AQT's Tree Game

Time limit: 0.6s **Memory limit:** 128M
Java 8: 1.0s

AQT is playing a game. The map in this game is a tree with N nodes and $N-1$ edges. Since it's a tree, there is exactly one path to connect any two nodes. **AQT** has M weapons and the weapon i can block the path from node a_i to node b_i with a cost c_i. There are T monsters living in the tree. A monster j will travel from node u_j to node v_j. **AQT** can catch a monster if the path he blocks is an exact subpath of the monster's path. **AQT** can reuse his weapon, and the path is automatically unblocked after he catches a monster. However, **AQT** thinks this game is not challenging enough. For each monster j, he wants to use the k_j^{th} minimal cost weapon among all the weapons which can catch the monster j. Can you write a program to help him?

Input Specification

The first line contains 3 integers, N, M, and T $(N, M, T \leq 40\,000)$, which represent the number of nodes, the number of weapons, and the number of monsters, respectively.

Each of the following $N-1$ lines contains 2 integers, a and b $(1 \leq a, b \leq N)$, representing an edge between node a and node b.

Each of the following M lines contains 3 integers, a, b and c ($1 \leq a, b \leq N$ and $a \neq b$, $0 \leq c \leq 10^9$), representing a weapon which can block the path from node a to node b with a cost of c.

Each of the following T lines contains 3 integers, a, b and k $(1 \leq a, b \leq N)$, representing a monster's path from node a to node b and the k^{th} min cost weapon **AQT** wants to choose. **It's guaranteed the k^{th} min cost weapon exists.**

Output Specification

Output one line for each monster j, the k_j^{th} min cost to catch the monster j.

Sample Input 1

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

Sample Output 1

```
5
4
```

Sample Input 2

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

Sample Output 2

```
6
5
6
4
4
2
2
2
2
2
```