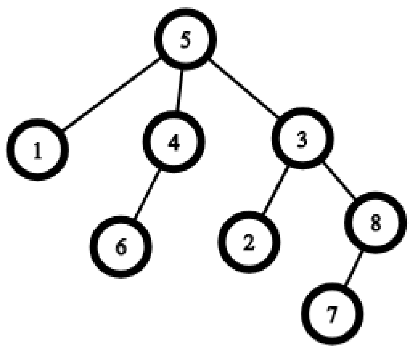
# Topic Stream Mashup: Trees

## A. Queen

1 second, 256 megabytes

You are given a rooted tree with vertices numerated from $1$ to $n$. A tree is a connected graph without cycles. A rooted tree has a special vertex named root.

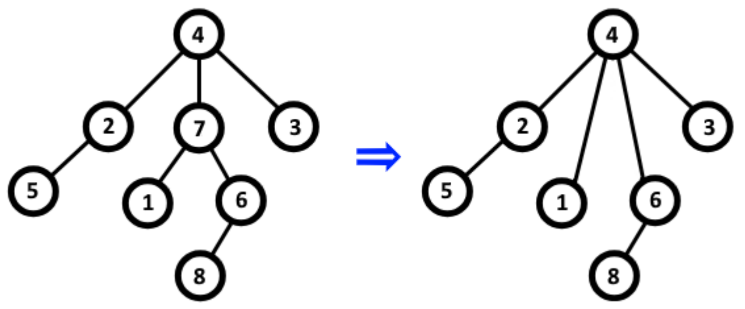
Ancestors of the vertex $i$ are all vertices on the path from the root to the vertex $i$, except the vertex $i$ itself. The parent of the vertex $i$ is the nearest to the vertex $i$ ancestor of $i$. Each vertex is a child of its parent. In the given tree the parent of the vertex $i$ is the vertex $p_i$. For the root, the value $p_i$ is $-1$.



An example of a tree with $n = 8$, the root is vertex $5$. The parent of the vertex $2$ is vertex $3$, the parent of the vertex $1$ is vertex $5$. The ancestors of the vertex $6$ are vertices $4$ and $5$, the ancestors of the vertex $7$ are vertices $8$, $3$ and $5$

You noticed that some vertices do not respect others. In particular, if $c_i = 1$, then the vertex $i$ does not respect any of its ancestors, and if $c_i = 0$, it respects all of them.

You decided to delete vertices from the tree one by one. On each step you select such a non-root vertex that it does not respect its parent and none of its children respects it. If there are several such vertices, you select the one with the **smallest number**. When you delete this vertex $v$, all children of $v$ become connected with the parent of $v$.



An example of deletion of the vertex $7$.

Once there are no vertices matching the criteria for deletion, you stop the process. Print the order in which you will delete the vertices. Note that this order is unique.

### Input

The first line contains a single integer $n$ ($1 \le n \le 10^5$) — the number of vertices in the tree.

The next $n$ lines describe the tree: the $i$-th line contains two integers $p_i$ and $c_i$ ($1 \le p_i \le n$, $0 \le c_i \le 1$), where $p_i$ is the parent of the vertex $i$, and $c_i = 0$, if the vertex $i$ respects its parents, and $c_i = 1$, if the vertex $i$ does not respect any of its parents. The root of the tree has $-1$ instead of the parent index, also, $c_i = 0$ for the root. It is guaranteed that the values $p_i$ define a rooted tree with $n$ vertices.

### Output

In case there is at least one vertex to delete, print the only line containing the indices of the vertices you will delete in the order you delete them. Otherwise print a single integer $-1$.

