Tree Flip

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

Alu will be given a rooted tree where each node contains an integer value of 0 or 1. He can perform flip operations at some nodes. This operation will flip the value of the selected node and the values of all the children of the selected node (note, it just flips the values of the children, not of all the descendants). Flipping a value means changing 0 to 1 or 1 to 0. Alu is intelligent so he performs this operation the minimum number of times to make the values of all nodes zero.

However, it's not fun if the tree is static. So here comes Begun into the scene. He has a tree. He can perform following updates on his tree:

Update 1: Begun flips the value of a particular node (note, the value of the children of the selected node are not flipped).

Update 2: Begun makes a particular node the root of the tree.

After each update, you need to output the minimum number of operations Alu requires to perform to make all the values in the tree zero. Note, Alu does not change the Begun's tree. You may imagine that Alu performs his operations on a copy of Begun's tree.

Input

Input begins with the number of test cases, $T\,(1 \leq T \leq 10^4)$.

Each case begins with two positive integers, the number of nodes n and the number of updates q. Next line contains n integers, the i^{th} integer is the value of the i^{th} node. Each of the next $n-1$ lines consists of two integers: u and $v\,(1 \leq u, v \leq n)$. These describe the edges of the tree.

Each of the next q lines contains two integers: $type\,(type \in \{1, 2\})$ and $x\,(1 \leq x \leq n)$. $type = 1$ denotes **Update 1** and $type = 2$ denotes **Update 2**. x denotes the node on which the update is performed. You may assume that the input tree is valid. Initially 1 is the root of the tree.

It is guaranteed that the sum of n doesn't exceed 10^5 and sum of q doesn't exceed 10^5 across all test cases.

Output

For each test case process the updates and then for each update print the answer in a single line.

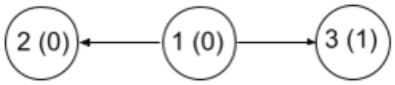
Example

standard input	standard output
1 3 3 0 0 1 1 2 3 1 1 1 2 2 1 1	2 1 1

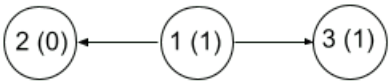
Note

Explanation for the example

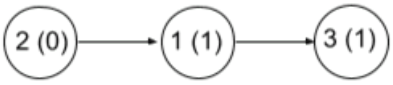
Following image contains the initial tree. Inside the brace, you will find the value of that node. The arrows of the edges go from parent to child.



First update is "1 1". It flips the value of the node 1. Alu needs two operations to make the entire tree zero. Operation on node 2, followed by operation on node 1.



Second update is "2 2". It makes the node 2 as the root of the tree. Alu needs only one operation on node 1 to make the entire tree zero.



Final update is "1 1". It flips the value of the node 1. Alu needs only one operation on node 3 to make the entire tree zero.