Maximum Distance

Before giving you the problem statement, we will define $dist([x_1, x_2, \dots x_k], [y_1, y_2, \dots y_k])$ as the number of indexes i ($1 \le i \le k$) such that $x_i \ne y_i$.

Task

You are given n and two arrays $a_1, a_2, \dots a_n$ and $b_1, b_2, \dots b_n$. Find the maximum $dist$ when choosing any two subarrays of your choice (one from a and one from b) that have the same length.
A subarray of c is a contiguous part of an array c, i. e. the array $c_i, c_{i+1}, \dots c_j$ for some $1 \le i \le j \le n$.

Input

The first line contains an integer t ($1 \le t \le 10^4$) the number of test cases. Then follows the description of the test cases.

The first line of each test case contains an integer n ($1 \le n \le 10\,000$).

The second line of each test case contains n integers $a_1, a_2, \dots a_n$ ($-10^9 \le a_i \le 10^9$).

The third line of each test case contains n integers $b_1, b_2, \dots b_n$ ($-10^9 \le b_i \le 10^9$).

It is guaranteed that the sum of n over all test cases is at most $10\,000$.

Output

For each test case, print a single integer - the maximum $dist$ of two subarrays of equal length, one from array a and the other on from array b.

Examples

stdin

```
7
6
-1 -4 2 -5 -2 10
4 4 -8 2 -4 -8
4
6 2 3 4
2 6 7 4
6
4 -7 -3 10 6 5
4 4 4 4 6 4
6
-4 2 -10 -2 -9 10
-4 -4 2 -2 -9 10
6
12 7 -7 -6 -9 -7
-9 12 -7 12 -9 12
4
1 2 3 4
5 6 7 4
4
1 2 3 4
1 2 3 5
```

stdout

```
6
3
5
5
5
3
3
```

The floor is lava!

You finally trapped Lavutz and his $K - 1$ friends in a room (in total there are K people). The room is very interesting, as it is split in n rows and m columns, and for each row i and column j ($1 \leq i \leq n$ and $1 \leq j \leq m$) there is a cell which is at $a_{i,j}$ units of height (distance in units from the ground). A person can move from cell (i, j) to one of the cells ($i - 1$, j), ($i + 1$, j), (i, $j - 1$), (i, $j + 1$) (if that doesn't go out of the boundaries of the room) in exactly 1 second, or they can just stay in cell (i, j).

You have a device which can raise the level of the lava, which is initially raised at level 0, but you don't want to hurt Lavutz or his friends. A person is hurt if the lava is raised at a level greater than the height of the cell where the person is at that moment.

Task

Now, you want to find for each L from 1 to $n \cdot m$ what is the minimum time you need to wait such that you can raise the lava to level L and nobody gets hurt, or -1 if for any amount of time you will wait, you can't raise the lava to level L without hurting anybody.

Input data

On the first line there are 3 integers: n, m ($1 \leq n, m \leq 700$)- the number of rows and columns and K ($1 \leq K \leq 10\ 000$) - the number of people in the room.

On the next n lines, there will be $a_{i,j}$ ($1 \leq a_{i,j} \leq n \cdot m$) the height of cell ($i$, j). ($1 \leq i \leq n$ and $1 \leq j \leq m$).

For each of the next K lines there will be two integers x, y ($1 \leq x \leq n$ and $1 \leq y \leq m$) - the starting position of the K people.

Output data

You will print $n \cdot m$ integers. The i-th integer is the answer for level i.

Constraints and clarifications

- Multiple people can move at the same time.
- Multiple people can be in the same cell at the same time.
- You can raise the lava to level 1 without hurting anybody (each person is at a height of at least 1).

Example

```
stdin
```

```
6 5 6
30 14 11 22 16
7 5 6 3 23
20 1 5 2 17
21 1 4 2 6
17 7 3 24 3
26 25 13 14 10
2 2
3 2
4 4
5 5
2 4
4 3
```

```
stdout
```

```
0 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 4 5 8 8 8 8
```

Explanation

If you want to raise the lava to level 2, you need to wait 1 second for the person in cell (3, 2).
If you want to raise the lava to level 6, you need to wait 2 seconds for the persons in cell (4, 3).

