B Balance by Elimination

Time limit: 3s

CC-BY 2.0 By Floyd Wilde on Flickr
Unidentified bright green tree

You are given a binary tree with n nodes. The nodes are conveniently numbered from 1 to n. Node 1 is the root of the binary tree.

The height of the subtree rooted at node u is:

$$h_u = 1 + \max\left(h_{\text{left child}}, h_{\text{right child}}\right)$$

If a left or right child doesn't exist, its subtree height is defined to be 0. In particular, if a node is a leaf, it has a height of 1.

You want the tree to become height-balanced. A node is height-balanced if:

$$|h_{\text{left child}} - h_{\text{right child}}| < 2$$

A binary tree is height-balanced if all its nodes are height-balanced.

Find a way to remove at most 1 leaf from the tree, such that the binary tree becomes height-balanced, or output that this is impossible. For example, the tree of the second sample input (visualized in Figure B.1) becomes balanced when removing node 5.

Input

The input consists of:

- One line containing a single integer n ($1 \leq n \leq 10^5$), the number of nodes in the binary tree.
- Then n lines follow, numbered from 1 to n. The ith line contains two integers, the labels of the left and right child of node i.

If a left child or right child does not exist, the corresponding integer is equal to 0. It is guaranteed that the input graph is a binary tree.

Output

Output a single integer:

- If the tree is already balanced, output "`balanced`".
- If it's impossible to make the tree height-balanced, output "`impossible`".
- Else, output the number of the leaf you want to remove.

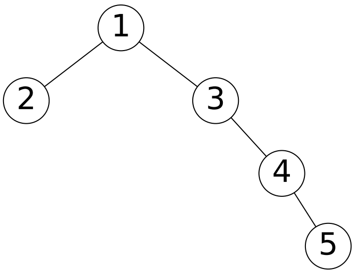

Figure B.1: Visualization of Sample Input 2.

Sample Input 1

```
4
4 2
0 0
0 0
0 3
```

Sample Output 1

```
balanced
```

Sample Input 2

```
5
2 3
0 0
0 4
0 5
0 0
```

Sample Output 2

```
5
```

Sample Input 3

```
4
2 0
3 0
4 0
0 0
```

Sample Output 3

```
impossible
```