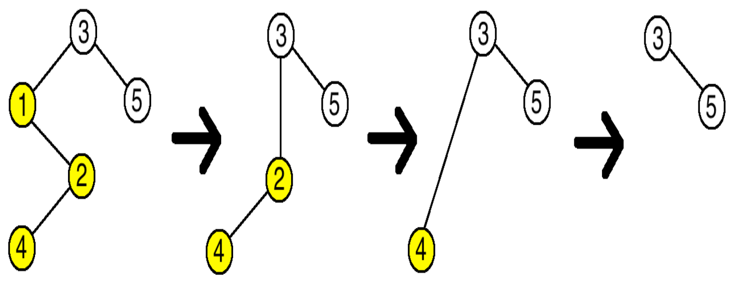
input
```
5
3 1
1 1
-1 0
2 1
3 0
```
output
```
1 2 4
```

input
```
5
-1 0
1 1
1 1
2 0
3 0
```
output
```
-1
```

input
```
8
2 1
-1 0
1 0
1 1
1 1
4 0
5 1
7 0
```
output
```
5
```

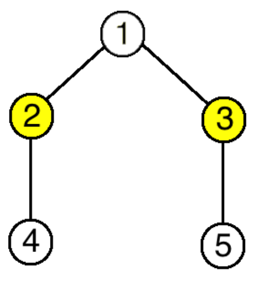
The deletion process in the first example is as follows (see the picture below, the vertices with $c_i = 1$ are in yellow):

- first you will delete the vertex $1$, because it does not respect ancestors and all its children (the vertex $2$) do not respect it, and $1$ is the smallest index among such vertices;
- the vertex $2$ will be connected with the vertex $3$ after deletion;
- then you will delete the vertex $2$, because it does not respect ancestors and all its children (the only vertex $4$) do not respect it;
- the vertex $4$ will be connected with the vertex $3$;
- then you will delete the vertex $4$, because it does not respect ancestors and all its children (there are none) do not respect it (vacuous truth);
- you will just delete the vertex $4$;
- there are no more vertices to delete.

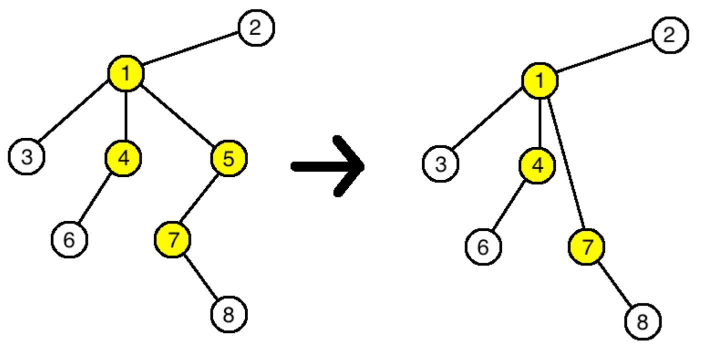


In the second example you don't need to delete any vertex:

- vertices $2$ and $3$ have children that respect them;
- vertices $4$ and $5$ respect ancestors.



In the third example the tree will change this way:



## B. Journey

2 seconds, 256 megabytes

There are $n$ cities and $n - 1$ roads in the Seven Kingdoms, each road connects two cities and we can reach any city from any other by the roads.

Theon and Yara Greyjoy are on a horse in the first city, they are starting traveling through the roads. But the weather is foggy, so they can't see where the horse brings them. When the horse reaches a city (including the first one), it goes to one of the cities connected to the current city. But it is a strange horse, it only goes to cities in which they weren't before. In each such city, the horse goes with equal probabilities and it stops when there are no such cities.

Let the length of each road be $1$. The journey starts in the city $1$. What is the expected length (expected value of length) of their journey? You can read about expected (average) value by the link https://en.wikipedia.org/wiki/Expected_value.

### Input

The first line contains a single integer $n$ ($1 \le n \le 100000$) — number of cities.

Then $n - 1$ lines follow. The $i$-th line of these lines contains two integers $u_i$ and $v_i$ ($1 \le u_i, v_i \le n$, $u_i \ne v_i$) — the cities connected by the $i$-th road.

It is guaranteed that one can reach any city from any other by the roads.

### Output

Print a number — the expected length of their journey. The journey starts in the city $1$.

Your answer will be considered correct if its absolute or relative error does not exceed $10^{-6}$.

Namely: let's assume that your answer is $a$, and the answer of the jury is $b$. The checker program will consider your answer correct, if $\frac{|a-b|}{max(1,b)} \le 10^{-6}$.

