Problem A. Shaking Trees

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

One day, Little Z met a towering tree on the road, which blocked his path. With no other choice, he had to figure out a way to remove the entire tree to continue forward. Little Z, being immensely strong, could shake a node of the tree continuously until it fell off, becoming a small tree, while all the leaves of the small tree would drift to the ground and disappear.

Now, Little Z wants to know how many shakes he needs to make all nodes of the tree fall to the ground. But this is too easy for him. He also finds that the tree is not very tall. So, please also tell Little Z how many ways he can achieve this with the minimum number of shakes.

Formally, there is a rooted tree containing n nodes, numbered from 1 to n. The root of the tree is node 1, and **the height of the tree does not exceed** 100. In each operation, Little Z can choose a node u, cut u from its parent (if it has one) to become the root of a new tree, and then delete all leaf nodes in the tree rooted at u.

Now, Little Z wants to know the minimum number of operations required to delete all nodes. You also need to answer the number of ways to delete all nodes with the minimum number of operations after modulo $10^9 + 7$.

Two operation schemes are considered the same if and only if they have the same number of operations and the node numbers operated on in each operation are the same.

Input

The first line contains an integer n ($1 \le n \le 2 \times 10^5$), representing the number of nodes in the tree.

The following $n - 1$ lines, each contain two integers u, v ($1 \le u, v \le n$), representing an edge on the tree.

It is guaranteed that the height of the tree does not exceed 100.

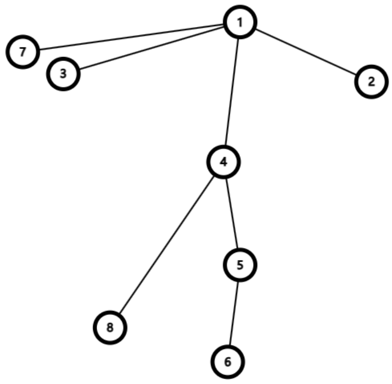
Output

Output one line with two integers, representing the minimum number of operations and the number of schemes modulo $10^9 + 7$.

Example

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

Note

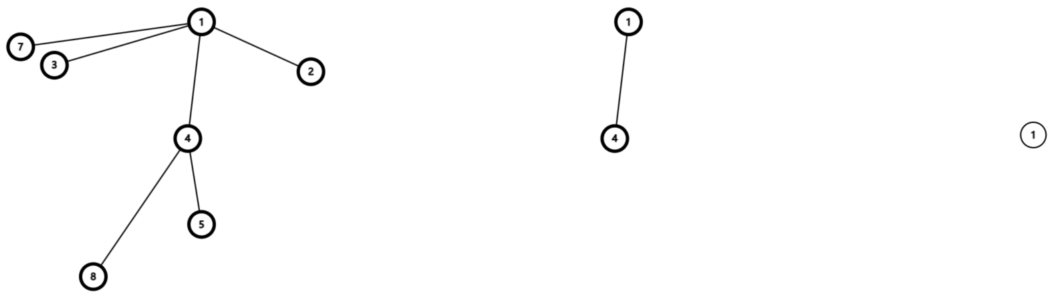


The tree in the example

In the example, it takes at least 4 operations to delete all nodes, and there are a total of 8 ways to delete all nodes with 4 operations.

These solutions are: $[6, 1, 4, 1], [6, 1, 1, 1], [1, 5, 4, 1], [1, 5, 1, 1], [1, 4, 1, 4], [1, 1, 1, 1], [1, 4, 4, 1], [1, 1, 4, 1]$

In the solution $[6, 1, 4, 1]$, all nodes are deleted as follows:



After selecting node 1, all nodes are deleted.

Problem B. Countless Me

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

I am Fluttershy, a plain and ordinary Pegasus pony of Equestria—well, after going through countless experiences with Twilight Sparkle and other friends, maybe I'm not that ordinary.

Like today, when I returned home, what caught my eye was countless versions of myself.

Luckily, I'm not Pinkie Pie; so instead of bouncing around chattering endlessly, they just quietly looked at me and gave me a question:

Given an array of non-negative integers a_i with length n, I may perform the following operation no more than n times:

- Choose i, j such that $1 \le i, j \le n$, $0 \le x \le 10^9$, then set $a_i \leftarrow a_i + x$ and $a_j \leftarrow a_j - x$

I must ensure that all a_i remain non-negative integers after each operation, reminded by one of us.