Basketball

Andrei is taking basketball throws. He wants to score n points.

As he doesn't have a lot of time, he wants to reach n points from as few throws as possible. As you know, in basketball we can score using throws which get either 2 or 3 points.

If Andrei can't reach n points, we should print `No` (exactly as it is written). Otherwise, we will print two integers a and b, which represent the minimum number of two point, respectively three point throws Andrei is going to use.

Input data

The first line will contain a single integer, n ($1 \leq n \leq 10^9$), representing the amount of points Andrei scored.

Output data

The first and the only line of the output will either contain two integers, a and b, or otherwise the message `No` if he cannot reach n points.

Example 1

stdin

```
11
```

stdout

```
1 3
```

Example 2

```
stdin
```

```
33
```

```
stdout
```

```
0 11
```

Edenland

Alice and Bob go to edenland. Edenland is a famous adventure park where people can play various games which take place inside of a forest.

At edenland, you have numerous tracks, each consisting of several games. We will only focus on one track. This track consists of n games, separated by platforms. Alice takes a_i time to finish the i^{th} game, while Bob takes b_i time. We consider that it takes no time to pass a platform. Alice goes on the track first, followed by Bob.

However, at edenland there is a special rule: you can't overtake the person in front of you, that is, if you start behind someone, you will always be behind that person. To complete a track, you first start by playing the first game, then the second, and so on. As Bob will always be behind Alice, this rule will not change, and Alice will start the track before Bob.

There is another very important rule at edenland: no two people can play the same game at the same time, as it is not safe. Thus, if Bob is on the platform before the i^{th} game, he will wait on that platform until Alice has finished the i^{th} game.

Now, you are curious, for q intervals $[l, r]$, what is the amount of time Bob will finish the track after Alice, if we consider only the track consisting of games $l, l + 1, \ldots, r$.

Input data

In the first line of input, you will read integer n ($1 \leq n \leq 2 \cdot 10^5$), representing the total number of games.

In the second lines of input, you will read n integers $a_1, a_2, \ldots, a_n$. ($1 \leq a_i \leq 10^9$)

In the third line of input, you will read n integers $b_1, b_2, \ldots, b_n$. ($1 \leq b_i \leq 10^9$)

In the fourth line of input, you will read one integer q, the number of queries ($1 \leq q \leq 2 \cdot 10^5$).

Every one of the next q lines contains two integers l and r. ($1 \leq l, r \leq n$)

Output data

You should output q space-separated integers, the i^{th} of them representing the answer for the i^{th} query.

Example 1

```
stdin
```

```
5
4 9 4 5 3
10 5 2 9 8
3
1 5
4 4
1 2
```

```
stdout
```

```
14
9
6
```

Explanation

In the second query of the first test, Alice will complete game 4 in 5 seconds, then Bob will start it and complete it 9 seconds later.

In the third query of the first test, Bob will start the first game when Alice finishes it. Alice will finish the second game 9 seconds later, while Bob will finish the first game 10 seconds later. Then, Bob will finish the second game another 5 seconds later, so he will end 6 seconds later than Alice.

The explanation for the rest of the examples is truly remarkable, but we consider the explanations for the second and third queries of the first test sufficient.

Example 2

```
stdin
```

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

```
stdout
```

```
9
4
2
2
7
```

Fiboxor

Consider the following integer sequence f:

$f_1 = f_2 = 1$, $f_k = |f_{k-1} - f_{k-2}| \oplus (f_{k-1} + f_{k-2})$, for $k \geq 3$.

Now, you are wondering, for q triples of integers (l, r, M), what is the sum of all f_i, $l \leq i \leq r$, modulo M. That is, find $(\sum\limits_{i=l}^{r} f_i)$ and print it's remainder modulo M.