**input**
```
4
1 2
1 3
2 4
```
**output**
```
1.500000000000000
```

**input**
```
5
1 2
1 3
3 4
2 5
```
**output**
```
2.000000000000000
```

In the first sample, their journey may end in cities $3$ or $4$ with equal probability. The distance to city $3$ is $1$ and to city $4$ is $2$, so the expected length is $1.5$.

In the second sample, their journey may end in city $4$ or $5$. The distance to the both cities is $2$, so the expected length is $2$.

## C. Sum in the tree

2 seconds, 256 megabytes

Mitya has a rooted tree with $n$ vertices indexed from $1$ to $n$, where the root has index $1$. Each vertex $v$ initially had an integer number $a_v \ge 0$ written on it. For every vertex $v$ Mitya has computed $s_v$: the sum of all values written on the vertices on the path from vertex $v$ to the root, as well as $h_v$ — the depth of vertex $v$, which denotes the number of vertices on the path from vertex $v$ to the root. Clearly, $s_1 = a_1$ and $h_1 = 1$.

Then Mitya erased all numbers $a_v$, and by accident he also erased all values $s_v$ for vertices with even depth (vertices with even $h_v$). Your task is to restore the values $a_v$ for every vertex, or determine that Mitya made a mistake. In case there are multiple ways to restore the values, you're required to find one which minimizes the total sum of values $a_v$ for all vertices in the tree.

### Input
The first line contains one integer $n$ — the number of vertices in the tree ( $2 \le n \le 10^5$). The following line contains integers $p_2$ , $p_3$ , ... $p_n$ , where $p_i$ stands for the parent of vertex with index $i$ in the tree ($1 \le p_i < i$). The last line contains integer values $s_1$, $s_2$, ..., $s_n$ ($-1 \le s_v \le 10^9$), where erased values are replaced by $-1$.

### Output
Output one integer — the minimum total sum of all values $a_v$ in the original tree, or $-1$ if such tree does not exist.

**input**
```
5
1 1 1 1
1 -1 -1 -1 -1
```
**output**
```
1
```

**input**
```
5
1 2 3 1
1 -1 2 -1 -1
```
**output**
```
2
```

**input**
```
3
1 2
2 -1 1
```
**output**
```
-1
```

## D. Kay and Snowflake

3 seconds, 256 megabytes

After the piece of a devilish mirror hit the Kay's eye, he is no longer interested in the beauty of the roses. Now he likes to watch snowflakes.

Once upon a time, he found a huge snowflake that has a form of the tree (connected acyclic graph) consisting of $n$ nodes. The root of tree has index $1$. Kay is very interested in the structure of this tree.

After doing some research he formed $q$ queries he is interested in. The $i$-th query asks to find a centroid of the subtree of the node $v_i$. Your goal is to answer all queries.

*Subtree* of a node is a part of tree consisting of this node and all it's descendants (direct or not). In other words, subtree of node $v$ is formed by nodes $u$, such that node $v$ is present on the path from $u$ to root.

*Centroid* of a tree (or a subtree) is a node, such that if we erase it from the tree, the maximum size of the connected component will be at least two times smaller than the size of the initial tree (or a subtree).

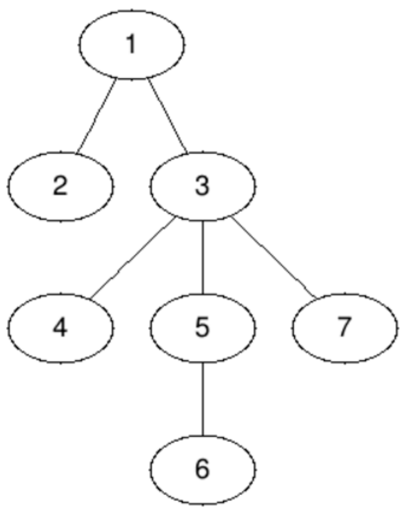
### Input
The first line of the input contains two integers $n$ and $q$ ($2 \le n \le 300\,000$, $1 \le q \le 300\,000$) — the size of the initial tree and the number of queries respectively.

The second line contains $n$ - $1$ integer $p_2, p_3, ..., p_n$ ($1 \le p_i \le n$) — the indices of the parents of the nodes from $2$ to $n$. Node $1$ is a root of the tree. It's guaranteed that $p_i$ define a correct tree.

