GPC-UPC BFS-DFS Contest I

A. King's Path

2 seconds, 256 megabytes

The black king is standing on a chess field consisting of 10^9 rows and 10^9 columns. We will consider the rows of the field numbered with integers from 1 to 10^9 from top to bottom. The columns are similarly numbered with integers from 1 to 10^9 from left to right. We will denote a cell of the field that is located in the i-th row and j-th column as (i, j).

You know that some squares of the given chess field are *allowed*. All allowed cells of the chess field are given as n segments. Each segment is described by three integers r_i, a_i, b_i ($a_i \le b_i$), denoting that cells in columns from number a_i to number b_i inclusive in the r_i-th row are allowed.

Your task is to find the minimum number of moves the king needs to get from square (x_0, y_0) to square (x_1, y_1), provided that he only moves along the allowed cells. In other words, the king can be located only on allowed cells on his way.

Let us remind you that a chess king can move to any of the neighboring cells in one move. Two cells of a chess field are considered neighboring if they share at least one point.

Input

The first line contains four space-separated integers x_0, y_0, x_1, y_1 ($1 \le x_0, y_0, x_1, y_1 \le 10^9$), denoting the initial and the final positions of the king.

The second line contains a single integer n ($1 \le n \le 10^5$), denoting the number of segments of allowed cells. Next n lines contain the descriptions of these segments. The i-th line contains three space-separated integers r_i, a_i, b_i ($1 \le r_i, a_i, b_i \le 10^9, a_i \le b_i$), denoting that cells in columns from number a_i to number b_i inclusive in the r_i-th row are allowed. Note that the segments of the allowed cells can intersect and embed arbitrarily.

It is guaranteed that the king's initial and final position are allowed cells. It is guaranteed that the king's initial and the final positions do not coincide. It is guaranteed that the total length of all given segments doesn't exceed 10^5.

Output

If there is no path between the initial and final position along allowed cells, print -1.

Otherwise print a single integer — the minimum number of moves the king needs to get from the initial position to the final one.

input

```
5 7 6 11
3
5 3 8
6 7 11
5 2 5
```

output

```
4
```

input

```
3 4 3 10
3
3 1 4
4 5 9
3 10 10
```

output

```
6
```

input

```
1 1 2 10
2
1 1 3
2 6 10
```

output

```
-1
```

B. Kefa and Park

2 seconds, 256 megabytes

Kefa decided to celebrate his first big salary by going to the restaurant.

He lives by an unusual park. The park is a rooted tree consisting of n vertices with the root at vertex 1. Vertex 1 also contains Kefa's house. Unfortunaely for our hero, the park also contains cats. Kefa has already found out what are the vertices with cats in them.

The leaf vertices of the park contain restaurants. Kefa wants to choose a restaurant where he will go, but unfortunately he is very afraid of cats, so there is no way he will go to the restaurant if the path from the restaurant to his house contains more than m **consecutive** vertices with cats.

Your task is to help Kefa count the number of restaurants where he can go.

Input

The first line contains two integers, n and m ($2 \le n \le 10^5$, $1 \le m \le n$) — the number of vertices of the tree and the maximum number of consecutive vertices with cats that is still ok for Kefa.

The second line contains n integers $a_1, a_2, \ldots, a_n$, where each a_i either equals to 0 (then vertex i has no cat), or equals to 1 (then vertex i has a cat).

Next $n - 1$ lines contains the edges of the tree in the format "$x_i\ y_i$" (without the quotes) ($1 \le x_i, y_i \le n$, $x_i \ne y_i$), where x_i and y_i are the vertices of the tree, connected by an edge.

It is guaranteed that the given set of edges specifies a tree.

