**LAB-9**

import java.util.Scanner;

class Main {

private int a[][];

private int N;

Main (int V)

{

N=V;

a = new int[N][N];

}

void RecursiveDFS(int s, boolean visited[],int a[][])

{

visited[s] = true;

System.out.print(s+ " ");

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

{

if ((!visited[i])&&(a[s][i]==1))

{

RecursiveDFS(i,visited,a);

}

}

}

void DFS(int s,int a[][])

{

boolean visited[] = new boolean[N];

RecursiveDFS(s,visited,a);

}

public static void main(String[] args) {

Main g = new Main(5);

int a[][] = new int[5][5];

Scanner sc = new Scanner(System.in);

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

for (int i=0;i<5;i++)

{

for (int j=0;j<5;j++)

{

int x = sc.nextInt();

a[i][j] = x;

}

}

System.out.println("Following is Depth First Traversal "+

"(starting from vertex 2)");

g.DFS(0,a);

}

}

m,n = map(int, input().split())

graph = [[0 for i in range(m)] for j in range(n)]

print(graph)

for i in range(m):

for j in range(n):

graph[i][j]=int(input("enter the value for "+str(i+1)+str(j+1)))

print(graph)

def bfs(graph, root):

visited, queue = set(), [root]

while queue:

tt = queue.pop(0)

for h in range(n):

if graph[tt][h]== 1:

if h not in visited:

visited.add(h)

queue.append(h)

l = []

for kk in visited:

l.append(kk+1)

print(l)

bfs(graph,0