My goal is to minimize the value of $a_1 \mid a_2 \mid \dots \mid a_n$ after the operations. Here, $\mid$ denotes the bitwise OR operator, and $x \mid y$ represents the bitwise OR of two non-negative integers x and y.

The bitwise OR operation compares two binary numbers bit by bit. In each bit position, if at least one of the bits is 1, then the result is 1; otherwise, it is 0. For example, the bitwise OR result of the binary numbers $10 = 1010_2$ and $12 = 1100_2$ is $14 = 1110_2$, and the result of $3 = 0011_2$ and $5 = 0101_2$ is $7 = 0111_2$.

Input

The first line contains a single integer n ($1 \le n \le 2 \times 10^5$), representing the length of the array.

The second line contains n non-negative integers a_i ($0 \le a_i \le 10^9$).

Output

A single line containing one integer representing the answer, the minimized $a_1 \mid a_2 \mid \dots \mid a_n$ value.

Example

standard input	standard output
7 1 9 1 9 8 1 0	5

Note

In the example, one feasible final result is $[5, 5, 5, 5, 5, 4, 0]$, which has a minimal result equal to 5. It can be proven that there is no better result.

One feasible method to achieve this result is $[1, 9, 1, 9, 8, 1, 0] \rightarrow [5, 5, 1, 9, 8, 1, 0] \rightarrow [5, 5, 5, 5, 8, 1, 0] \rightarrow [5, 5, 5, 5, 5, 4, 0]$, requiring a total of 3 steps.

Problem C. TreeBag and LIS

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

TreeBag enjoys participating in the contests on the famous international competitive programming website CF (ColaFries) in his dreams, and every time he can achieve a single-digit rank in his dreams. One day, he suddenly became curious about the longest increasing subsequence of the array formed by his ranks after each competition. Therefore, he came up with the following question:

Given an integer x, construct a string s consisting only of decimal digits(i.e. `0,1,2,...,8,9`) and of length not exceeding 10^5, such that the sum of the values of all longest strictly increasing subsequences of s equals x.

Since TreeBag is BAKA when he wakes up from his dreams, he asks you to help him solve this problem.

The value of a string composed only of decimal digits is defined as the value of the decimal number it represents. For example, the sum of all longest strictly increasing subsequences of the string `001243` is $0124 + 0123 + 0124 + 0123 = 494$.

A string t is a subsequence of s if and only if t can be obtained by deleting several characters (possibly 0 characters) from s without changing the order of the remaining characters. For example, the subsequence of `1011101` can be `0`, `1`, `11111`, `101`, `0111`, but not `000`, `101010`, `11100`.

A string t is a strictly increasing subsequence of s if and only if t is a subsequence of s and for any $1 \le i < |t|$ ($|t|$ represents the length of t), it holds that $t_i < t_{i+1}$. Here, characters are compared by their numeric value.

A string t is the longest strictly increasing subsequence of s if and only if t is a strictly increasing subsequence of s and there is no strictly increasing subsequence of s longer than t.

Input

A single line containing one integer x ($0 \le x \le 10^{13}$), representing the sum of the values of all longest strictly increasing subsequences of the string you need to construct.

Output

Output a string s of length not exceeding 10^5, such that the sum of the values of all its longest strictly increasing subsequences equals x.

It can be shown that there is at least one answer that satisfies the requirements.

If there are multiple answers that satisfy the requirements, output any one of them.

Examples

standard input	standard output
494	001243
0	0

Note

In the first example, the sum of the values of all longest strictly increasing subsequences of the string `001243` is $0124 + 0123 + 0124 + 0123 = 494$.

Problem D. ICPC

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

Mystia's Izakaya has held a new edition of the International Championship of Portion Consumption (ICPC). Although Mystia did not allocate spots for Whitejade House University (WHU) for destroying the venue in the previous year, the university still sent a team to participate, led by Yuyuko.

On a long table, there are n seats arranged in a row, with the i-th seat from left to right having a dish with a portion size of a_i. Yuyuko starts at the s-th seat, and at the end of each second, she can move to any seat adjacent to her current seat (she can also choose to stay at her current seat). Any dish on a seat she reaches will be consumed by her.