Output

A single integer — the number of distinct leaves of a tree the path to which from Kefa's home contains at most m consecutive vertices with cats.

input

```
4 1
1 1 0 0
1 2
1 3
1 4
```

output

```
2
```

input

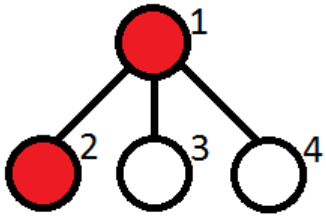
```
7 1
1 0 1 1 0 0 0
1 2
1 3
2 4
2 5
3 6
3 7
```

output

```
2
```

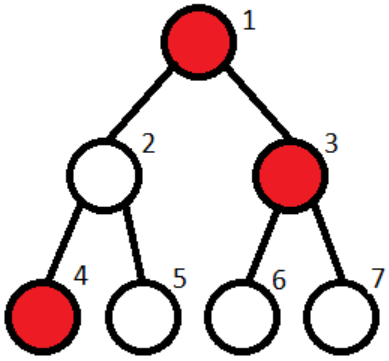
Let us remind you that a *tree* is a connected graph on n vertices and $n - 1$ edge. A *rooted* tree is a tree with a special vertex called *root*. In a rooted tree among any two vertices connected by an edge, one vertex is a parent (the one closer to the root), and the other one is a child. A vertex is called a *leaf*, if it has no children.

Note to the first sample test:



The vertices containing cats are marked red. The restaurants are at vertices 2, 3, 4. Kefa can't go only to the restaurant located at vertex 2.

Note to the second sample test:



The restaurants are located at vertices 4, 5, 6, 7. Kefa can't go to restaurants 6, 7.

C. Transformation: from A to B

1 second, 256 megabytes

Vasily has a number a, which he wants to turn into a number b. For this purpose, he can do two types of operations:

- multiply the current number by 2 (that is, replace the number x by $2 \cdot x$);
- append the digit 1 to the right of current number (that is, replace the number x by $10 \cdot x + 1$).

You need to help Vasily to transform the number a into the number b using only the operations described above, or find that it is impossible.

Note that in this task you are not required to minimize the number of operations. It suffices to find any way to transform a into b.

Input

The first line contains two positive integers a and b ($1 \le a < b \le 10^9$) — the number which Vasily has and the number he wants to have.

Output

If there is no way to get b from a, print "NO" (without quotes).

Otherwise print three lines. On the first line print "YES" (without quotes). The second line should contain single integer k — the length of the transformation sequence. On the third line print the sequence of transformations $x_1, x_2, ..., x_k$, where:

- x_1 should be equal to a,
- x_k should be equal to b,
- x_i should be obtained from x_{i-1} using any of two described operations ($1 < i \le k$).

If there are multiple answers, print any of them.

input
2 162
output
YES 5 2 4 8 81 162

input
4 42
output
NO

input
100 40021
output
YES 5 100 200 2001 4002 40021

D. Centroids

4 seconds, 512 megabytes

Tree is a connected acyclic graph. Suppose you are given a tree consisting of n vertices. The vertex of this tree is called *centroid* if the size of each connected component that appears if this vertex is removed from the tree doesn't exceed $\frac{n}{2}$.

You are given a tree of size n and can perform no more than one edge replacement. *Edge replacement* is the operation of removing one edge from the tree (without deleting incident vertices) and inserting one new edge (without adding new vertices) in such a way that the graph remains a tree. For each vertex you have to determine if it's possible to make it centroid by performing no more than one edge replacement.

Input

The first line of the input contains an integer n ($2 \le n \le 400\,000$) — the number of vertices in the tree. Each of the next $n - 1$ lines contains a pair of vertex indices u_i and v_i ($1 \le u_i, v_i \le n$) — endpoints of the corresponding edge.

Output

Print n integers. The i-th of them should be equal to 1 if the i-th vertex can be made centroid by replacing no more than one edge, and should be equal to 0 otherwise.

input
3 1 2 2 3
output
1 1 1

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

In the first sample each vertex can be made a centroid. For example, in order to turn vertex 1 to centroid one have to replace the edge $(2, 3)$ with the edge $(1, 3)$.

E. Broken Tree

4 seconds, 256 megabytes

You are given a tree that has n vertices, which are numbered from 1 to n, where the vertex number one is the root. Each edge has weight w_i and strength p_i.

