Dynamic Tree Test

Time limit: 1.4s **Memory limit:** 256M

Today, we'll be practicing modifications on a tree!

Input

The first line contains two integers, N and M, denoting that there are N vertices and M queries.

Then there are $N-1$ lines, each line containing two integers x and y, denoting that there is an edge between x and y in the tree.

Then there are N more lines, each containing one number: the initial weight of each vertex.

Then the next line contains the root.

Then there are M lines:

The first number is K.

$K=0$ means subtree modification. K is followed by x and y. This operation sets all vertex weights in the subtree of x to y.

$K=1$ means change root. The line contains one additional integer x, representing the new root of the tree.

$K=2$ means path modification. K is followed by integers x, y, z. This operation sets z as the vertex weight of all vertices on the path from x to y.

$K=3$ means subtree min. K is followed by x, the root of the queried subtree.

$K=4$ means subtree max. K is followed by x, the root of the queried subtree.

$K=5$ means increment subtree. K is followed by x and y, the root of the queried subtree and the value to increment by.

$K=6$ means path increment. K is followed by x, y, z. This operation increments all vertex weights on the path from x to y by z.

$K=7$ means path min. K is followed by x and y, the endpoints of the queried path.

$K=8$ means path max. K is followed by x and y, the endpoints of the queried path.

$K=9$ means change parent. K is followed by x and y. This operation changes the parent of x to y. If y is in the subtree of this operation, do nothing.

$K=10$ means path sum. K is followed by x and y, and asks for the sum of the weights on the path from x to y.

$K=11$ means subtree sum. K is followed by x, and asks for the sum of the weights in the subtree root at x.

Output

Print an answer for each query. All answers go on their own lines.

Sample Input 1

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

Sample Output 1

```
9
1
1
```

Sample Input 2

```
10 12
2 1
3 2
4 2
5 3
6 4
7 5
8 2
9 4
10 9
791
868
505
658
860
623
393
717
410
173
4
0 8 800
1 4
2 8 2 103
3 9
4 4
5 7 304
6 8 8 410
7 10 8
8 1 8
9 6 9
10 2 3
11 5
```

Sample Output 2

```
173
860
103
791
608
1557
```

Constraints

$N, M \leq 10^5$

All intermediate values can be stored in a C++ `int`.