# A. Anti Light's Cell Guessing

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are playing a game on a $n \times m$ grid, in which the computer has selected some cell $(x, y)$ of the grid, and you have to determine which one.

To do so, you will choose some $k$ and some $k$ cells $(x_1, y_1)$, $(x_2, y_2)$, $\ldots$, $(x_k, y_k)$, and give them to the computer. In response, you will get $k$ numbers $b_1$, $b_2$, $\ldots$ $b_k$, where $b_i$ is the manhattan distance from $(x_i, y_i)$ to the hidden cell $(x, y)$ (so you know which distance corresponds to which of $k$ input cells).

After receiving these $b_1$, $b_2$, $\ldots$, $b_k$, you have to be able to determine the hidden cell. What is the smallest $k$ for which is it possible to always guess the hidden cell correctly, no matter what cell computer chooses?

As a reminder, the manhattan distance between cells $(a_1, b_1)$ and $(a_2, b_2)$ is equal to $|a_1 - a_2| + |b_1 - b_2|$.

### Input

The first line of the input contains a single integer $t$ ($1 \le t \le 10^4$) — the number of test cases. The description of test cases follows.

The single line of each test case contains two integers $n$ and $m$ ($1 \le n, m \le 10^9$) — the number of rows and the number of columns in the grid.

### Output

For each test case print a single integer — the minimum $k$ for that test case.

**Example**

| input |
| --- |
| 2<br>2 3<br>3 1 |
| **output** |
| 2<br>1 |

### Note

In the first test case, the smallest such $k$ is $2$, for which you can choose, for example, cells $(1, 1)$ and $(2, 1)$.

Note that you can't choose cells $(1, 1)$ and $(2, 3)$ for $k = 2$, as both cells $(1, 2)$ and $(2, 1)$ would give $b_1 = 1, b_2 = 2$, so we wouldn't be able to determine which cell is hidden if computer selects one of those.

In the second test case, you should choose $k = 1$, for it you can choose cell $(3, 1)$ or $(1, 1)$.

# B. Kalindrome Array

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

An array $[b_1, b_2, \ldots, b_m]$ is a palindrome, if $b_i = b_{m+1-i}$ for each $i$ from $1$ to $m$. Empty array is also a palindrome.

An array is called **kalindrome**, if the following condition holds:

- It's possible to select some integer $x$ and delete some of the elements of the array equal to $x$, so that the remaining array (after gluing together the remaining parts) is a palindrome.

Note that you don't have to delete all elements equal to $x$, and you don't have to delete at least one element equal to $x$.

For example :

- $[1, 2, 1]$ is kalindrome because you can simply not delete a single element.
- $[3, 1, 2, 3, 1]$ is kalindrome because you can choose $x = 3$ and delete both elements equal to $3$, obtaining array $[1, 2, 1]$, which is a palindrome.
- $[1, 2, 3]$ is not kalindrome.

You are given an array $[a_1, a_2, \ldots, a_n]$. Determine if $a$ is kalindrome or not.

### Input

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

The first line of each test case contains a single integer $n$ ($1 \le n \le 2 \cdot 10^5$) — the length of the array.

The second line of each test case contains $n$ integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le n$) — elements of the array.

It's guaranteed that the sum of $n$ over all test cases won't exceed $2 \cdot 10^5$.

### Output

For each test case, print YES if $a$ is kalindrome and NO otherwise. You can print each letter in any case.

**Example**

| input |
| --- |
| 4<br>1<br>1<br>2<br>1 2<br>3<br>1 2 3<br>5<br>1 4 4 1 4 |
| **output** |
| YES<br>YES<br>NO<br>YES |

### Note

In the first test case, array $[1]$ is already a palindrome, so it's a kalindrome as well.

In the second test case, we can choose $x = 2$, delete the second element, and obtain array $[1]$, which is a palindrome.

In the third test case, it's impossible to obtain a palindrome.

In the fourth test case, you can choose $x = 4$ and delete the fifth element, obtaining $[1, 4, 4, 1]$. You also can choose $x = 1$, delete the first and the fourth elements, and obtain $[4, 4, 4]$.