Botanist Innokentiy, who is the only member of the jury of the Olympiad in Informatics, doesn't like broken trees.

The tree is broken if there is such an edge the strength of which is less than the sum of weight of subtree's edges to which it leads.

It is allowed to reduce weight of any edge by arbitrary integer value, but then the strength of its edge is reduced by the same value. It means if the weight of the edge is 10, and the strength is 12, then by the reducing the weight by 7 its weight will equal 3, and the strength will equal 5.

It is not allowed to increase the weight of the edge.

Your task is to get the tree, which is not broken, by reducing the weight of edges of the given tree, and also all edged should have the positive weight, moreover, the total weight of all edges should be as large as possible.

It is obvious that the strength of edges can not be negative, however it can equal zero if the weight of the subtree equals zero.

Input
The first line contains the integer n ($1 \le n \le 2 \cdot 10^5$) — the number of vertices in the tree. The next $n - 1$ lines contains the description of edges. Each line contains four integers x, y, w, p ($1 \le x, y \le n$, $1 \le w \le 10^9$, $0 \le p \le 10^9$), where x and y — vertices which connect the edge (the vertex number x is the parent of the vertex number y), w and p are the weight and the strength of the edge, accordingly. It is guaranteed that the edges describe the tree with the root in the vertex 1.

Output
If it is impossible to get unbroken tree from the given tree, print `-1` in the only line.

Otherwise, the output data should contain n lines:

In the first line print the number n — the number of vertices on the tree.

In the next $n - 1$ lines print the description of edges of the resulting tree. Each line should contain four integers x, y, w, p ($1 \le x, y \le n$, $1 \le w \le 10^9$, $0 \le p \le 10^9$), where x and y — vertices, which the edge connects (the vertex number x is the parent of the vertex number y), w and p are the new weight and the strength of the edge, accordingly.

Print edges in the same order as they are given in input data: the first two integers of each line should not be changed.

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

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

input
```
5
1 2 2 4
2 4 1 9
4 5 5 6
4 3 4 8
```
output
```
5
1 2 2 4
2 4 1 9
4 5 1 2
4 3 2 6
```

input
```
7
1 2 5 2
2 3 4 3
1 4 3 7
4 5 4 1
4 6 3 2
6 7 1 6
```
output
```
7
1 2 5 2
2 3 2 1
1 4 3 7
4 5 3 0
4 6 3 2
6 7 1 6
```

F. The Tag Game

1 second, 256 megabytes

Alice got tired of playing the tag game by the usual rules so she offered Bob a little modification to it. Now the game should be played on an undirected rooted tree of n vertices. Vertex 1 is the root of the tree.

Alice starts at vertex 1 and Bob starts at vertex x ($x \neq 1$). The moves are made in turns, Bob goes first. In one move one can either stay at the current vertex or travel to the neighbouring one.

The game ends when Alice goes to the same vertex where Bob is standing. Alice wants to minimize the total number of moves and Bob wants to maximize it.

You should write a program which will determine how many moves will the game last.

Input
The first line contains two integer numbers n and x ($2 \le n \le 2 \cdot 10^5$, $2 \le x \le n$).

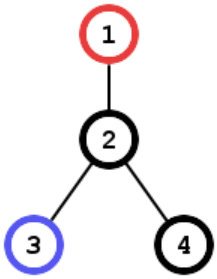
Each of the next $n - 1$ lines contains two integer numbers a and b ($1 \le a, b \le n$) — edges of the tree. It is guaranteed that the edges form a valid tree.

