 megabytes

The Chinese zodiac is a traditional classification scheme based on the Chinese calendar that **assigns an animal and its reputed attributes to each year in a repeating twelve-year cycle**. In traditional Chinese culture, the Chinese zodiac is very important and exists as a reflection of Chinese philosophy and culture. Chinese folkways hold that one's personality is related to the attributes of their zodiac animal. Originating from China, the zodiac and its variations remain popular in many East Asian and Southeast Asian countries, such as Japan, South Korea, Vietnam, Singapore, Nepal, Bhutan, Cambodia, and Thailand.

Number	1	2	3	4	5	6
Animal	Rat	Ox	Tiger	Rabbit	Loong	Snake
Number	7	8	9	10	11	12
Animal	Horse	Goat	Monkey	Rooster	Dog	Pig

2024 is a year of loong. The “loong”, a creature full of mystery, has long captured the imagination of the Chinese. Chinese people have revered the loong since ancient times, believing the immortal creature brings good fortune, wisdom, and immense power. For thousands of years, the loong has eventually become the spiritual and cultural symbol of the Chinese nation.

Colin loves to research the relationship between the zodiac and years, but he is not good at math. So he turns to you for help! Now, given a year x , Colin wants to know the nearest year of loong after this year.

Input

A single positive integer x ($1 \leq x \leq 9999$)

Output

A single integer, representing the answer.

Examples

<code>standard input</code>	<code>standard output</code>
2022	2024
2024	2036
9999	10004

Problem M. Matching on a Tree

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

Colin recently raised a graph theory problem: MCMMPDNT (Minimum Cost Maximum Matching Problem based on the Depth of Nodes in a Tree).

Given a tree with n nodes rooted at node 1. Define the distance between two nodes as the number of edges on the simple path connecting them. Define the depth of the i -th node as the distance between node i and the root. It's also guaranteed that, except for depth 0, there are an even number of nodes at each depth.

Colin then creates a new undirected graph G with n nodes. If node i and node j ($i \neq j$) have the same depth on the tree, then there will be an undirected edge in G connecting node i and node j , and the weight of that edge is the distance between node i and node j on the tree.

For a graph $G = (V, E)$, a matching M in G is a set of pairwise non-adjacent edges; that is, no two edges share common vertices. A maximum matching is a matching that contains the largest possible number of edges. The cost of a matching M is the sum of the weights of the edges in M . The minimum cost maximum matching is a maximum matching with the minimum cost.

Colin wants to know what the cost of the minimum cost maximum matching in G is.

Input

The first line contains a single integer n ($1 \leq n \leq 10^6$), representing the number of nodes on the tree.

For the following $n - 1$ lines, each line contains two integers u_i, v_i ($1 \leq u_i, v_i \leq n, u_i \neq v_i$), representing an edge of the tree.

It's guaranteed that the input edges form a tree, and except for depth 0, there are an even number of nodes at each depth.

Output

A single integer, representing the minimum cost of a maximum matching in G .

Example

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

Note

One solution for the minimum cost maximum matching of G is $\{(2, 3), (4, 5), (6, 7)\}$, and its cost $= 2 + 2 + 4 = 8$.