Birthday Present

Today is problem setter's best friend Jenny’s birthday. Long ago, Jenny, being a very clever girl, asked the problem setter to perform some queries on a tree but he couldn’t do it. Now, he seeks your help to impress her on her birthday by answering those queries.

Recall that a tree is a connected acyclic undirected graph with **n** nodes and **n-1** edges. In each node there are some flowers. The two type of queries are described as:

1 u v x

2 u y

For first type, you have to calculate the minimum number of vertices needed to be visited on the path from **v** to **u**, starting at **v**, to collect at least **x(1<=x<=1e18)** flowers, where **v** is a descendant of **u**. **Note that you cannot visit any node that is not in the path from v to u and you cannot skip any node of the path from v to that node you've chosen at last.** If it's impossible to collect at least **x** flowers visiting all the vertices from **v** to **u** then you have to print **-1**.

For second type, you have to add y**(y can be negative)** to the existing amount flowers in node **u**. It is guaranteed that after this operation, flowers in a node will be non-negative and sum of flowers of all node of the tree will be at most 10^18.

Note that 1 is the root of the tree.

Input

The first line of the input contains two integers **n**(2 <= n <= 10^5) and **q**(1 <= q <= 10^5) where n is the number of vertices of the tree and q is the number of queries you have to perform.

Each of the next n-1 lines contains two integers **a**(1 <= a <= n) and **b**(1 <= b <= n) which denote an edge between a and b. The next line contains n non-negative integers c[1],c[2],..,c[n] (0 <= c[i] <= 10^13) where c[i] denotes the number of flowers in i’th node. Next q lines contain queries of the format described above.

Output

For each query of the first type print minimum number of nodes you have to visit to collect at least **x**(1 <= x <= 10^18) flowers. If it's impossible to collect at least **x** flowers visiting all the vertices from **v** to **u** then print **-1**.

Example

Input:
6 5
1 2
1 3
2 4
2 5
5 6
2 3 13 5 7 11
1 1 6 10
1 1 6 12
1 1 6 19
2 5 5
1 1 6 23

Output:
1
2
3
2