Yuyuko wants to know the maximum total portion size of the dishes she can consume. You need to calculate the maximum total portion size of the dishes Yuyuko can consume if she starts from the s-th seat and moves for t seconds, for all positive integers s, t that satisfy $1 \le s \le n, 1 \le t \le 2n$.

Input

The first line contains one integer n ($1 \le n \le 5000$).

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^9$).

Output

Let $F_{i,j}$ represent the maximum total portion size of the dishes that Yuyuko can consume starting from seat i and moving for j seconds. To verify that you have calculated values of all $F_{i,j}$ for $1 \le i \le n, 1 \le j \le 2n$, please output an integer $\bigoplus_{i=1}^{n} \left(i + \bigoplus_{j=1}^{2n} j \cdot F_{i,j} \right)$.

Here, $\bigoplus$ denotes the bitwise XOR operation. Specifically, if $a \oplus b$ represents the bitwise XOR of two non-negative integers a and b, then $\bigoplus_{i=1}^{n} x_i = x_1 \oplus x_2 \oplus \ldots \oplus x_n$ (where $x_1, x_2, \ldots, x_n$ are non-negative integers).

This output method is only to reduce the amount of program output and is unrelated to the solution of the problem.

Examples

standard input	standard output
3 1 2 3	61
6 7 2 1 3 0 8	59

Note

In the first example, values of $F_{i,j}$ are

$$F = \begin{bmatrix} 3 & 6 & 6 & 6 & 6 & 6 \\ 5 & 5 & 6 & 6 & 6 & 6 \\ 5 & 6 & 6 & 6 & 6 & 6 \end{bmatrix}$$

In the second example, values of $F_{i,j}$ are

$$F = \begin{bmatrix} 9 & 10 & 13 & 13 & 21 & 21 & 21 & 21 & 21 & 21 & 21 & 21 \\ 9 & 9 & 10 & 14 & 14 & 21 & 21 & 21 & 21 & 21 & 21 & 21 \\ 4 & 10 & 12 & 13 & 14 & 14 & 21 & 21 & 21 & 21 & 21 & 21 \\ 4 & 11 & 13 & 13 & 13 & 14 & 21 & 21 & 21 & 21 & 21 & 21 \\ 8 & 8 & 11 & 13 & 14 & 21 & 21 & 21 & 21 & 21 & 21 & 21 \\ 8 & 11 & 12 & 14 & 21 & 21 & 21 & 21 & 21 & 21 & 21 & 21 \end{bmatrix}$$

Problem E. Boomerang

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

I am Fluttershy, and I've encountered countless versions of myself. Even though they are not noisy at all, I still take them with me to search for my friends. I must figure this out.

In Cloudsdale, I found my friend, Rainbow Dash. She is a handsome Pegasus pony, completely different from me. Unfortunately, Rainbow Dash also encountered a problem. She told me:

The organization WHU (World Horse Unite) often spreads some news in the news, but unfortunately, some news, after a period of dissemination, is confirmed to be fake news. In order to save face, WHU has to spread another news to refute it.

"And they always ask me to refute it,"Rainbow said loudly, "but I'm a little confused."

After organizing Rainbow's description, for simplicity, I assume that the message is transmitted on a tree with n nodes and $n-1$ edges.

The fake news spread by WHU starts from node r at time 0, and spreads to neighboring nodes every unit of time. Formally, at time t, all nodes within distance t from r receive this message, that is, the set of nodes $V(r,t) = \{v|\mathrm{dis}(r,v) \le t\}$, where $\mathrm{dis}(u,v)$ denotes the number of edges of the unique simple path between two nodes u and v on the tree.

At time t_0, WHU will commission Rainbow to choose a new node r' to refute the news, and every unit of time, the refutation will spread to nodes within distance no more than k from the current node, i.e., the rate of refutation is k times faster than that of fake news. Formally, at time t ($t \ge t_0$), all nodes within distance $k(t-t_0)$ from r' receive the refutation, that is, the set of nodes $V'(r',t) = \{v|\mathrm{dis}(r',v) \le k(t-t_0)\}$.