Each of the following $q$ lines contain a single integer $v_i$ ($1 \le v_i \le n$) — the index of the node, that define the subtree, for which we want to find a centroid.

### Output
For each query print the index of a centroid of the corresponding subtree. If there are many suitable nodes, print any of them. It's guaranteed, that each subtree has at least one centroid.

**input**
```
7 4
1 1 3 3 5 3
1
2
3
5
```
**output**
```
3
2
3
6
```



The first query asks for a centroid of the whole tree — this is node $3$. If we delete node $3$ the tree will split in four components, two of size $1$ and two of size $2$.

The subtree of the second node consists of this node only, so the answer is $2$.

Node $3$ is centroid of its own subtree.

The centroids of the subtree of the node $5$ are nodes $5$ and $6$ — both answers are considered correct.

## E. Tree with Maximum Cost

2 seconds, 256 megabytes

You are given a tree consisting exactly of $n$ vertices. Tree is a connected undirected graph with $n-1$ edges. Each vertex $v$ of this tree has a value $a_v$ assigned to it.

Let $dist(x, y)$ be the distance between the vertices $x$ and $y$. The distance between the vertices is the number of edges on the simple path between them.

Let's define the cost of the tree as the following value: firstly, let's fix some vertex of the tree. Let it be $v$. Then the cost of the tree is $\sum_{i=1}^{n} dist(i, v) \cdot a_i$.

Your task is to calculate the **maximum possible cost** of the tree if you can choose $v$ arbitrarily.

### Input
The first line contains one integer $n$, the number of vertices in the tree ( $1 \le n \le 2 \cdot 10^5$).

The second line of the input contains $n$ integers $a_1, a_2, \dots, a_n$ ( $1 \le a_i \le 2 \cdot 10^5$), where $a_i$ is the value of the vertex $i$.

Each of the next $n-1$ lines describes an edge of the tree. Edge $i$ is denoted by two integers $u_i$ and $v_i$, the labels of vertices it connects ( $1 \le u_i, v_i \le n, u_i \ne v_i$).

It is guaranteed that the given edges form a tree.

### Output
Print one integer — the **maximum possible cost** of the tree if you can choose any vertex as $v$.

**input**

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

**output**

```
121
```

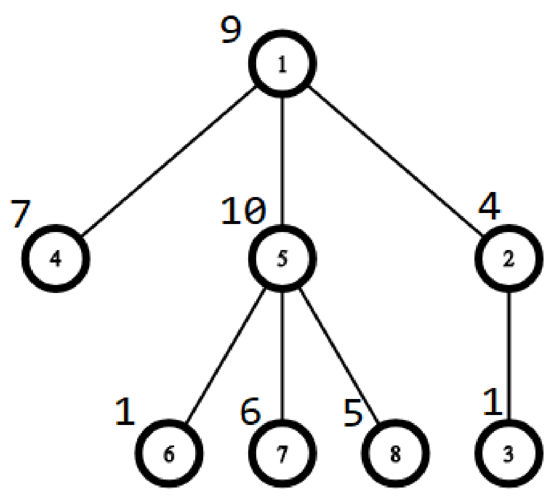
**input**

```
1
1337
```

**output**

```
0
```

Picture corresponding to the first example:



You can choose the vertex $3$ as a root, then the answer will be $2 \cdot 9 + 1 \cdot 4 + 0 \cdot 1 + 3 \cdot 7 + 3 \cdot 10 + 4 \cdot 1 + 4 \cdot 6 + 4 \cdot 5 = 18 + 4 +$ .

In the second example tree consists only of one vertex so the answer is always $0$.

## F. Tree Queries

2 seconds, 256 megabytes

You are given a rooted tree consisting of $n$ vertices numbered from $1$ to $n$. The root of the tree is a vertex number $1$.

A tree is a connected undirected graph with $n-1$ edges.

You are given $m$ queries. The $i$-th query consists of the set of $k_i$ distinct vertices $v_i[1], v_i[2], \dots, v_i[k_i]$. Your task is to say if there is a path from the root to some vertex $u$ such that each of the given $k$ vertices is either belongs to this path or has the distance $1$ to some vertex of this path.

