B. Apple Tree

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a rooted tree with n vertices. In each leaf vertex there's a single integer — the number of apples in this vertex.

The *weight* of a subtree is the sum of all numbers in this subtree leaves. For instance, the weight of a subtree that corresponds to some leaf is the number written in the leaf.

A tree is *balanced* if for every vertex v of the tree all its subtrees, corresponding to the children of vertex v, are of equal weight.

Count the minimum number of apples that you need to remove from the tree (specifically, from some of its leaves) in order to make the tree balanced. Notice that you can always achieve the goal by just removing all apples.

Input

The first line contains integer n $(2 \le n \le 10^5)$, showing the number of vertices in the tree. The next line contains n integers $a_1, a_2, ..., a_n$ $(0 \le a_i \le 10^8)$, a_i is the number of apples in the vertex number i. The number of apples in non-leaf vertices is guaranteed to be zero.

Then follow $n - 1$ lines, describing the tree edges. Each line contains a pair of integers x_i, y_i $(1 \le x_i, y_i \le n, x_i \ne y_i)$ — the vertices connected by an edge.

The vertices are indexed from 1 to n. Vertex 1 is the root.

Output

Print a single integer — the minimum number of apples to remove in order to make the tree balanced.

Please, do not write the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the sin, cout streams `cin`, `cout` or the `%I64d` specifier.

Examples

input
6 0 0 12 13 5 6 1 2 1 3 1 4 2 5 2 6
output
6