Now r and t_0 are determined, but Rainbow is not sure where to start refuting, nor how fast to refute. Therefore, I need to answer for all $1 \le k \le n$, when is the earliest time that fake news can be refuted when taking any r'. Formally, find the earliest time t such that the refutation covers all the nodes covered by the fake news, i.e., $V(r,t) \subseteq V'(r',t)$.

Input

The first line contains an integer n ($1 \le n \le 2 \times 10^5$), indicating the number of nodes in the tree.

The next $n-1$ lines each contain two integers u and v, representing an edge in the tree.

The following line contains two integers r $(1 \le r \le n)$ and t_0 $(1 \le t_0 \le n)$, representing the parameters as described above.

Output

A single line containing n integers, where the i-th integer represents the earliest time value for the above inquiry when $k = i$.

Examples

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

Note

In the first sample, the refutation is two units of time later than the fake news, and the fake news will cover all the nodes at $t = 4$. When $k = 1$, only $r' = 3$ can be chosen to cover all the nodes when $t = 4$; when $k = 2$, $r' = 2, 3$ can be chosen, and when $t = 3$, the fake news covers the nodes $\{1, 2, 3, 4\}$, and the refutation covers the nodes $\{1, 2, 3, 4\}$ and $\{1, 2, 3, 4, 5\}$, respectively, which satisfy the complete coverage; when $k = 3$, $r' = 2, 3, 4$ can be chosen, and all nodes are covered by the refutation when $t = 3$; when $k = 4$, any r', all nodes at time 3 are always covered.

In the second example, for $k = 1, 2, \ldots, n$, a feasible choice for r' is $1, 2, \ldots, 2$.

Problem F. Custom-Made Clothes

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

This is an interactive problem.

"Oh, poor Opalescence,"I sighed as I watched Fluttershies surround the white Persian cat.

Rarity is always so busy, and honestly, she probably struggles to solve my problems. She is not really adept at this kind of magic.

While she was busy making customized clothing, she even turned around and gave me a problem. Her problem can be abstracted as follows:

There is an $n \times n$ matrix of positive integers $a_{i,j}$ ($1 \leq a_{i,j} \leq n \times n$), which satisfies that for each $i > 1$, $a_{i,j} \geq a_{i-1,j}$; and for each $j > 1$, $a_{i,j} \geq a_{i,j-1}$.

You're provided with only the size n of the matrix, without knowledge of its element values $a_{i,j}$. However, you can inquire whether a given position $a_{i,j}$ in the matrix is not greater than a specified integer x.

You need to find the k-th largest value in $a_{i,j}$ using no more than 50000 queries.

Note that $a_{i,j}$ may have identical values. The k-th largest value is defined as the value of the k-th element when sorting the $n \times n$ elements in descending order.

Interaction Protocol

First, read the parameters n and k ($1 \leq n \leq 1000$, $1 \leq k \leq n \times n$) from the standard input.

Then, you can make no more than 50000 queries. To make a query, output "`? i j x`" ($1 \leq i, j \leq n$, $1 \leq x \leq n \times n$) on a separate line, and you should read the response from standard input. If $a_{i,j} \leq x$, the standard input will return 1, otherwise 0.

Finally, to give your answer, output "`! x`" on a separate line, representing the k-th largest value in the matrix. The output of the answer is **not** counted towards the limit of 50000 queries. After that, your program should terminate.

After outputting a query, do not forget to output end of line and flush the output. To do this:

- `fflush(stdout)` or `cout.flush()` for C++;

- `System.out.flush()` for Java;
- `stdout.flush()` for Python;

It is guaranteed that the matrix remains unchanged during the interaction.

Example

standard input	standard output
2 3	? 1 1 2
1	? 1 2 2
1	? 2 1 2
0	? 2 2 2
0	? 1 2 1
0	! 2

Note

One possible matrix in the example is $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$.

Problem G. Pack

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

You have n items of type A and m items of type B. Each A item has a value of a and each B item has a value of b.

You now need to determine a packaging solution for a certain product, which includes a certain number of item A and a certain number of item B (it is allowed to contain only item A or only item B), satisfying the total value of the items to be k. **In this packaging solution**, you aim to package as many products as possible until the remaining A and B items are not sufficient to form a complete product.

Your task is to find the minimum number of items that remain unpackaged among all possible packaging solutions.