# C. Keshi Is Throwing a Party

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Keshi is throwing a party and he wants everybody in the party to be happy.

He has $n$ friends. His $i$-th friend has $i$ dollars.

If you invite the $i$-th friend to the party, he will be happy only if at most $a_i$ people in the party are strictly richer than him and at most $b_i$ people are strictly poorer than him.

Keshi wants to invite as many people as possible. Find the maximum number of people he can invite to the party so that every invited person would be happy.

### Input

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

The first line of each test case contains a single integer $n$ $(1 \le n \le 2 \cdot 10^5)$ — the number of Keshi's friends.

The $i$-th of the following $n$ lines contains two integers $a_i$ and $b_i$ $(0 \le a_i, b_i < n)$.

It is guaranteed that the sum of $n$ over all test cases doesn't exceed $2 \cdot 10^5$.

### Output

For each test case print the maximum number of people Keshi can invite.

**Example**

| input |
| --- |
| 3<br>3<br>1 2<br>2 1<br>1 1<br>2<br>0 0<br>0 1<br>2<br>1 0<br>0 1 |
| **output** |
| 2<br>1<br>2 |

### Note

In the first test case, he invites the first and the second person. If he invites all of them, the third person won't be happy because there will be more than $1$ person poorer than him.

# D. Not Quite Lee

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

*Lee couldn't sleep lately, because he had nightmares. In one of his nightmares (which was about an unbalanced global round), he decided to fight back and propose a problem below (which you should solve) to balance the round, hopefully setting him free from the nightmares.*

A non-empty array $b_1, b_2, \ldots, b_m$ is called **good**, if there exist $m$ integer sequences which satisfy the following properties:

- The $i$-th sequence consists of $b_i$ consecutive integers (for example if $b_i = 3$ then the $i$-th sequence can be $(-1, 0, 1)$ or $(-5, -4, -3)$ but not $(0, -1, 1)$ or $(1, 2, 3, 4)$).
- Assuming the sum of integers in the $i$-th sequence is $sum_i$, we want $sum_1 + sum_2 + \ldots + sum_m$ to be equal to $0$.

You are given an array $a_1, a_2, \ldots, a_n$. It has $2^n - 1$ nonempty subsequences. Find how many of them are **good**.

As this number can be very large, output it modulo $10^9 + 7$.

An array $c$ is a subsequence of an array $d$ if $c$ can be obtained from $d$ by deletion of several (possibly, zero or all) elements.

### Input

The first line contains a single integer $n$ $(2 \le n \le 2 \cdot 10^5)$ — the size of array $a$.

The second line contains $n$ integers $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le 10^9)$ — elements of the array.

### Output

Print a single integer — the number of nonempty **good** subsequences of $a$, modulo $10^9 + 7$.

**Examples**

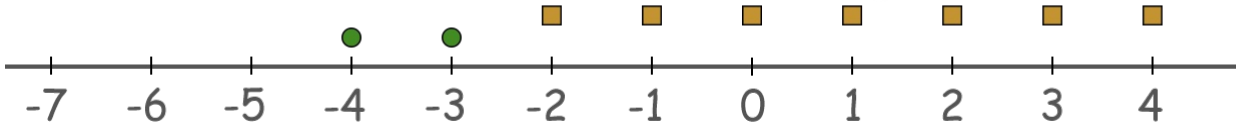
| input |
| --- |
| 4<br>2 2 4 7 |
| **output** |
| 10 |

| input |
| --- |
| 10<br>12391240 103904 1000000000 4142834 12039 142035823 1032840 49932183 230194823 984293123 |
| **output** |
| 996 |

### Note

For the first test, two examples of **good** subsequences are $[2, 7]$ and $[2, 2, 4, 7]$:

For $b = [2, 7]$ we can use $(-3, -4)$ as the first sequence and $(-2, -1, \ldots, 4)$ as the second. Note that subsequence $[2, 7]$ appears twice in $[2, 2, 4, 7]$, so we have to count it twice.