Output
Print the total number of moves Alice and Bob will make.

input
```
4 3
1 2
2 3
2 4
```
output
```
4
```

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

In the first example the tree looks like this:



The red vertex is Alice's starting position, the blue one is Bob's. Bob will make the game run the longest by standing at the vertex 3 during all the game. So here are the moves:

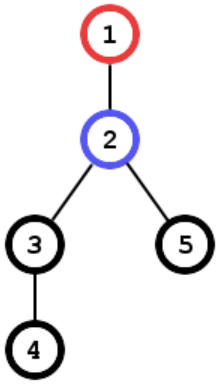
B: stay at vertex 3

A: go to vertex 2

B: stay at vertex 3

A: go to vertex 3

In the second example the tree looks like this:



The moves in the optimal strategy are:

B: go to vertex 3

A: go to vertex 2

B: go to vertex 4

A: go to vertex 3

B: stay at vertex 4

A: go to vertex 4

G. Police Stations

2 seconds, 256 megabytes

Inzane finally found Zane with a lot of money to spare, so they together decided to establish a country of their own.

Ruling a country is not an easy job. Thieves and terrorists are always ready to ruin the country's peace. To fight back, Zane and Inzane have enacted a very effective law: from each city it must be possible to reach a police station by traveling at most d kilometers along the roads.

There are n cities in the country, numbered from 1 to n, connected only by exactly $n - 1$ roads. All roads are 1 kilometer long. It is initially possible to travel from a city to any other city using these roads. The country also has k police stations located in some cities. In particular, the city's structure satisfies the requirement enforced by the previously mentioned law. Also note that there can be multiple police stations in one city.

However, Zane feels like having as many as $n - 1$ roads is unnecessary. The country is having financial issues, so it wants to minimize the road maintenance cost by shutting down as many roads as possible.

Help Zane find the maximum number of roads that can be shut down without breaking the law. Also, help him determine such roads.

Input

The first line contains three integers n, k, and d ($2 \le n \le 3 \cdot 10^5$, $1 \le k \le 3 \cdot 10^5$, $0 \le d \le n - 1$) — the number of cities, the number of police stations, and the distance limitation in kilometers, respectively.

The second line contains k integers $p_1, p_2, \ldots, p_k$ ($1 \le p_i \le n$) — each denoting the city each police station is located in.

The i-th of the following $n - 1$ lines contains two integers u_i and v_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$) — the cities directly connected by the road with index i.

It is guaranteed that it is possible to travel from one city to any other city using only the roads. Also, it is possible from any city to reach a police station within d kilometers.

Output

In the first line, print one integer s that denotes the maximum number of roads that can be shut down.

In the second line, print s distinct integers, the indices of such roads, in any order.

If there are multiple answers, print any of them.

input

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

output

```
1
5
```

input

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

output

```
2
4 5
```

In the first sample, if you shut down road 5, all cities can still reach a police station within $k = 4$ kilometers.

In the second sample, although this is the only largest valid set of roads that can be shut down, you can print either 4 5 or 5 4 in the second line.

H. Okabe and City

4 seconds, 256 megabytes

Okabe likes to be able to walk through his city on a path lit by street lamps. That way, he doesn't get beaten up by schoolchildren.

Okabe's city is represented by a 2D grid of cells. Rows are numbered from 1 to n from top to bottom, and columns are numbered 1 to m from left to right. Exactly k cells in the city are lit by a street lamp. It's guaranteed that the top-left cell is lit.

Okabe starts his walk from the top-left cell, and wants to reach the bottom-right cell. Of course, Okabe will only walk on lit cells, and he can only move to adjacent cells in the up, down, left, and right directions. However, Okabe can also temporarily light all the cells in any single row or column at a time if he pays 1 coin, allowing him to walk through some cells not lit initially.

Note that Okabe can only light a single row or column at a time, and has to pay a coin every time he lights a new row or column. To change the row or column that is temporarily lit, he must stand at a cell that is lit initially. Also, once he removes his temporary light from a row or column, all cells in that row/column not initially lit are now not lit.

Help Okabe find the minimum number of coins he needs to pay to complete his walk!

Input
The first line of input contains three space-separated integers n, m, and k ($2 \le n, m, k \le 10^4$).

Each of the next k lines contains two space-separated integers r_i and c_i ($1 \le r_i \le n$, $1 \le c_i \le m$) — the row and the column of the i-th lit cell.

It is guaranteed that all k lit cells are distinct. It is guaranteed that the top-left cell is lit.