Input

There are multiple test cases. The first line inputs an integer T $(1 \le T \le 50)$ indicating the number of test case groups. For each test case:

The first line contains five positive integers n, m, a, b, k $(1 \le n, m \le 10^9, 1 \le a, b \le 10^9, 1 \le k \le 10^9)$.

Output

For each test case, output a single integer on a new line, representing the minimum number of items that cannot be packaged.

Example

standard input	standard output
8 10 8 2 3 12 3 3 2 2 8 1 3 2 2 10 6 6 3 5 16 341 329 5 6 741 727 521 18 9 576 290646493 622766369 133 76 578504001 285261289 308082376 109 3 922747797	3 2 4 0 122 48 2764422 5914615

Note

In the first test case, there are 10 objects of type A with a value of 2 and 8 objects of type B with a value of 3. They can be packaged into products with a composition of 3 A objects and 2 B objects. This way, they can be packaged into 3 products, leaving 1 A object and 2 B objects remaining.

In the second test case, there are 3 objects of type A with a value of 2 and 3 objects of type B with a value of 2. They can be packaged into products with a composition of 2 A objects and 2 B objects. This way, they can be packaged into 1 product, leaving 1 A object and 1 B object remaining.

Problem H. Wings of Crystals

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

Everyone is aware that Flandre has wings adorned with crystals. However, it is less commonly known that her wings are actually made by herself.

Flandre has been given a tree composed of n crystals interconnected by $n-1$ edges. This arrangement ensures that each crystal is directly or indirectly connected to others. The *beauty value* of the i-th crystal is denoted as a_i.

A "wing" is defined as a sequence of arbitrary length with non-repeating elements $b_1, b_2, \ldots, b_k$ $(1 \le b_i \le n)$ such that for every integer $1 \le i < k$, the b_i-th crystal and the b_{i+1}-th crystal is connected by a single edge within the tree. The *beauty value* of a wing is calculated as the square of the sum of the *beauty values* of all included crystals, expressed mathematically as $\left(\sum_{i=1}^{k} a_{b_i}\right)^2$.

Flandre wants to craft some wings from the crystals in this tree, such that each crystal is contained by no more than one wing. Help her determine the maximum possible sum of the *beauty values* for the wings she can create.

Input

The first line contains one integer n $(1 \le n \le 2 \times 10^5)$.

The second line contains n integers $a_1, a_2, \ldots, a_n$ $(0 \le a_i \le 5000)$.

Each of the following $n-1$ lines contains two integers u, v $(1 \le u, v \le n)$, denoting an edge connecting the u-th and the v-th crystal in the tree. It is guaranteed that the given edges form a tree.

Output

Output a single integer, the maximum possible sum of *beauty values* for the wings.

Examples

standard input	standard output
1 19	361
3 100 100 500 1 2 1 3	490000
6 10 20 20 10 20 20 1 2 1 3 4 5 4 6 1 4	5000
9 0 1 0 1 1 0 1 1 1 1 2 1 3 1 4 1 5 2 6 2 7 3 8 6 9	14

Note

In the first example, Flandre can craft a single wing of the only crystal with beauty value $19^2 = 361$.

In the second example, Flandre can craft a single wing consisting of all n crystals with beauty value $(100 + 100 + 500)^2 = 490000$.

In the third example, Flandre can craft two wings $[2, 1, 3]$ and $[5, 4, 6]$. The sum of their beauty values is $(20 + 10 + 20)^2 + (20 + 10 + 20)^2 = 5000$.

Problem I. Cyclic Apple Strings

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Leading the Flutterhies, we arrived at the Sweet Apple Acres.

"Only to see her lower her body, two front hooves pressed against the ground, waist twisted, legs kicked, two powerful hind legs striking the trunk like bullets. With a muffled sound accompanied by the rustling of leaves, the entire tree's apples fell into the basket without missing a beat,"whispered one of the Flutterhies.

"Fluttershy, how do you have so many?"Would I talk like that, I had only just risen to this doubt when I was interrupted by Applejack's voice coming from far away.

"There's nothing wrong with that,"another Fluttershy said coolly, as the apples in Applejack's basket inexplicably flew up one by one.

Strange, why would some of me even have magic? I hurriedly helped Applejack gather the apples and stopped the actions of Flutterhies.

