Mock INOI #9

A. Permutation Recovery

3 seconds, 256 megabytes

There is a permutation P of N elements, but you don't know what it is. Instead, you are given Q queries. Each query is a tuple of three integers (L, R, I), such that $1 \le L \le I \le R \le N$, and signifies that the minimum value between the positions L to R inclusive is at position I. Note that the positions in this problem are one-indexed.

Find any permutation that satisfies all these queries. In particular, if multiple exist, you can output any one. If no permutations exist, print -1, exactly once.

Input

The first line contains two space-separated integers N and Q. This means that the permutation has length N, and there are Q queries.

The following Q lines contain the queries, each of which is three space-separated integers L, R and I.

Output

If no permutations exist, output -1.

Otherwise, output N space-separated integers, representing any permutation satisfying all the constraints.

Scoring

For all subtasks, $N, Q \le 3 \times 10^5$.

- Subtask 1 [4 points]: $L = R$ for all queries.
- Subtask 2 [8 points]: $I = R$ for all queries.
- Subtask 3 [11 points]: $N \le 10$ and $Q \le 10$.
- Subtask 4 [17 points]: $N \le 20$ and $Q \le 10$.
- Subtask 5 [19 points]: No two query ranges intersect. In other words, there exists a way to rearrange the query list, such that $R_i < L_{i+1}$ for all i from 1 to $Q - 1$.
- Subtask 6 [24 points]: $N, Q \le 2000$
- Subtask 7 [17 points]: No additional constraints.

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

input
2 2 1 2 1 1 2 2
output
-1

B. Life's Journey

6 seconds, 512 megabytes

Our lives are very long. On the other hand, for most ordinary people, it can be considered very repetitive. This question is about this paradox of life.

We say that there are n different tasks we do in life. For example, a task could be "studying", or "watching anime". In addition, there are m different *unidirectional* connections between the tasks. For example, if we are "talking with friends", we may "play football with friends" soon after. In other words, the map of life may be considered a graph with n nodes and m unidirectional edges, the i-th of which is from node u_i to node v_i. Each edge takes time w_i to traverse, meaning that if we are at task u_i at time d, we will be at task v_i at time $d + w_i$.

It is known that life lasts for a time L. In that time, we will be born, go through various struggles and joys, and eventually die (return to where we came from). In other words, life can be depicted as a journey on the "map of life" starting at node 1 at time 0, going through various nodes (which may include node 1 itself), and finally returning to node 1 at time L. Note that you return to node 1 at time L **exactly**, as that is when you are fated to die - no earlier, no later.

Doing a task gives some *happiness*. That is, each node has a value c_i, such that if you pass through this node you gain that value. Since life is very repetitive, you may pass through a single node many times, and gain this value each time.

However, although life is very repetitive, it would be a mistake to say that this drear and repetition is all there is. We say that there are k potential *major events* in life, which occur at time t_i in node x_i. If you happen to be doing the task x_i at time t_i, you will gain y_i units of happiness. For example, if you complete the task "confess your love" at the perfect time, you may get married, which would undoubtedly make you very happy.

Considering both the boring tasks and the major events, the question remains: what is the maximum amount of happiness you can get on this journey of life?

Input

The first line of input contains four integers, n, m, L, k. This is the number of tasks, number of unidirectional connections, length of life, and number of potential major events respectively.

The next line contains n integers. The i-th integer is c_i - the happiness gained from completing task i.

The following m lines each contain three integers u_i, v_i, w_i, denoting a unidirectional connection from task u_i to task v_i which takes time w_i to traverse.

The final k lines each contain three integers t_i, x_i, y_i, denoting a potential major event at time t_i in node x_i which gives happiness y_i.

Output

Output one integer - the maximum possible total happiness that can be gained in life. If it is impossible to return to the node 1 at time *exactly* L, the map of life given to you was clearly inaccurate, so output -1 in this case.

Scoring

For all subtasks, $1 \le n \le 50$, $n \le m \le 550$, $0 \le k \le 200$, $1 \le t_i \le L \le 10^9$, $1 \le w_i \le 5$, $1 \le c_i \le 55000$, $1 \le u_i, v_i, x_i \le n$, $1 \le y_i \le 10^9$.

In addition, it is guaranteed that $u_i \neq v_i$ for all i, all t_i are distinct, and that there is an outgoing edge from each node. However, it is possible that there may be connections with the same endpoints, in other words that $u_i = u_j$ and $v_i = v_j$ for some $1 \le i < j \le n$.

- Subtask 1 [20 points]: $n \le 5$, $m \le 50$, $L \le 5$
- Subtask 2 [20 points]: $m \le 50$, $L \le 55000$
- Subtask 3 [10 points]: $m \le 50$, $n = m$, $u_i = i$, $v_i = (i \mod n) + 1$
- Subtask 4 [15 points]: $m \le 50$, $k = 0$
- Subtask 5 [10 points]: $m \le 50$, $k \le 10$
- Subtask 6 [10 points]: $m \le 50$
- Subtask 7 [15 points]: No additional constraints.

input

```
3 4 11 0
1 3 4
1 2 1
2 1 3
2 3 2
3 1 4
```

output

```
13
```

input

```
4 8 16 3
3 1 2 4
1 2 1
1 3 1
1 3 2
3 4 3
2 3 2
3 2 1
4 2 1
4 1 5
3 3 5
1 2 5
5 4 20
```

output

```
39
```

In sample 1, one possible best route is $1 \to 2 \to 1 \to 2 \to 3 \to 1$.

In sample 2, the best route is $1 \to 3 \to 4 \to 2 \to 3 \to 4 \to 1$.

C. Protection

2 seconds, 1024 megabytes

UFDS country contains N cities, numbered from 0 to $N-1$, connected by $N-1$ roads. It is possible to travel between any two cities in the country.

You have enough power to protect any K cities you wish. You can also choose to protect less than K cities. However, the K cities you protect must all be connected. In other words, if you choose to protect two cities A and B, you must ensure that all cities on the path between A and B are protected too.

Now, consider the K cities you have connected. A range of cities from L to R is covered if all the cities L, $L+1$ ... R are chosen in the set of K protected cities. The length of a range is the number of cities in the range, so $R-L+1$.

If you choose the cities to protect optimally, what is the longest range of cities you can cover? Output the length of this longest range.

Input

The first line contains two integers, N and K.

The following $N-1$ lines contain the roads in the country, defined by the two endpooints of the roads.

Output

Output a single integer, the length of the longest range of cities you can cover.

Scoring

For all subtasks, $1 \leq K \leq N \leq 3 \times 10^5$.

- Subtask 1 [4 points]: $K = 1$
- Subtask 2 [6 points]: $K \leq 2, N \leq 1500$
- Subtask 3 [4 points]: $K \leq 2$
- Subtask 4 [10 points]: $N \leq 20$
- Subtask 5 [18 points]: $N \leq 2000$
- Subtask 6 [5 points]: The distance between cities A and B in the country is $|A-B|$.
- Subtask 7 [17 points]: There exists some pair of cities (A, B) in the country such that the distance between A and B is $N-1$.
- Subtask 8 [15 points]: $N \leq 50000$
- Subtask 9 [21 points]: No additional constraints

input

```
10 6
3 9
9 5
1 8
8 5
7 4
6 0
3 6
6 2
0 7
```

output

```
3
```