Problem C. DFS Order 2

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 seconds
Memory limit:	512 megabytes

Prof. Pang has a rooted tree that is rooted at vertex 1 and has n nodes. These n nodes are numbered from 1 to n.

Now he wants to start the depth-first search at the root. He wonders for each node v, how many ways it can appear in the j-th position of **depth-first search order**. The depth-first search order is the order of nodes visited during the depth-first search. A node appears in the j-th ($1 \le j \le n$) position in this order means it is visited after $j-1$ other nodes. Because sons of a node can be iterated in arbitrary order, multiple possible depth-first orders exist.

Prof. Pang wants to know for each node v, how many different **depth-first search order**s such that v appears in the j-th position. For each v, j ($1 \le v, j \le n$), compute the answer. Because the answer can be very large, output it modulo 998244353.

Following is a pseudo-code for the depth-first search on a rooted tree. After calling MAIN(), `dfs_order` is the depth-first search order.

Algorithm 1 An implementation of depth-first search

```
procedure DFS(vertex x)
    Append x to the end of dfs_order
    for each son y of x do                ▷ Sons can be iterated in arbitrary order.
        DFS(y)
    end for
end procedure
procedure MAIN()
    Let dfs_order be a global variable
    dfs_order ← empty list
    DFS(1)
end procedure
```

Input

The first line contains one integer n ($1 \le n \le 500$), the number of vertices in the tree.

Each of the next $n-1$ lines describes an edge of the tree. Edge i is denoted by two integers u_i and v_i, the labels of vertices it connects ($1 \le u_i, v_i \le n, u_i \ne v_i$).

It is guaranteed that the given edges form a tree.

Output

For each vertex v from 1 to n, output one line containing n integers modulo 998244353. The j-th integer in the v-th line should be the number of different depth-first search orders such that v appears in the j-th position.

Example

standard input	standard output
5 1 2 1 3 3 4 3 5	4 0 0 0 0 0 2 0 0 2 0 2 2 0 0 0 0 1 2 1 0 0 1 2 1