第 3 题：米特运输(meat)，运行时限 2s，内存上限 128M，100 分。

【问题描述】

米特是 D 星球上一种非常神秘的物质，蕴含着巨大的能量。在以米特为主要能源的 D 星上，这种米特能源的运输和储存一直是一个大问题。

D 星上有 N 个城市，我们将其顺序编号为 1 到 N，1 号城市为首都。这 N 个城市由 N-1 条**单向**高速通道连接起来，构成一棵以 1 号城市（首都）为根的树，高速通道的方向由树中的**儿子指向父亲**。树按深度分层：根结点深度为 0，属于第 1 层；根结点的子节点深度为 1，属于第 2 层；依此类推，深度为 i 的结点属于第 i+1 层。

建好高速通道之后，D 星人开始考虑如何具体地储存和传输米特资源。由于发展程度不同，每个城市储存米特的能力不尽相同，其中第 i 个城市建有一个容量为 A[i]的米特储存器。这个米特储存器除了具有储存的功能，还具有自动收集米特的能力。如果到了晚上六点，有某个储存器处于未满的状态，它就会自动收集大气中蕴含的米特能源，在早上六点之前就能收集满；但是，只有在储存器完全空的状态下启动自动收集程序才是安全的，未满而又非空时启动可能有安全隐患。早上六点到七点间，根节点城市（1 号城市）会将其储存器里的米特消耗殆尽。根节点不会自动搜集米特，它只接受子节点传输来的米特。早上七点，城市之间启动米特传输过程，传输过程逐层递进：先是第 2 层节点城市向第 1 层（根节点城市，即 1 号城市）传输，直到第 1 层的储存器满或第 2 层的储存器全为空；然后是第 3 层向第 2 层传输，直到对于第 2

层的每个节点，其储存器满或其子节点（位于第 3 层）的储存器全为空；依此类推，直到最后一层传输完成。传输过程一定会在晚上六点前完成。

由于技术原因，运输方案需要满足以下条件：（1）**不能**让某个储存器到了晚上六点传输结束时还处于**非空但又未满**的状态，这个时候储存器仍然会启动自动收集米特的程序，而给已经储存有米特的储存器启动收集程序可能导致危险，也就是说要让储存器到了晚上六点时要么空要么满；（2）关于首都——即 1 号城市的特殊情况， 每天早上六点到七点间 1 号城市中的米特储存器里的米特会自动被消耗殆尽，即运输方案不需要考虑首都的米特怎么运走；（3）除了 1 号城市，每个节点必须在其子节点城市向它运输米特之前将这座城市的米特储存器中原本存有的米特**全部**运出去给父节点，不允许储存器中残存的米特与外来的米特发生混合；（4）运向某一个城市的若干个来源的米特数量必须**完全相同**，不然，这些来源不同的米特按不同比例混合之后可能发生危险。

现在 D 星人已经建立好高速通道，每个城市也有了一定储存容量的米特储存器。为了满足上面的限制条件，可能需要重建一些城市中的米特储存器。你可以，也只能，将某一座城市（包括首都）中原来存在的米特储存器摧毁，再新建一座任意容量的新的米特储存器，其容量可以是小数（在输入数据中，储存器原始容量是正整数，但重建后可以是小数），不能是负数或零，使得需要被重建的米特储存器的数目尽量少。

【输入格式】

输入文件名为meat.in。

第一行是一个正整数N，表示城市的数目。

接下来N行，每行一个正整数，其中的第i行表示第i个城市原来存在的米特储存器的容量。

再接下来是N-1行，每行两个正整数a,b表示城市b到城市a有一条高速通道(a≠b)。

【输出格式】

输出文件名为meat.out。

输出文件仅包含一行，一个整数，表示最少的被重建（即修改储存器容量）的米特储存器的数目。

【输入输出样例】

meat.in	meat.out
5 5 4 3 2 1 1 2 1 3 2 4 2 5	3

【样例解释】

一个最优解是将A[1]改成8，A[3]改成4，A[5]改成2。这样，2和3运给1的量相等，4和5运

给2的量相等，且每天晚上六点的时候，1,2满，3,4,5空，满足所有限制条件。

【数据范围】

对于20%的数据满足N≤20；

对于50%的数据满足N≤2000；

对于100%的数据满足N≤500000, A[i]≤10^8。

【编译命令】

对于c++语言： g++ -o meat meat.cpp –lm

对于c语言： gcc -o meat meat.c –lm

对于pascal语言： fpc meat.pas