By $|x|$ we denote the absolute value of x and by XOR we denote the bitwise XOR operation.

Input data

In the first line of input, you will read one integer q ($1 \leq q \leq 2 \cdot 10^5$).

In the next q lines, you will read three integers l, r ($1 \leq l \leq r \leq 10^9$) and M ($10^8 < M < 10^9$, M is prime).

Output data

Output q lines, the i^{th} of them representing the answer for the i^{th} triple (l, r, M).

Example

stdin

```
4
1 3 998244353
2 4 998244353
4232324 12345678 998244353
92345678 998244353 998244353
```

stdout

```
4
5
652671816
367684397
```

Explanation

The first four values of f are: $f_1 = 1, f_2 = 1, f_3 = 2, f_4 = 2$.

Thus, the answer for the first question is $1 + 1 + 2 = 4$ and the answer for the second question is $1 + 2 + 2 = 5$.

Suceava

Suceava city consists of N neighborhoods, indexed from 1 to N, connected by $N-1$ bidirectional streets. It is widely recognized that you can travel between any two neighborhoods by using one or more streets.

Suceava is home to M gangs, all of which are in conflict and indexed from 1 to M. Initially, each street is occupied by a gang.

Over the course of T days, some gang may conquer a street occupied by another gang.

We define the insecurity level of a gang as the maximum number of streets you can traverse that are occupied by the gang, while ensuring each street is traversed only once.

Being given Q queries (g, t), you need to determine the insecurity level of gang g on day t.

Input

The first line contains two integers N, M $(2 \leq N \leq 10^5, 2 \leq M \leq 10^5)$, the number of neighborhoods and the number of gangs. The next $N-1$ lines contain three integers u, v, g $(1 \leq u \neq v \leq N, 1 \leq g \leq M)$—the street connecting neighborhoods u and v, occupied by clan g.

The following line contains integer T $(1 \leq T \leq 10^5)$, the number of days. The next T lines contains three integers u, v, g $(1 \leq u \neq v \leq N, 1 \leq g \leq M)$—the street connecting neighborhoods u and v that is being conquered by gang g.

The following line contains integer Q $(1 \leq Q \leq 10^5)$, the number of queries. The next Q lines contains two integers g, t $(1 \leq g \leq M, 1 \leq t \leq T)$—describing the query.

Output

For each query print the answer on a single line.

Example 1

stdin

```
6 3
4 3 2
1 2 1
3 2 2
5 2 1
2 6 1
4
6 2 2
2 5 2
2 3 1
6 2 1
4
2 3
1 2
1 4
3 2
```

stdout

```
2
1
2
0
```

Example 2

stdin

```
8 3
1 2 2
```

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

stdout

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

Minimize Sum

Sumin likes deques, so his teacher gave him a cool deque game. The game works based on an array a, and is played as follows:

1. There will initially be a variable $T = 0$ and a deque containing just the element 0.
2. Sumin will go through the array elements, starting with the 1st one and ending at the nth one. At the i^{th} element Gigel must choose one of two ways to progress the game:
    - add to T the first element of the deque, then put a_i at the front of the deque (that is $T := T + deque.front(),\ deque.push\ front(a_i)$)
    - add to T the last element of the deque, then put a_i at the back of the deque (that is $T := T + deque.back(),\ deque.push\ back(a_i)$)

After the n^{th} element is processed, the score of the game will be equal to the current value of T. Grigoutz wants to minimize this final score T but he is unsure if he received the minimum score that is possible. Can you help him calculate it?

Task

Sumin asks you to write a program which given n and the number of minutes to write each of the essays will print the minimum value of T after the n operations.

Input

The first line contains an integer t ($1 \leq t \leq 10^4$) the number of test cases. Then follows the description of the test cases.

The first line contains a single integer n ($1 \leq n \leq 2 \cdot 10^5$) the length of the array.