Formally, the act of stopping can be abstracted into the following problem:

Given a string s of length n consisting only of `0` and `1`. I can perform the following operation any number of times: choose a substring of s and a positive number k, then cyclically shift the substring k positions to the left.

String a is a substring of string b if and only if a can be obtained by deleting some characters from the beginning and end of b (which can be none or all of them).

Assume there is a string $s = s_0s_1s_2 \dots s_{m-2}s_{m-1}$, shifting it k positions to the left will result in $s_{k'}s_{k'+1} \dots s_{m-1}s_0s_1 \dots s_{k'-2}s_{k'-1}$, where $k' = k \bmod m$.

A string s consisting only of `0` and `1` is sorted if and only if for all $1 \le i < n$, $s_i \le s_{i+1}$. Here, characters are compared based on their numerical values.

For example, shifting the substring `1011` of string `0101100` two positions to the left results in `0111000`.

I need to find the minimum number of operations required to make s sorted.

Input

One line containing a string s consisting only of `0` and `1`, where $|s|$ denotes the length of s ($1 \le |s| \le 10^5$).

Output

Output one line containing an integer, representing the minimum number of operations required to make s sorted.

Examples

standard input	standard output
01010101	3
11001010001	3

Note

In the first example, one feasible set of minimal operations is `01010101` $\to$ `00110101` $\to$ `00011101` $\to$ `00001111`, where we shift the substring $[1,2]$ by 1 position, the substring $[2,4]$ by 2 positions, and the substring $[3,6]$ by 3 positions. Please note that string indices start from 0.

In the second example, one feasible set of minimal operations is `1100101001` $\to$ `0011101001` $\to$ `0011100011` $\to$ `0000011111`, where we shift the substring $[0,3]$ by 2 positions, the substring $[6,8]$ by 1 position, and the substring $[2,7]$ by 3 positions.

Problem J. Gensokyo Autobahn

Input file:	`standard input`
Output file:	`standard output`
Time limit:	6 seconds
Memory limit:	1024 megabytes

The Mystia's Izakaya has n branches in Gensokyo, numbered from 1 to n. There are m one-way roads of length 1 between these n branches, with the i-th road leading from branch u_i to branch v_i.

To strengthen the connections between the branches, Mystia decided to build some more roads, so she hired k construction teams. For the i-th team, Mystia will uniformly randomly select an integer j from $[1, m] \cap \mathbb{Z}$ and assign the i-th construction team to build a_i one-way roads of length 1 from branch u_j to branch v_j. The integers j selected for the construction teams are independent of each other.

The two largest branches of the Izakaya are branch 1 and branch n. Your task is to help Mystia calculate the expected number of shortest paths from branch 1 to branch n after the road construction is completed. The answer should be modulo 998244353.

Input

The first line contains three integers n, m, k ($2 \le n \le 2 \times 10^5, 1 \le m \le 2 \times 10^5, 1 \le k \le 2 \times 10^5$).

The i-th line of the following m lines contains two integers u_i, v_i ($1 \le u_i, v_i \le n$), representing a one-way road from branch u_i to branch v_i.

The next line contains k integers $a_1, a_2, \ldots, a_k$ ($1 \le a_i \le 10^9$), representing the number of roads each construction team will build.

It is guaranteed that there is at least one path from branch 1 to branch n.

There may be multiple edges and self-loops in the graph.

Output

Output a single integer, the expected number of shortest paths from branch 1 to branch n modulo 998244353.

Formally, let $M = 998244353$. It can be shown that the answer can be expressed as an irreducible fraction $\frac{p}{q}$, where p and q are integers and $q \not\equiv 0 \pmod{M}$. Output such an integer x that $0 \le x < M$ and $x \cdot q \equiv p \pmod{M}$.

Examples

standard input	standard output
4 4 1 1 2 1 3 2 4 3 4 2	4
3 5 2 1 1 1 2 1 2 2 1 2 3 5 3	399297752
9 8 5 1 3 3 2 1 4 4 2 4 5 2 9 5 9 7 8 1 2 3 4 5	308052020

Problem K. Party Games

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

After finally rounding up the Fluttershies at Sweet Apple Acres, we made our way to Pinkie Pie's Sugarcube Corner.

"Wheee!!! Fluttershies, let's have a party!"

