



Problem B: Stones on a Tree

Time limit: 1.0s; Memory limit: 256 MB

You are given a tree with n vertices. You have to put some stones on every vertex: a_i stones on vertex i , where $0 \leq a_i \leq 10^9$. The sequence $(a_1, a_2, \dots, a_n)$ is called a placement.

For a fixed placement, Alice and Bob play the following game with one token:

- First, Alice chooses a starting vertex and puts the token on it. This is not a move.
- Then the players take turns making moves, Alice moves first.
- Let v be the vertex with the token. In one move, the player does exactly one of the following:
 - Take: remove one or more stones from vertex v , the token stays on v .
 - Jump: move the token to a vertex u adjacent to v such that the token has never been on u before (the starting vertex counts as visited), and u currently has exactly as many stones as v . No stones are removed during a jump.
- The player who cannot make a move loses.

A placement is good if Bob wins no matter which starting vertex Alice chooses, assuming both players play optimally.

Count the number of good placements. Since the answer may be large, print it modulo 998,244,353.

Input

- The first line contains a single integer n ($1 \leq n \leq 5000$) — the number of vertices.
- Each of the next $n - 1$ lines contains two integers u and v ($1 \leq u, v \leq n, u \neq v$) — an edge of the tree.
- It is guaranteed that the given edges form a tree.

Output

Print a single integer — the number of good placements modulo 998,244,353.

Sample

Sample 1

Sample Input	Sample Output
3 1 2 2 3	2

Sample 2

Sample Input	Sample Output
7 1 2 1 3	8



1 4 2 5 2 6 3 7	
--------------------------	--

Sample 3

Sample Input	Sample Output
8 1 2 1 3 2 4 1 5 2 6 3 7 4 8	16

Note

In the first test, the tree is the path 1 - 2 - 3, and the good placements are (0, 1, 0) and (1, 0, 1). Let us check that (0, 1, 0) is good:

- Alice starts at vertex 1. Vertex 1 has no stones, so Alice cannot take. Vertex 2 has 1 stone, not 0, so Alice cannot jump. Alice loses immediately.
- Alice starts at vertex 3. This is the same as vertex 1.
- Alice starts at vertex 2. Alice cannot jump, because vertex 2 has 1 stone and its neighbours have 0. So Alice takes the stone, and now vertex 2 has 0 stones. Bob jumps to vertex 1, since both vertices have 0 stones. Alice cannot take (vertex 1 is empty) and cannot jump (vertex 2 is already visited), so Alice loses.

The placement (0, 0, 0) is not good: Alice starts at vertex 2 and immediately jumps to vertex 1, since both vertices have 0 stones. Bob cannot take and cannot jump back to vertex 2, so Bob loses.