The second line of each test case contains n integers $a_1, a_2, \ldots a_n$ ($1 \leq a_i \leq 10^9$) - the elements of the array.

It is guaranteed that the sum of the values of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, you have to print a single integer - the minimum value of T

Examples

stdin

```
3
2
1 2
5
9 3 6 5 9
8
5 6 4 8 3 1 0 10
```

stdout

```
0
14
19
```

Example description

For the first test case, Sumin will go for the first choice, resulting in $T = 0$ and the deque containing $1\ 0$. After this, he will go with the second choice, adding 0 to T, and the deque contains $1\ 0\ 2$ in this order.
For the second test case, Sumin will go for the first choice, resulting in $T = 0$ and the

deque containing $9\ 0$. After this, he will go with the second choice, adding 0 to T, and the deque contains $9\ 0\ 3$ in this order. Third choice is for the back, $T = 3$ and deque is $9\ 0\ 3\ 6$. Next is the back once again, resulting in $T = 9$ and the deque is $9\ 0\ 3\ 6\ 5$. Final choice is still on the back side of the deque, resulting in $T = 14$ and the deque being $9\ 0\ 3\ 6\ 5\ 9$.

AI Thoughts

As you all know, AI use is on the rise. As our friend Gigel dislikes this, he came up with a plan of sabotaging the growth of AI.

An AI brain works in intricate ways. First lets consider an infinite grid where there are n neurons placed. We know for each neuron its position in the grid and its color, given as a tuple (x, y, col). An AI Thought is described by:

- The length of the neural sequence m, meaning that our thought is created by a list of m neurons.
- The colors we want to use for our list of neurons, $col_1, col_2, \dots\ col_m$. This means that the i^{th} neuron in our thought list should have the color col_i. As an important observation, we are allowed to use the same neuron **multiple times** in our list.
- The time we need to generate this thought is $manh(i_1, i_2) + manh(i_2, i_3) + \dots + manh(i_{m-1}, i_m)$, where $manh(i, j)$ is the Manhattan distance between points (neurons) i and j in the grid($|x_i - x_j| + |y_i - y_j|$).

Now, depending on how we choose the neurons, the time needed for generating a thought can differ. As Gigel hates AI, he wants to prove the world how inefficient it is, so he asks for your help. In order to do this, you need to compute for T thoughts the maximum possible amount of time for generating each thought.

Input data

The first line of the input contains n, the number of neurons ($1 \le n \le 2 \cdot 10^5$).

The next n lines of the input contain the neurons we can use for our thoughts ($-10^9 \le x_i, y_i \le 10^9$), ($1 \le col \le 2 \cdot 10^5$).

The next line contains t, the number of thoughts we need to analyze ($1 \le t \le 2 \cdot 10^5$).

The next t lines contain m_i, the number of points in the thought $(1 \le m_i \le 2 \cdot 10^5)$ and then m_i inteegers representing the colors we need to use for the neurons in this thought ($1 \le col_i \le 2 \cdot 10^5$).

It is guaranteed that the sum of values m_i is at most $5 \cdot 10^5$ and each color col_i used is present in at least one neuron from the set of neurons.

Output data

The output will contain t lines, on each line we have the answer for the i^{th} question.

Example

stdin

```
8
1 1 3
-1 4 2
3 2 4
4 1 1
-2 -3 4
-1 2 1
2 2 3
3 0 2
2
5
1 2 1 2 4
3
3 1 2
```

stdout

```
32
11
```

Explanation

For the first thought, we can use the fourth, the second, the fourth, the second and the fifth neuron, in this order.

For the second thought, we can use the first, the fourth and the second neuron, in this order.