There was no way around it; we had to join in and play with Pinkie Pie.

Pinkie Pie invented a two-player game, and we took turns challenging her. Specifically, the game goes as follows:

There are n integers $1, 2, 3, \dots, n$ arranged in a row **from left to right**. Both Pinkie Pie and I will take turns trying the following operation:

- If the bitwise XOR sum of the remaining integers is not 0, remove either the leftmost or the rightmost integer from the row without changing the order of the remaining numbers.

If the current player cannot make a move, they lose the game.

The game consists of T rounds, assuming "I" always starts first each round, and both of us aim for victory by making optimal moves. The question is whether "I" can achieve victory in each round of the game.

The bitwise XOR sum of several numbers $a_1, a_2, \dots, a_m$ is denoted by $a_1 \oplus a_2 \oplus \cdots \oplus a_m$. Particularly, the XOR sum of an empty set is 0.

The XOR operation, denoted by $\oplus$, is a binary operation that compares two binary numbers bit by bit. At each position, if the corresponding bits are not all 1 or not all 0, the result is 1; otherwise, it's 0.

Input

The first line contains an integer T ($1 \le T \le 10^5$), indicating the number of rounds in the game.

Following that are T lines, each containing an integer n ($1 \le n \le 10^6$), representing the number of integers in the corresponding round of the game.

Output

For each round of the game, output one line. If "I" can win, output "Fluttershy"; otherwise, output "Pinkie Pie"(without the quotes).

Example

standard input	standard output
3 1 2 3	Fluttershy Pinkie Pie Pinkie Pie

Note

In the first example, "I" choose to take 1, leaving Pinkie Pie unable to make a move. Thus, "I" win.

In the second example, regardless of whether "I" choose to take 1 or 2, Pinkie Pie can always take the remaining numbers, rendering "I" unable to make a move and lose the game.

In the third example, the initial XOR sum is already 0, so "I" cannot make a move and lose the game.

Problem L. Magic Fairies

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

"Fluttershy, this really isn't Discord's trickery," Twilight Sparkle said to me upon returning from the Canterlot Library. "It's just a group of mischievous magic fairies."

"Magic fairies?"

"They are a pure form of magical beings, capable of freely manipulating natural magic fields to construct external shells. They imitated your appearance to find you, probably just because they really like you."

"Is that so?" I glanced at the Fluttershies, and they nodded slightly before emitting a faint glow, blurring our vision.

"It's over now, now that we know all this, but they left behind a puzzle." We both watched as they gradually transformed into a group of shining butterflies.

"Let's take a look together." The butterflies drifted away in the sunlight, gradually disappearing into the play of light and shadow.

Specifically, the fairies have built n rectangular pillars of width 1 right next to each other on the numbered axes, and the height of the column at position i is h_i.

In particular, imaginary infinite-height pillars are considered to exist in front of the first pillar and behind the last one, i.e., h_0 and h_{n+1} are treated as infinite height. Also, no two pillars have the same height.

Next, q **independent** queries x, V are given, indicating that water falls vertically from the infinite height at position x, covering an area of V units. We need to answer how many of the tops of the pillars were submerged by water after the water flow was complete, i.e., they were covered with water of thickness greater than 0.

In particular, for convenience, it is guaranteed that all V are even.

Water flow: Assume that the water is an ideal fluid and no atmospheric pressure here, so the water always tends to flow to somewhere strictly lower, even when the water is on a horizontal stage. In particular, if the water is on the column surface of some pillars whose height is greater on both sides, the water will **equal** flow to both sides. Since no two pillars have the same height, this will only happen on the surface of pillar where the water initially falls.

Input

The first line contains an integer n ($1 \le n \le 2 \times 10^5$), representing the number of pillars.

The second line contains n integers $h_1, h_2, ..., h_n$ ($0 \le h_i \le 10^9$), representing the heights of the pillars. It is guaranteed that the heights of the pillars are all distinct.

The third line contains an integer q ($1 \le q \le 2 \times 10^5$), indicating the number of queries.

Following are q lines, each containing two integers x and V ($1 \le x \le n$, $2 \le V \le 10^9$, V is even), representing the position where the rain falls and the amount of rainfall.

