Double Agents (`trees`)

The British spy agency MI6 is planning to infiltrate a network of double agents into the sinister criminal organisation SPECTRE. SPECTRE has N employees, numbered from 0 to $N-1$, and their organisation chart is a tree on these N vertices.

MI6 will select a non-empty set S of SPECTRE employees to turn into double agents. Because of the risk of further treachery (i.e. possible triple-agents), it is essential that the chain of communication between any two double-agents passes only through 'innocent' employees. That is, for any two distinct employees a and b in S, the simple path between a and b in the tree must not contain any other employees in S.

Your task is to count the number of possible sets S of double agents. Since this value can be quite large, output the value modulo $10^9 + 7$.

Implementation

You will have to submit a single `.cpp` source file.

> ☞ Among this task's attachments you will find a template `trees.cpp` with a sample implementation.

You have to implement the following function:

C++
```
int count_sets(int N, vector<int> U, vector<int> V);
```

- Integer N represents the number of employees of SPECTRE.
- The arrays U and V, indexed from 0 to $N-2$, contains the values $U_0, U_1, \ldots, U_{N-2}$ and $V_0, V_1, \ldots, V_{N-2}$, where U_i and V_i are the endpoints of the i-th edge in the organisation tree.
- The function should return the number of possible sets modulo $10^9 + 7$.

The grader will call the function `count` and will print its return value to the output file.

Sample Grader

The task's directory contains a simplified version of the jury grader, which you can use to test your solution locally. The simplified grader reads the input data from `stdin`, calls the functions that you must implement, and finally writes the output to `stdout`.

The input is made up of N lines, containing:

- Line 1: the integer N.
- Line $2+i$ ($0 \le i \le N-2$): the integers U_i and V_i.

The output is made up of a single line, containing the value returned by the function `count`.

Constraints

- $2 \le N \le 500\,000$.
- $0 \le U_i, V_i \le N-1$.
- The nodes form a valid tree.

Scoring

Your program will be tested on a set of test cases grouped by subtask. To obtain the score associated to a subtask, you need to correctly solve all the test cases it contains.

- **Subtask 1** [**0 points**]: Sample test cases.
- **Subtask 2** [**20 points**]: $N \leq 16$.
- **Subtask 3** [**15 points**]: The tree is a path. A path is a tree where the nodes can be arranged in some order $P_0, P_1, \ldots, P_{N-1}$ in which there exists an edge between nodes P_i and P_{i+1} for all $0 \leq i \leq N-2$.
- **Subtask 4** [**15 points**]: $N \leq 500$, and the tree is a caterpillar. A caterpillar is a path with some additional leaf nodes attached. That is, the vertices of the tree can be written as $P_0, \ldots, P_k, Q_0, \ldots, Q_l$, where $P_0, \ldots, P_k$ is a path and each Q_i is a leaf (i.e. has degree 1).
- **Subtask 5** [**15 points**]: $N \leq 5000$, and the tree is a caterpillar.
- **Subtask 6** [**10 points**]: $N \leq 500\,000$, and the tree is a caterpillar.
- **Subtask 7** [**25 points**]: No additional constraints.

Examples

stdin	stdout
3 0 1 1 2	6
5 0 1 0 2 1 3 1 4	17

Explanation

In the **first sample case**, the tree is a path $(0 - 1 - 2)$. There are 6 possible sets in this tree: $\{0\}, \{1\}, \{0,1\}, \{2\}, \{0,2\}, \{1,2\}$.

The set $\{0,1,2\}$ is not permitted as 1 lies on the simple path between 0 and 2.

In the **second sample case**, the tree is a path $(3 - 1 - 0 - 2)$ with a single additional leaf (4). There are 17 possible sets in this tree.