KSumT

Tutz likes math problems; this time he is trying to solve the following problem:

You are given K, S, T and are asked how many sequences of K integers (more formally $a_1, a_2, a_3, \cdots, a_K$) follow these rules:

- $a_i > 0$, for any i ($1 \leq i \leq K$)
- $a_1 + a_2 + a_3 + \cdots + a_K = S $
- All subarrays of length T have the same product. More formally, $a_1 \cdot a_2 \cdot a_3 \cdot \ ... \ \cdot a_T = a_2 \cdot a_3 \cdot a_4 \cdot \ ... \ \cdot a_{T+1} = \cdots = a_{K-T+1} \cdot a_{K-T+2} \cdot a_{K-T+3} \cdot \ ... \ \cdot a_K$

You should find the number of such arrays modulo $10^9 + 7$ ($10^9 + 7$ is a prime number).

Input data

The first line contains 3 integers K, S, T ($1 \leq K, S, T \leq 5 \cdot 10^6$).

Output data

Output one integer — the number of possible sequences modulo $10^9 + 7$

Example 1

```
stdin
```

```
5 13 3
```

```
stdout
```

```
15
```

Explanation

The 15 sequences in the first example are:
$[1,1,9,1,1]$, $[1,2,7,1,2]$, $[1,3,5,1,3]$, $[1,4,3,1,4]$, $[1,5,1,1,5]$,
$[2,1,7,2,1]$, $[2,2,5,2,2]$, $[2,3,3,2,3]$, $[2,4,1,2,4]$, $[3,1,5,3,1]$,
$[3,2,3,3,2]$, $[3,3,1,3,3]$, $[4,1,3,4,1]$, $[4,2,1,4,2]$, $[5,1,1,5,1]$

Example 2

```
stdin
```

```
15 44 9
```

```
stdout
```

```
1162800
```

Parallelogram

Bob has n sticks. The i-th stick has length a_i. He needs to choose a tuple of 4 sticks (i, j, k, p) such that:

- $1 \leq i < j < k < p \leq n$;
- by moving and rotating the sticks any way you want (without breaking them) you can obtain a parallelogram with sides of lengths a_i, a_j, a_k, a_p.

Task

Bob is interested if there is such a tuple. He asks you to print `YES` if you can choose 4 sticks satisfying the above properties, and `NO` otherwise.

Input

The first line contains an integer t ($1 \leq t \leq 10^4$) the number of test cases. Then follows the description of each test case.

The first line of each test case contains a single integer n ($1 \leq n \leq 2 \cdot 10^5$) the number of sticks.

The second line of each test case contains n integers $a_1, a_2, \ldots a_n$ ($1 \leq a_i \leq n$) - the length of the sticks.

It is guaranteed that the sum of the values of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, you have to print the answer to Bob's question.

Examples

```
stdin
```

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

```
stdout
```

```
YES
NO
YES
```

Blabla

After hearing enough nonsense discussions, X decided to be concise with the next problem’s statement.

Given an array of length n, find out how many subarrays exist such that the difference between the maximum and minimum values in it is at least equal to the difference between the first and the last position of the subarray.

More formally, if we have the subarray $[L, R]$, whose maximum value is A and whose minimum value is B, we want $A - B \geq R - L$.

Input data

The first line of the input contains n ($1 \leq n \leq 2 \cdot 10^5$), the length of the array.

The next line of the input contains the array v of length n, where v_i ($1 \leq v_i \leq 2 \cdot 10^5$) is the i_{th} value of the array.

Output data

The output will contain a single integer, representing the number of subarrays found.

Example 1

stdin

```
7
1 2 3 1 1 3 2
```

```
stdout
```

```
17
```

Example 2

```
stdin
```

```
12
4 2 3 1 2 3 4 5 6 4 6 1
```

```
stdout
```

```
43
```

Dush

Sannu went on a trip to the mountains together with his friend group. Altogether there were N people gathered in a cabin in order to watch all of One Piece, and they decided to go eat at a nearby restaurant on the third day of their trip. Although their rented cabin had enough bathrooms, only one shower was usable. From Sannu's keen observations during the second day, they discovered that that shower has running water just at some moments during the day, and you cannot control the temperature of the water. Sometimes there is hot water, sometimes lukewarm, sometimes only cold water and during the rest of the day there is no water running from the shower. Knowing the preferred water temperature for each person, as well as how much each person stays in the shower, please find out what is the earliest moment of the day in which everyone has showered (and is ready to leave for the restaurant). Do note that, due to the size of the shower cabin, only one person can take a shower at a given time, and since anyone would be annoyed if they had to interrupt their shower, each person must get into the shower exactly once (in only one showering timeslot).

