Problem B. Red Black Tree

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

BaoBao has just found a rooted tree with n vertices and $(n-1)$ weighted edges in his backyard. Among the vertices, m of them are red, while the others are black. The root of the tree is vertex 1 and it's a red vertex.

Let's define the cost of a red vertex to be 0, and the cost of a black vertex to be the distance between this vertex and its nearest red ancestor.

Recall that

- The length of a path on the tree is the sum of the weights of the edges in this path.
- The distance between two vertices is the length of the shortest path on the tree to go from one vertex to the other.
- Vertex u is the ancestor of vertex v if it lies on the shortest path between vertex v and the root of the tree (which is vertex 1 in this problem).

As BaoBao is bored, he decides to play q games with the tree. For the i-th game, BaoBao will select k_i vertices $v_{i,1}, v_{i,2}, \ldots, v_{i,k_i}$ on the tree and try to minimize the maximum cost of these k_i vertices by changing at most one vertex on the tree to a red vertex.

Note that

- BaoBao is free to change any vertex among all the n vertices to a red vertex, NOT necessary among the k_i vertiecs whose maximum cost he tries to minimize.
- All the q games are independent. That is to say, the tree BaoBao plays with in each game is always the initial given tree, NOT the tree modified during the last game by changing at most one vertex.

Please help BaoBao calculate the smallest possible maximum cost of the given k_i vertices in each game after changing at most one vertex to a red vertex.

Input

There are multiple test cases. The first line of the input is an integer T, indicating the number of test cases. For each test case:

The first line contains three integers n, m and q ($2 \le m \le n \le 10^5$, $1 \le q \le 2 \times 10^5$), indicating the size of the tree, the number of red vertices and the number of games.

The second line contains m integers $r_1, r_2, \ldots, r_m$ ($1 = r_1 < r_2 < \cdots < r_m \le n$), indicating the red vertices.

The following $(n-1)$ lines each contains three integers u_i, v_i and w_i ($1 \le u_i, v_i \le n$, $1 \le w_i \le 10^9$), indicating an edge with weight w_i connecting vertex u_i and v_i in the tree.

For the following q lines, the i-th line will first contain an integer k_i ($1 \le k_i \le n$). Then k_i integers $v_{i,1}, v_{i,2}, \ldots, v_{i,k_i}$ follow ($1 \le v_{i,1} < v_{i,2} < \cdots < v_{i,k_i} \le n$), indicating the vertices whose maximum cost BaoBao has to minimize.

It's guaranteed that the sum of n in all test cases will not exceed 10^6, and the sum of k_i in all test cases will not exceed 2×10^6.

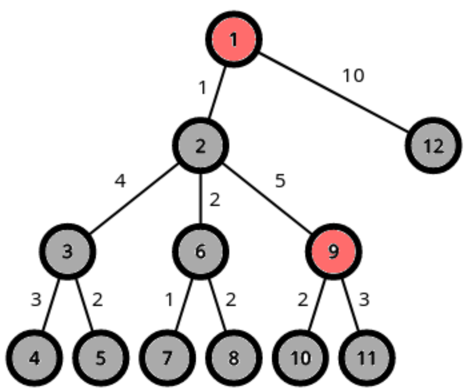
Output

For each test case output q lines each containing one integer, indicating the smallest possible maximum cost of the k_i vertices given in each game after changing at most one vertex in the tree to a red vertex.

Example

standard input	standard output
2	4
12 2 4	5
1 9	3
1 2 1	8
2 3 4	0
3 4 3	0
3 5 2	0
2 6 2	
6 7 1	
6 8 2	
2 9 5	
9 10 2	
9 11 3	
1 12 10	
3 3 7 8	
4 4 5 7 8	
4 7 8 10 11	
3 4 5 12	
3 2 3	
1 2	
1 2 1	
1 3 1	
1 1	
2 1 2	
3 1 2 3	

Note



The first sample test case is shown above. Let's denote $C(v)$ as the cost of vertex v.

For the 1st game, the best choice is to make vertex 2 red, so that $C(3) = 4$, $C(7) = 3$ and $C(8) = 4$. So the answer is 4.

For the 2nd game, the best choice is to make vertex 3 red, so that $C(4) = 3$, $C(5) = 2$, $C(7) = 4$ and $C(8) = 5$. So the answer is 5.

For the 3rd game, the best choice is to make vertex 6 red, so that $C(7) = 1$, $C(8) = 2$, $C(10) = 2$ and $C(11) = 3$. So the answer is 3.

For the 4th game, the best choice is to make vertex 12 red, so that $C(4) = 8$, $C(5) = 7$ and $C(12) = 0$. So the answer is 8.