Green circles denote $(-3, -4)$ and orange squares denote $(-2, -1, \ldots, 4)$.
For $b = [2, 2, 4, 7]$ the following sequences would satisfy the properties: $(-1, 0)$, $(-3, -2)$, $(0, 1, 2, 3)$ and $(-3, -2, \ldots, 3)$

## E. AmShZ and G.O.A.T.

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Let's call an array of $k$ integers $c_1, c_2, \ldots, c_k$ **terrible**, if the following condition holds:

- Let $AVG$ be the $\frac{c_1+c_2+\ldots+c_k}{k}$(the average of all the elements of the array, it doesn't have to be integer). Then the number of elements of the array which are bigger than $AVG$ should be **strictly** larger than the number of elements of the array which are smaller than $AVG$. Note that elements equal to $AVG$ don't count.

  For example $c = \{1, 4, 4, 5, 6\}$ is **terrible** because $AVG = 4.0$ and 5-th and 4-th elements are greater than $AVG$ and 1-st element is smaller than $AVG$.

Let's call an array of $m$ integers $b_1, b_2, \ldots, b_m$ **bad**, if at least one of its non-empty subsequences is terrible, and **good** otherwise.

You are given an array of $n$ integers $a_1, a_2, \ldots, a_n$. Find the minimum number of elements that you have to delete from it to obtain a **good** array.

An array is a subsequence of another array if it can be obtained from it by deletion of several (possibly, zero or all) elements.

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

The first line of each test case contains a single integer $n$ ($2 \le n \le 2 \cdot 10^5$) — the size of $a$.

The second line of each testcase contains $n$ integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) — elements of array $a$.

In each testcase for any $1 \le i < n$ it is guaranteed that $a_i \le a_{i+1}$.

It is guaranteed that the sum of $n$ over all testcases doesn't exceed $2 \cdot 10^5$.

### Output
For each testcase, print the minimum number of elements that you have to delete from it to obtain a **good** array.

**Example**

| input |
| --- |
| 4<br>3<br>1 2 3<br>5<br>1 4 4 5 6<br>6<br>7 8 197860736 212611869 360417095 837913434<br>8<br>6 10 56026534 405137099 550504063 784959015 802926648 967281024 |
| **output** |
| 0<br>1<br>2<br>3 |

### Note
In the first sample, the array $a$ is already **good**.

In the second sample, it's enough to delete 1, obtaining array $[4, 4, 5, 6]$, which is **good**.

## F. Mashtali: a Space Oddysey

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

*Lee was planning to get closer to Mashtali's heart to proceed with his evil plan(which we're not aware of, yet), so he decided to beautify Mashtali's graph. But he made several rules for himself. And also he was too busy with his plans that he didn't have time for such minor tasks, so he asked you for help.*

Mashtali's graph is an **undirected** weighted graph with $n$ vertices and $m$ edges with weights equal to either 1 or 2. Lee wants to direct the edges of Mashtali's graph so that it will be as beautiful as possible.

Lee thinks that the beauty of a directed weighted graph is equal to the number of its Oddysey vertices. A vertex $v$ is an Oddysey vertex if $|d^+(v) - d^-(v)| = 1$, where $d^+(v)$ is the sum of weights of the outgoing from $v$ edges, and $d^-(v)$ is the sum of the weights of the incoming to $v$ edges.

Find the largest possible beauty of a graph that Lee can achieve by directing the edges of Mashtali's graph. In addition, find any way to achieve it.

Note that you have to orient each edge.

### Input
The first line contains two integers $n$ and $m$ ($1 \le n \le 10^5;\ 1 \le m \le 10^5$) — the numbers of vertices and edges in the graph.

The $i$-th line of the following $m$ lines contains three integers $u_i$, $v_i$ and $w_i$ ($1 \le u_i, v_i \le n;\ u_i \ne v_i;\ \mathbf{w_i} \in \{\mathbf{1}, \mathbf{2}\}$) — the endpoints of the $i$-th edge and its weight.

Note that the graph doesn't have to be connected, and it might contain multiple edges.

### Output
In the first line print a single integer — the maximum beauty of the graph Lee can achieve.

In the second line print a string of length $m$ consisting of 1s and 2s — directions of the edges.