Output
Print the minimum number of coins Okabe needs to pay to complete his walk, or `-1` if it's not possible.

input
4 4 5 1 1 2 1 2 3 3 3 4 3
output
2

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

input
2 2 4 1 1 1 2 2 1 2 2
output
0

input
5 5 4 1 1 2 2 3 3 4 4
output
3

In the first sample test, Okabe can take the path $(1,1) \to (2,1) \to (2,3) \to (3,3) \to (4,3) \to (4,4)$, paying only when moving to $(2, 3)$ and $(4, 4)$.

In the fourth sample, Okabe can take the path $(1,1) \to (1,2) \to (2,2) \to (3,2) \to (3,3) \to (3,4) \to (4,4) \to (5,4) \to (5,5)$, paying when moving to $(1, 2)$, $(3, 4)$, and $(5, 4)$.

I. Maximum average XOR paths

3 seconds, 256 megabytes

Given an undirected weighted graph with n vertices, m edges and k special nodes, you will have to process q queries:

For each query you will have to answer the maximum average path distance that could be made starting from one given node to any of the k special nodes.

The distance of any path is defined as the bitwise xor sum of their weights. For instance, if some path has weights of $w_1, w_2, \ldots w_l$, its distance will be $w_1 \oplus w_2 \oplus w_3 \oplus \ldots \oplus w_l$, where $\oplus$ is the Bitwise XOR operation, and its average distance will be $\frac{w_1 \oplus w_2 \oplus w_3 \oplus \ldots \oplus w_l}{l}$.

Input
The first line of input contains three integers n, m and k ($2 \le n \le 10^5, 1 \le m \le 10^5, 1 \le k < n$) — The number of vertices, edges and the special nodes in this graph.

The second line of input contains k **distinct** integers x_i ($1 \le x_i \le n$) — The i-th integer is the i-th special node.

The following m lines contain three integers a_i, b_i and w_i ($1 \le a_i, b_i \le n, 1 \le w_i \le 100$) — The description of the i-th edge between nodes a_i and b_i and weight w_i. It is guaranteed that $a_i \neq b_i$.

The next line contains an integer q ($1 \le q \le 10^5$) — The number of queries.

The following q lines contain a single integer u_i each — The reference node u_i for the query. It is guaranteed that u_i is not special.

Output
Print a single line for each query — The maximum average path distance that could be made starting from the given node in the query to any of the k special nodes. If no path connects the given node to any of the k special node print "unreachable".

Your answer will be considered correct if its absolute or relative error doesn't exceed 10^{-4}.

input
4 2 1 2 2 3 4 3 4 3 3 1 3 4
output
unreachable 4.000000 3.500000

For the sample case, it's easy to notice that the only paths that exist are:

1) $3 - 4 \to \text{Average} = 3$

2) $2 - 4 \to \text{Average} = \frac{7}{2} = 3.5$

3) $3 - 2 \to \text{Average} = 4$

J. Cow and Fields

2 seconds, 256 megabytes

Bessie is out grazing on the farm, which consists of n fields connected by m bidirectional roads. She is currently at field 1, and will return to her home at field n at the end of the day.

The Cowfederation of Barns has ordered Farmer John to install one extra bidirectional road. The farm has k special fields and he has decided to install the road between two different special fields. He may add the road between two special fields that already had a road directly connecting them.

After the road is added, Bessie will return home on the shortest path from field 1 to field n. Since Bessie needs more exercise, Farmer John must **maximize** the length of this shortest path. Help him!

Input
The first line contains integers n, m, and k ($2 \le n \le 2 \cdot 10^5$, $n - 1 \le m \le 2 \cdot 10^5$, $2 \le k \le n$) — the number of fields on the farm, the number of roads, and the number of special fields.

The second line contains k integers $a_1, a_2, \ldots, a_k$ ($1 \le a_i \le n$) — the special fields. All a_i are distinct.

The i-th of the following m lines contains integers x_i and y_i ($1 \le x_i, y_i \le n$, $x_i \neq y_i$), representing a bidirectional road between fields x_i and y_i.

