Problem Link:

<https://leetcode.com/problems/find-eventual-safe-states/?envType=daily-question&envId=2025-01-24>

Solution:

class Solution {

public:

vector<int> eventualSafeNodes(vector<vector<int>>& graph) {

int n = graph.size();

vector<int> state(n, 0);

vector<int> result;

function<bool(int)> dfs = [&](int node) {

if(state[node] == 1)

return false;

if(state[node] == 2)

return true;

state[node] = 1;

for(int neighbor : graph[node])

{

if(!dfs(neighbor))

{

state[node] = 3;

return false;

}

}

state[node] = 2;

return true;

};

for(int i = 0; i < n; ++i)

{

if (dfs(i)) {

result.push\_back(i);

}

}

return result;

}

};