If you decide to direct the $i$-th edge from vertex $u_i$ to vertex $v_i$, $i$-th character of the string should be 1. Otherwise, it should be 2.

**Examples**

| input |
| --- |
| 6 7 |

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

**output**

2
1212212

**input**

6 7
1 2 2
1 3 2
2 3 2
1 4 2
4 5 2
2 5 2
2 6 2

**output**

0
1212212

**input**

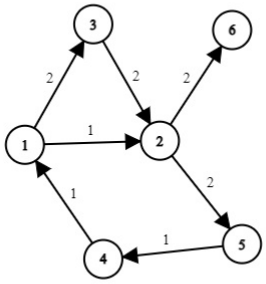
6 7
1 2 1
1 3 1
2 3 1
1 4 1
4 5 1
2 5 1
2 6 1

**output**

2
1212212

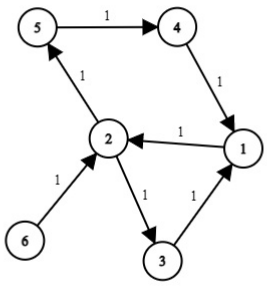
**Note**

Explanation for the first sample:




vertices $2$ and $5$ are Oddyseys.

Explanation for the third sample:




vertices $1$ and $6$ are Oddyseys.

## G. AmShZ Wins a Bet

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Right before the UEFA Euro 2020, AmShZ and Safar placed bets on who'd be the champion, AmShZ betting on Italy, and Safar betting on France.

Of course, AmShZ won. Hence, Safar gave him a bracket sequence $S$. Note that a bracket sequence is a string made of '(' and ')' characters.

AmShZ can perform the following operation any number of times:

- First, he cuts his string $S$ into three (possibly empty) contiguous substrings $A$, $B$ and $C$. Then, he glues them back by using a '('

and a ')' characters, resulting in a new string $S$ = $A$ + "(" + $B$ + ")" + $C$.
For example, if $S$ = "))((" and AmShZ cuts it into $A$ = "", $B$ = "))", and $C$ = "((", He will obtain $S$ = "))((((" as a new string.

After performing some (possibly none) operations, AmShZ gives his string to Keshi and asks him to find the initial string. Of course, Keshi might be able to come up with more than one possible initial string. Keshi is interested in finding the lexicographically smallest possible initial string.

Your task is to help Keshi in achieving his goal.

A string $a$ is lexicographically smaller than a string $b$ if and only if one of the following holds:

- $a$ is a prefix of $b$, but $a \neq b$;
- in the first position where $a$ and $b$ differ, the string $a$ has a letter that appears earlier in the alphabet than the corresponding letter in $b$.

### Input
The only line of input contains a single string $S$ — the string after the operations $(1 \leq |S| \leq 3 \cdot 10^5)$.

It is guaranteed that the first character of $S$ is ')'.

### Output
Print the lexicographically smallest possible initial string before operations.

**Example**

| input |
| --- |
| )(()(())) |
| **output** |
| )((())) |

### Note
In the first sample, you can transform ")((()))" into ")((()(()))" by splitting it into ")(", empty string, and "(()))". It can be shown that this is the lexicographically smallest possible initial string

## H. Squid Game

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

After watching the new ~~over-rated~~ series Squid Game, Mashtali and Soroush decided to hold their own Squid Games! Soroush agreed to be the host and will provide money for the winner's prize, and Mashtali became the Front Man!

$m$ players registered to play in the games to win the great prize, but when Mashtali found out how huge the winner's prize is going to be, he decided to ~~kill~~ eliminate all the players so he could take the money for himself!

Here is how evil Mashtali is going to eliminate players:

There is an unrooted tree with $n$ vertices. Every player has $2$ special vertices $x_i$ and $y_i$.

In one operation, Mashtali can choose any vertex $v$ of the tree. Then, for each remaining player $i$ he finds a vertex $w$ on the simple path from $x_i$ to $y_i$, which is the closest to $v$. If $w \neq x_i$ and $w \neq y_i$, player $i$ will be eliminated.

Now Mashtali wondered: "What is the minimum number of operations I should perform so that I can remove every player from the game and take the money for myself?"