It is guaranteed that one can reach any field from every other field. It is also guaranteed that for any pair of fields there is at most one road connecting them.

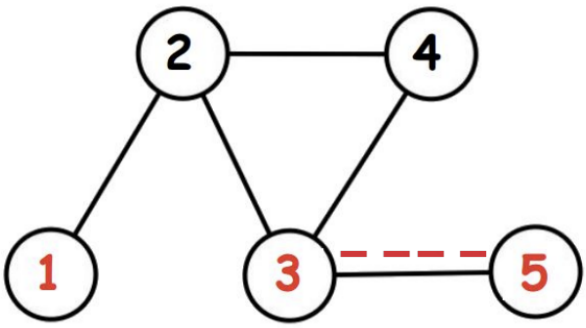
Output

Output one integer, the **maximum** possible length of the shortest path from field 1 to n after Farmer John installs one road optimally.

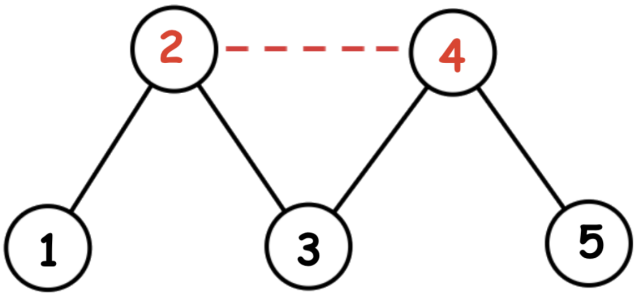
input
5 5 3 1 3 5 1 2 2 3 3 4 3 5 2 4
output
3

input
5 4 2 2 4 1 2 2 3 3 4 4 5
output
3

The graph for the first example is shown below. The special fields are denoted by red. It is optimal for Farmer John to add a road between fields 3 and 5, and the resulting shortest path from 1 to 5 is length 3.



The graph for the second example is shown below. Farmer John must add a road between fields 2 and 4, and the resulting shortest path from 1 to 5 is length 3.



K. Jiro's Job

2 seconds, 512 megabytes

Fox Jiro is one of the staffs of the ACM-ICPC 2018 Asia Yokohama Regional Contest and is responsible for designing the network for the venue of the contest. His network consists of N computers, which are connected by M cables. The i-th cable connects the a_i-th computer and the b_i-th computer, and it carries data in both directions. Your team will use the S-th computer in the contest, and a judge server is the T-th computer.

He decided to adjust the routing algorithm of the network to maximize the performance of the contestants through the magical power of prime numbers. In this algorithm, a packet (a unit of data carried by the network) is sent from your computer to the judge server passing through the cables a prime number of times if possible. If it is impossible, the contestants cannot benefit by the magical power of prime numbers. To accomplish this target, a packet is allowed to pass through the same cable multiple times.

You decided to write a program to calculate the minimum number of times a packet from S to T needed to pass through the cables. If the number of times a packet passes through the cables cannot be a prime number, print -1.

Input

The first line of the input consists of four integers N, M, S and T $(2 \leq N \leq 10^5, 1 \leq M \leq 10^5, 1 \leq S, T \leq N, S \neq T)$.

The i-th line of the following M lines consists of two integers a_i and b_i $(1 \leq a_i < b_i \leq N)$, which means the i-th cables connects the a_i-th computer and the b_i-th computer in the network. You can assume that the network satisfies the following conditions.

- The network has no multi-edge, i.e., $(a_i, b_i) \neq (a_j, b_j)$ for all i, j $(1 \leq i < j \leq M)$.
- The packets from N computers are reachable to T by passing through some number of cables. The number is not necessarily a prime.

Output

If there are ways such that the number of times a packet sent from S to T passes through the cables is a prime number, print the minimum prime number of times in one line. Otherwise, print -1.

input
5 5 1 5 1 2 1 4 2 3 3 4 3 5
output
3

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

input
2 1 1 2 1 2
output
3

input
3 3 1 2 1 2 1 3 2 3
output
2