### Input

The first line of the input contains two integers $n$ and $m$ ($2 \le n \le 2 \cdot 10^5$, $1 \le m \le 2 \cdot 10^5$) — the number of vertices in the tree and the number of queries.

Each of the next $n-1$ lines describes an edge of the tree. Edge $i$ is denoted by two integers $u_i$ and $v_i$, the labels of vertices it connects $(1 \le u_i, v_i \le n, u_i \ne v_i)$.

It is guaranteed that the given edges form a tree.

The next $m$ lines describe queries. The $i$-th line describes the $i$-th query and starts with the integer $k_i$ ($1 \le k_i \le n$) — the number of vertices in the current query. Then $k_i$ integers follow: $v_i[1], v_i[2], \dots, v_i[k_i]$ ($1 \le v_i[j] \le n$), where $v_i[j]$ is the $j$-th vertex of the $i$-th query.

It is guaranteed that all vertices in a single query are distinct.

It is guaranteed that the sum of $k_i$ does not exceed $2 \cdot 10^5$ ($\sum\limits_{i=1}^{m} k_i \le 2 \cdot 10^5$).

### Output

For each query, print the answer — "$\texttt{YES}$", if there is a path from the root to some vertex $u$ such that each of the given $k$ vertices is either belongs to this path or has the distance $1$ to some vertex of this path and "$\texttt{NO}$" otherwise.

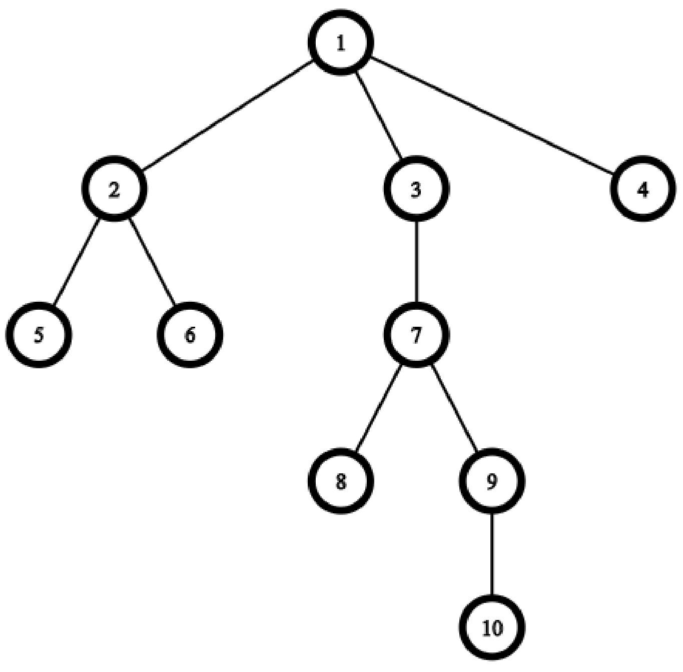
**input**

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

**output**

```
YES
YES
YES
YES
NO
NO
```

The picture corresponding to the example:



Consider the queries.

The first query is $[3, 8, 9, 10]$. The answer is "$\texttt{YES}$" as you can choose the path from the root $1$ to the vertex $u = 10$. Then vertices $[3, 9, 10]$ belong to the path from $1$ to $10$ and the vertex $8$ has distance $1$ to the vertex $7$ which also belongs to this path.

The second query is $[2, 4, 6]$. The answer is "$\texttt{YES}$" as you can choose the path to the vertex $u = 2$. Then the vertex $4$ has distance $1$ to the vertex $1$ which belongs to this path and the vertex $6$ has distance $1$ to the vertex $2$ which belongs to this path.

The third query is $[2, 1, 5]$. The answer is "$\texttt{YES}$" as you can choose the path to the vertex $u = 5$ and all vertices of the query belong to this path.

The fourth query is $[4, 8, 2]$. The answer is "$\texttt{YES}$" as you can choose the path to the vertex $u = 9$ so vertices $2$ and $4$ both have distance $1$ to the vertex $1$ which belongs to this path and the vertex $8$ has distance $1$ to the vertex $7$ which belongs to this path.

The fifth and the sixth queries both have answer "$\texttt{NO}$" because you cannot choose suitable vertex $u$.