Since he was only thinking about the money, he couldn't solve the problem by himself and asked for your help!

### Input
The first line contains $2$ integer $n$ and $m$ $(1 \leq n, m \leq 3 \cdot 10^5)$ — the number of vertices of the tree and the number of players.

The second line contains $n-1$ integers $par_2, par_3, \ldots, par_n$ $(1 \leq par_i < i)$ — denoting an edge between node $i$ and $par_i$.

The $i$-th of the following $m$ lines contains two integers $x_i$ and $y_i$ $(1 \leq x_i, y_i \leq n, x_i \neq y_i)$ — the special vertices of the $i$-th player.

### Output
Print the minimum number of operations Mashtali has to perform.

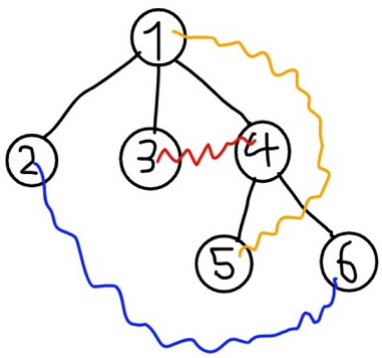
If there is no way for Mashtali to eliminate all the players, print $-1$.

**Examples**

| input |
| --- |
| 6 3<br>1 1 1 4 4<br>1 5<br>3 4<br>2 6 |
| **output** |
| 2 |

| input |
| --- |
| 5 3<br>1 1 3 3<br>1 2<br>1 4<br>1 5 |
| **output** |
| -1 |

### Note
Explanation for the first sample:




In the first operation, Mashtali can choose vertex $1$ and eliminate players with colors red and blue. In the second operation, he can choose vertex $6$ and eliminate the player with orange color.

In the second sample, Mashtali can't eliminate the first player.

# I. Mashtali vs AtCoder

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

After many unsuccessful tries, Mashtali decided to ~~copy~~ modify an AtCoder problem. So here is his ~~copied~~ new problem:

There is a tree with $n$ vertices and some non-empty set of the vertices are pinned to the ground.

Two players play a game against each other on the tree. They alternately perform the following action:

- Remove an edge from the tree, then remove every connected component that has no pinned vertex.
  The player who cannot move loses(every edge has been deleted already).

You are given the tree, but not the set of the pinned vertices. Your task is to determine, for each $k$, the winner of the game, if only the vertices $1, 2, 3, \ldots, k$ are pinned and both players play optimally.

### Input

The first line of input contains an integer $n$ — the number of vertices ($1 \leq n \leq 3 \cdot 10^5$).

The $i$-th of the following $n-1$ lines contains two integers $u_i, v_i$ ($1 \leq u_i, v_i \leq n$, $u_i \neq v_i$) — the endpoints of the $i$-th edge. It's guaranteed that these edges form a tree.

### Output

Print a string of length $n$. The $i$-th character should be '1' if the first player wins the $i$-th scenario, and '2' otherwise.

**Examples**

| input |
| --- |
| 5<br>1 2<br>2 3<br>2 4<br>4 5 |
| **output** |
| 11122 |

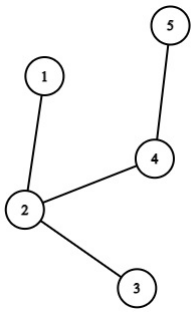
| input |
| --- |
| 5<br>1 2<br>2 3<br>1 4<br>4 5 |
| **output** |
| 21122 |

| input |
| --- |
| 6<br>1 2<br>2 4<br>5 1<br>6 3<br>3 2 |
| **output** |
| 111111 |

| input |
| --- |
| 7<br>1 2<br>3 7<br>4 6<br>2 3<br>2 4<br>1 5 |
| **output** |
| 2212222 |

### Note

Below you can see the tree in the first sample :




If $k = 1$ then the first player can cut the edge $(1, 2)$.

If $k = 2$ or $k = 3$, the first player can cut the edge $(2, 4)$, after that only the edges $(1, 2)$ and $(2, 3)$ remain. After the second players move, there will be a single edge left for the first player to cut. So first player wins.