import java.util.InputMismatchException;

import java.util.Scanner;

import java.util.Stack;

public class ConnectedGraph {

private final Stack<Integer> stack;

public ConnectedGraph() {

stack = new Stack<>();

}

public String dfs(int matrix[][], int source) {

int num\_of\_nodes = matrix[source].length - 1;

int visited[] = new int[num\_of\_nodes + 1];

int element = source;

int i = source;

visited[source] = 1;

stack.push(source);

while (!stack.isEmpty()) {

element = stack.peek();

i = element;

while (i <= num\_of\_nodes) {

if (matrix[element][i] == 1 && visited[i] == 0) {

stack.push(i);

visited[i] = 1;

element = i;

i = 1;

continue; }

i++; }

stack.pop(); }

boolean connected = false;

for (int vertex = 1; vertex <= num\_of\_nodes; vertex++) {

if (visited[vertex] == 1) {

connected = true;

}

else {

connected = false;

break; } }

if (connected){

System.out.println("The graph is connected");

return "is connected";

}else{

System.out.println("The graph is disconnected");

return "not connected";

} }

public static void main(String []arg) {

int num\_of\_nodes, source;

Scanner scanner = null;

try {

System.out.println("Enter the number of nodes in the graph");

scanner = new Scanner(System.in);

num\_of\_nodes = scanner.nextInt();

int matrix[][] = new int[num\_of\_nodes + 1][num\_of\_nodes + 1];

System.out.println("Enter the adjacency matrix");

for (int i = 1; i <= num\_of\_nodes; i++)

for (int j = 1; j <= num\_of\_nodes; j++)

matrix[i][j] = scanner.nextInt();

System.out.println("Enter the source for the graph");

source = scanner.nextInt();

ConnectedGraph directedConnectivity= new ConnectedGraph();

directedConnectivity.dfs(matrix, source);

}catch(InputMismatchException inputMismatch) {

System.out.println("Wrong Input format");

} scanner.close();

} }