#include <iostream>

#include <vector>

#include <stack>

using namespace std;

const int V = 8;

vector<bool> visited(V, false);

bool IsElementInStack(stack<int> s, int b) {

size\_t size = s.size();

for (int i = 0; i < size; i++) {

if (b == s.top()) {

return true;

}

s.pop();

}

return false;

}

void DFS(vector<vector<int>> G, int a) {

stack<int> s;

s.push(a);

while (!s.empty()) { // removes a node from the stack and visits its children

int b = s.top();

s.pop();

if (!visited.at(b)) { // marks b as visited

visited.at(b) = true;

cout << b << "->";

}

// push adjacent unvisited nodes onto stack

for (int i = 0; i < V; i++) {

if (G.at(b).at(i) == 1 && !visited.at(i)) {

if (IsElementInStack(s, i) == false) {

s.push(i);

}

}

}

}

}

void componentInGraph() {

if (visited.size() < 9) {

cout << "Number 9 is not in graph, graph is too small to contain 9." << endl;

}

else if (visited.at(9)) {

cout << "Number 9 is in the graph." << endl;

}

else {

cout << "Number 9 is not in the graph" << endl;

}

}

void connectComponent(vector<vector<int>> G) {

int sections = 1;

for (int i = 0; i < V; i++) {

if (!visited.at(i)) {

cout << "Section " << sections << ": ";

sections++;

DFS(G, i);

cout << endl;

}

}

}

int main() {

cout << "Depth First Search, time complexity: O(V + E) using an Adjacency List." << endl;

cout << "Where V is the number of Vertexes and E is the number of edges." << endl;

vector<vector<int>> G = {

{ 0, 1, 1, 1, 0, 0, 0, 0 },

{ 1, 0, 0, 0, 0, 0, 0, 0 },

{ 1, 0, 0, 1, 1, 0, 0, 0 },

{ 1, 0, 1, 0, 0, 0, 0, 0 },

{ 0, 0, 1, 0, 0, 0, 0, 0 },

{ 0, 0, 0, 0, 0, 0, 1, 0 },

{ 0, 0, 0, 0, 0, 1, 0, 0 },

{ 0, 0, 0, 0, 0, 0, 0, 0 }

};

connectComponent(G);

componentInGraph();

return 0;

}

Output:

Depth First Search, time complexity: O(V + E) using an Adjacency List.

Where V is the number of Vertexes and E is the number of edges.

Section 1: 0->3->2->4->1->

Section 2: 5->6->

Section 3: 7->

Number 9 is not in graph, graph is too small to contain 9.

A screenshot of a computer

AI-generated content may be incorrect.