| Program No: | 21 |
| --- | --- |
| Roll No: | 1510 |
| Title of Program: | Unit 6: Graph |
| Objective: | Implement Graphs   1. Use adjacency Matrix 2. To illustrate Depth First iteration |

**CODE for adjacency Matrix:**

import java.util.Scanner;

class AdjMat {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

// input vertices

System.out.print("Enter the amount of vertices: ");

int v = sc.nextInt();

int[][] mat = new int[v][v];

System.out.print("Enter the number of edges: ");

int edges = sc.nextInt();

for (int i = 1; i <= edges; i++) {

System.out.print("Enter the Source: ");

int source = sc.nextInt();

System.out.print("Enter the Destination: ");

int dest = sc.nextInt();

mat[source][dest] = 1;

mat[dest][source] = 1;

}

// display

System.out.println("The Adjacency Matrix: ");

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

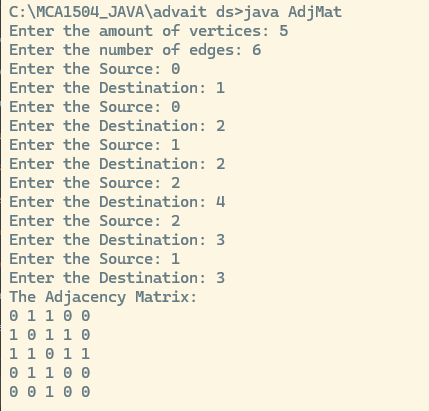
for (int j = 0; j < v; j++) {

System.out.print(mat[i][j] + " ");

}

System.out.println(); } }}

**OUTPUT:**

****

**Code for Depth First Search:**

class DFT {

int[][] adj;

boolean[] visited;

int[] stack;

int tos;

public DFT(int v) {

adj = new int[v][v];

visited = new boolean[v];

stack = new int[v];

tos = -1;

}

void addEdge(int src, int dest) {

adj[src][dest] = 1;

adj[dest][src] = 1;

}

void performDFT(int x) {

push(x);

System.out.println("Depth First Traversal: ");

while (tos != -1) {

int curr = pop();

if (!visited[curr]) {

visited[curr] = true;

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

for (int i = adj.length - 1; i >= 0; i--) {

if (adj[curr][i] == 1 && !visited[i]) {

push(i);

}

} // end for

} // end if

}

} // end of DFT

void push(int node) {

tos++;

stack[tos] = node;

}

int pop() {

int tmp = stack[tos];

tos--;

return tmp;

}

public static void main(String[] args) {

DFT g = new DFT(5);

g.addEdge(0, 1);

g.addEdge(0, 2);

g.addEdge(0, 3);

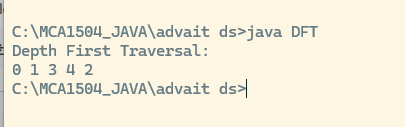
g.addEdge(1, 3);

g.addEdge(2, 4);

g.addEdge(3, 4);

g.performDFT(0); }}

**OUTPUT:**

****