## G. GCD Counting

4.5 seconds, 256 megabytes

You are given a tree consisting of $n$ vertices. A number is written on each vertex; the number on vertex $i$ is equal to $a_i$.

Let's denote the function $g(x, y)$ as the greatest common divisor of the numbers written on the vertices belonging to the simple path from vertex $x$ to vertex $y$ (including these two vertices). Also let's denote $dist(x, y)$ as the number of vertices on the simple path between vertices $x$ and $y$, including the endpoints. $dist(x, x) = 1$ for every vertex $x$.

Your task is calculate the maximum value of $dist(x, y)$ among such pairs of vertices that $g(x, y) > 1$.

### Input

The first line contains one integer $n$ — the number of vertices $(1 \le n \le 2 \cdot 10^5)$.

The second line contains $n$ integers $a_1$, $a_2$, ..., $a_n$ $(1 \le a_i \le 2 \cdot 10^5)$ — the numbers written on vertices.

Then $n-1$ lines follow, each containing two integers $x$ and $y$ $(1 \le x, y \le n, x \ne y)$ denoting an edge connecting vertex $x$ with vertex $y$. It is guaranteed that these edges form a tree.

### Output

If there is no pair of vertices $x, y$ such that $g(x, y) > 1$, print $0$. Otherwise print the maximum value of $dist(x, y)$ among such pairs.

**input**

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

**output**

```
1
```

**input**

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

**output**

```
2
```

**input**

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

**output**

```
0
```

## H. Expected diameter of a tree

3 seconds, 256 megabytes

Pasha is a good student and one of MoJaK's best friends. He always have a problem to think about. Today they had a talk about the following problem.

We have a forest (acyclic undirected graph) with $n$ vertices and $m$ edges. There are $q$ queries we should answer. In each query two vertices $v$ and $u$ are given. Let $V$ be the set of vertices in the connected component of the graph that contains $v$, and $U$ be the set of vertices in the connected component of the graph that contains $u$. Let's add an edge between some vertex $a \in V$ and some vertex in $b \in U$ and compute the value $d$ of the resulting component. If the resulting component is a tree, the value $d$ is the *diameter* of the component, and it is equal to `-1` otherwise. What is the expected value of $d$, if we choose vertices $a$ and $b$ from the sets uniformly at random?

Can you help Pasha to solve this problem?

The *diameter* of the component is the maximum *distance* among some pair of vertices in the component. The *distance* between two vertices is the minimum number of edges on some path between the two vertices.

Note that queries don't add edges to the initial forest.

### Input

The first line contains three integers $n$, $m$ and $q(1 \leq n, m, q \leq 10^5)$ — the number of vertices, the number of edges in the graph and the number of queries.

Each of the next $m$ lines contains two integers $u_i$ and $v_i$ $(1 \leq u_i, v_i \leq n)$, that means there is an edge between vertices $u_i$ and $v_i$.

It is guaranteed that the given graph is a forest.

Each of the next $q$ lines contains two integers $u_i$ and $v_i$ $(1 \leq u_i, v_i \leq n)$ — the vertices given in the $i$-th query.

### Output

For each query print the expected value of $d$ as described in the problem statement.

Your answer will be considered correct if its absolute or relative error does not exceed $10^{-6}$. Let's assume that your answer is $a$, and the jury's answer is $b$. The checker program will consider your answer correct, if $\frac{|a-b|}{\max(1,b)} \leq 10^{-6}$.

**input**

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

**output**

```
-1
2.0000000000
```

**input**

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

**output**

```
-1
2.6666666667
2.6666666667
```

In the first example the vertices $1$ and $3$ are in the same component, so the answer for the first query is `-1`. For the second query there are two options to add the edge: one option is to add the edge $1$ - $2$, the other one is $2$ - $3$. In both ways the resulting diameter is $2$, so the answer is $2$.

In the second example the answer for the first query is obviously `-1`. The answer for the second query is the average of three cases: for added edges $1$ - $2$ or $1$ - $3$ the diameter is $3$, and for added edge $1$ - $4$ the diameter is $2$. Thus, the answer is $\frac{3+3+2}{3} \approx 2.66666666$.