Input data

The first line of the input contains two integers N and M ($1 \leq N \leq 20$ and $1 \leq M \leq 10^5$) – the number of people at the cabin and the number of timeslots where there is water running in the shower.

The next N lines contain two integers X_i and Y_i ($-1 \leq X_i \leq 1$ and $1 \leq Y_i \leq 10^9$) – the type of water preferred by the i-th person, as well as the time needed for the i^{th} person's shower.

The next M lines each contain three numbers S_i, D_i and T_i ($1 \leq S_i, D_i \leq 10^9$, $-1 \leq T_i \leq 1$) – starting time, the time duration in which there is water and the water type of the i^{th} timeslot with water. Furthermore, these timeslots are not overlapping ($S_i + D_i \leq S_{i+1}$) for all $1 \leq i \leq M$.

Output data

On the first line there will should one integer that represents the minimum time of the day at which everyone is showered.

Example 1

```
stdin
```

```
3 5
-1 5
1 7
1 3
2 2 -1
5 5 -1
20 9 1
40 10 1
60 20 1
```

```
stdout
```

```
43
```

Explanation

In the first case, the first person will take a shower in the second timeslot, the second person in the third timeslot, and the third one in the fourth timeslot, finishing his shower at time 43.

Example 2

```
stdin
```

```
3 5
-1 5
```

```
1 7
1 3
2 2 -1
5 5 -1
20 10 1
40 10 1
60 20 1
```

stdout

```
30
```

Explanation

In the second case, the first person will take a shower in the second timeslot, the second person in the third timeslot, and the third one in the third timeslot after the second one, finishing his shower at time 30.

Dragons

You are playing a game in which you must defend your village from a dragon.

The village can be represented as a tree (a connected acyclic graph) with N nodes, indexed form 1 to N, each node representing fortifications of varying heights. The height of fortification i is denoted as h_i.

The dragon has a power level of P and starts flying at the base of fortification u (i. e. its initial height is 0). Its goal is to attack fortification v. It flies along the path from u to v and when it encounters a fortification x with a height h_x greater than or equal to its current height, it loses h_x points from its power and continues flying at a height of h_x. Unfortunately, there's a bug in the game: if the dragon's current power P_{crt} becomes less than 0, it immediately becomes $-P_{crt}$. You have the ability to shuffle the fortifications along the path from u to v. Your objective is to find an arrangement such that the dragon's power is reduced to 0 when it reaches fortification v, preventing the dragon from attacking it.
Being given Q scenarios (P, u, v), you should find if it's possible to shuffle the fortifications along the path from u from v so the dragon won't attack the tower v.

Input

The first line contains one integer N ($2 \leq N \leq 10^4$), the number of nodes. The second line contains N integers $h_1, h_2, \ldots, h_N$ ($1 \leq h_i \leq 10^3$).
Each of the following $N - 1$ lines contains two integers u and v, meaning that there is an edge between u and v.
The next line contains one integer Q ($1 \leq Q \leq 10^4$), the number of scenarios. Each of the next Q lines contains three integers P ($1 \leq P \leq 10^3$), u and v, describing a scenario.

Output

For each scenario, output “YES” if it’s possible to shuffle the fortifications so the dragon won’t attack, or “NO” otherwise.

Example

stdin

```
9
1 2 3 4 5 6 7 8 9
1 2
1 3
1 4
2 5
3 6
3 7
4 8
6 9
3
5 7 13
5 7 1
9 8 12
```

stdout

```
YES
NO
YES
```