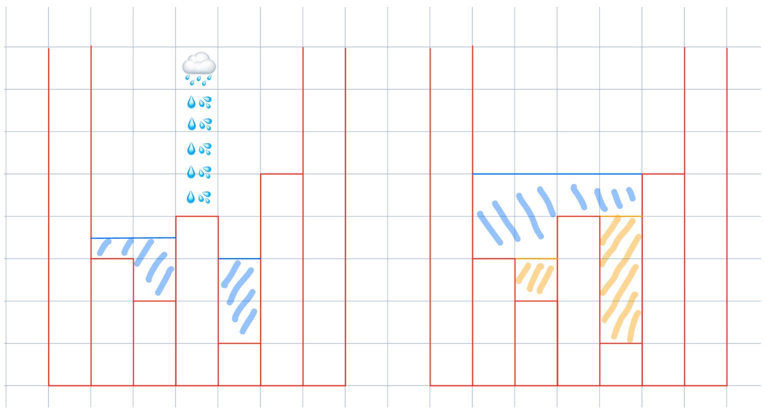
Output

Output q lines, each containing an integer, representing the answer to each query.

Example

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

Note



For the first query of the example, the blue line on the left graph indicates its water level. 4 units of water flow equally from position 3 to both sides. Eventually, positions 1, 2, and 4 are covered with water, with heights of 3.5, 3.5, and 3, respectively.

For the second query of the example, the blue line on the right graph indicates its water level. Specifically, positions 1, 2, 3, and 4 are covered with a water level of 5.

For the third query of the example, the orange line on the right indicates its water level. Specifically, positions 2 and 4 are covered with water levels of 3 and 4, respectively. Note that although water has flowed through position 3, there is no water accumulation at position 3 in the end, so it is not covered.

Problem M. Merge

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

Siri is playing a WeChat mini-game that involves a two-player duel, so he called Dagu to play the game with him.

The game process is as follows: At the start of the game, both players are given a number of soldiers. Siri receives n soldiers, the i-th of which with a combat power of a_i; Dagu also receives several soldiers. Each round, both sides summon one still-surviving soldier to fight. The side with the higher combat power will defeat the opponent's soldier and directly kill the opponent's general, winning the game. If the combat powers are equal, both soldiers die simultaneously and the next round begins. If all soldiers on both sides have fallen, the game is declared a draw.

To more easily win the game, Siri has installed a cheat that allows him to perform the following operation any number of times: merge two soldiers whose combat powers differ exactly by 1 into a new soldier with a combat power equal to the sum of the original two soldiers' combat powers.

Clearly, in this game, sorting the sequence of each player's soldiers' combat powers in descending order, the player with the lexicographically larger sequence has a guaranteed winning strategy. Help Siri use the cheat to merge soldiers so that his sequence of soldiers' combat powers, when sorted in descending order, is lexicographically as large as possible. You only need to output the combat powers of the merged soldiers in any order.

A sequence $a = [a_1, a_2, \ldots, a_{|a|}]$ is lexicographically smaller than a sequence $b = [b_1, b_2, \ldots, b_{|b|}]$ if and only if at least one of the following is true:

- a is an empty sequence and b is not an empty sequence.
- Both a and b are not empty sequences and $a_1 < b_1$.
- Both a and b are not empty sequences and $a_1 = b_1$ and $[a_2, a_3, \ldots, a_{|a|}]$ is lexicographically smaller than $[b_2, b_3, \ldots, b_{|b|}]$.

Input

The first line contains two integers n ($1 \le n \le 2 \times 10^5$), representing the number of soldiers Siri has.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^{18}$), where the i-th integer a_i represents the combat power of the i-th soldier among Siri's soldiers.

Output

The first line contains an integer k, representing the number of Siri's soldiers after merging.

The second line contains k integers, representing the combat powers of Siri's soldiers after merging.

Examples

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

Note

In the first example, the optimal strategy is:

- Merge $2, 3$ to form 5
- Merge $4, 5$ to form 9

In the second example, the optimal strategy is:

- Merge $2, 3$ to form 5
- Merge $4, 5$ to form 9
- Merge $3, 4$ to form 7

In the third example, the optimal strategy is:

- Merge $1, 2$ to form 3
- Merge $2, 3$ to form 5
- Merge $4, 5$ to form 9