Problem J
Jumbled Graph

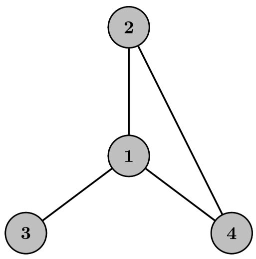
Let's revisit a very basic problem of graph theory: given an connected undirected graph of n vertices (numbered 1 to n), what is the depth-first-search (DFS) order of this graph. The DFS order could be generated by the following pseudocode:

```
dfs_order = []  # empty list

def DFS(u):
    visited[u] = True
    dfs_order.append(u)
    for vertex v that is directly connected to u:
        # note that the order of v is totally random
        if visited[v] == False:
            DFS(v)

DFS(random(1,n))
```

Thus, there are 12 valid DFS orders for the following graph:



1. `1,2,4,3`
2. `1,3,2,4`
3. `1,3,4,2`
4. `1,4,2,3`
5. `2,1,3,4`
6. `2,1,4,3`
7. `2,4,1,3`
8. `3,1,2,4`
9. `3,1,4,2`
10. `4,1,2,3`
11. `4,1,3,2`
12. `4,2,1,3`

Given a permutation of 1 to n, your task is to determine how many undirected graph takes this permutation as a valid DFS order? Note that, two graphs $G1$ and $G2$ are considered different iff there exists two vertices u and v ($u \neq v$) where $G1$ contains an edge between u and v and $G2$ does not contains it; or vice versa.

Input

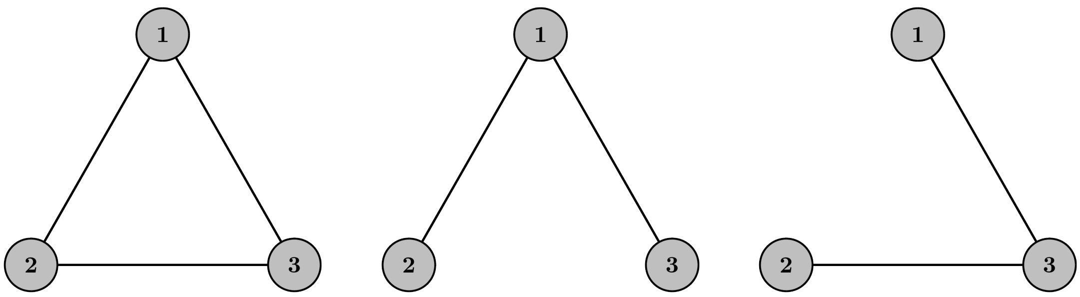
- The first line contains an integer n ($1 \leq n \leq 16$) — the number of vertices.
- The second line contains n integers $a_1, a_2, \ldots a_n$ representing a DFS order. It is guaranteed that $(a_1, a_2, \ldots a_n)$ is a permutation of $(1, 2, \ldots, n)$.

Output

You need to print the number of graphs that takes this permutation to be its DFS order modulo $998\,244\,353$.

Sample explanation

- In the first sample, there is only one graph — the complete graph takes `2,1` as a valid DFS order.
- In the second sample, these 3 graphs takes `3,1,2` as a valid DFS order.



Sample Input 1	Sample Output 1
2 2 1	1

Sample Input 2	Sample Output 2
3 3 1 2	3