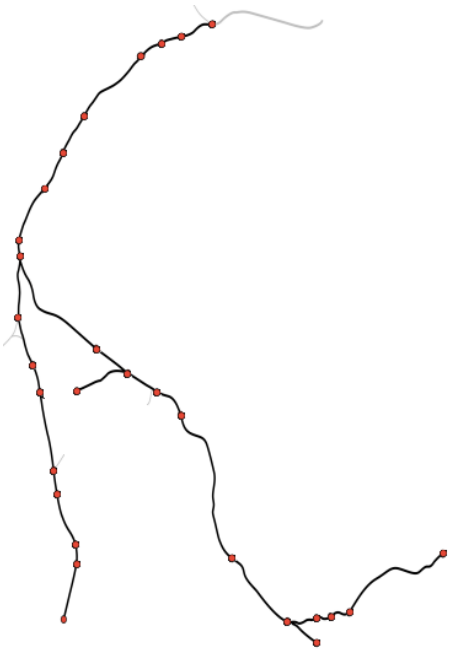
8114 Rainbow Roads



The Transit Authority of Greater Podunk is planning its holiday decorations. They want to create an illuminated display of their light rail map in which each stretch of track between stations an be illuminated in one of several colors.

At periodic intervals, the controlling software will choose two stations at random and illuminate all of the segments connecting those two stations. By design, for any two stations on the Greater Podunk Railway, there is a unique path connecting the two.

For maximum color and cheer, the display designers want to avoid having two adjacent segments of track lighting up in the same color. They fear, however, that they may have deviated from this guideline in the process of building the display. One of them has gone so far as to propose a means of measuring just how far from that ideal they may have fallen.

You are given a tree with n nodes (stations), onveniently numbered from 1 to n. Each edge in this tree has one of n colors. A path in this tree is called a *rainbow* if all edges adjacent to the path have different colors. Also, a node is alled *good* if every simple path with that node as one of its endpoints is a rainbow path. (A simple path is a path that does not repeat any vertex or edge.)

Your team is to write a program that will find all the *good* nodes in the given tree.

Input

The input file contains several test cases, each of them as described below.

The first line of input contains a single integer n ($1 \leq n \leq 50,000$). Each of the next $n-1$ lines contains three space-separated integers a_i, b_i, and c_i ($1 \leq a_i, b_i, c_i \leq n$; $a_i \neq b_i$), describing an edge of color c_i that connects nodes a_i and b_i.

It is guaranteed that the given edges form a tree.

Output

For each test case, the output must follow the description below.

Your program should first print a line containing k, the number of good nodes. On the next k lines, print the indices of all good nodes in increasing numerical order, one per line. No leading or trailing whitespace or leading zeroes are to appear on an output line.

Note: For the first sample below, node 3 is good because all paths that have node 3 as an endpoint are rainbow. In parti ular, even though the path $3 \to 4 \to 5 \to 6$ has two edges of the same color (i.e. $3 \to 4, 5 \to 6$), it is still rainbow because these edges are not adjacent.

Sample Input

```
8
1 3 1
2 3 1
```

```
3 4 3
4 5 4
5 6 3
6 7 2
6 8 2
8
1 2 2
1 3 1
2 4 3
2 7 1
3 5 2
5 6 2
7 8 1
10
9 2 1
9 3 1
9 4 2
9 5 2
9 1 3
9 6 4
1 8 5
1 10 5
6 7 9
```

Sample Output

```
4
3
4
5
6
0
4
1
6
7
9
```