**// Program04:** Write and execute a menu driven program to print all the nodes reachable from a given starting node in a graph to arrange nodes in topological order using DFS method and source removal technique.

#include <stdio.h>

#include <stdlib.h>

int s[100], count, res[100];

//-------------------------------------------------------------------------------------

void dfs(int n, int a[100][100], int source)

{

int i;

s[source] = 1;

for (i = 1; i <= n; i++) {

if (a[source][i] == 1 && s[i] == 0)

dfs(n, a, i);

}

count = count + 1;

res[count] = source;

}

//-------------------------------------------------------------------------------------

void topologicalDFS(int n, int a[100][100]) {

int i, j;

for (i = 1; i <= n; i++)

{

s[i] = 0;

}

count = 0;

for (j = 1; j <= n; j++)

{

if (s[j] == 0)

{

dfs(n, a, j);

}

}

for (i = n; i >= 1; i--)

{

printf("--> %d ", res[i]);

}

printf("\n");

}

void topologicalSourceRemoval(int n, int a[100][100], int sc[100]) {

int i, j;

for (i = 1; i <= n; i++) {

if (sc[i] == 0) {

printf("%d\t", i);

sc[i] = -1;

for (j = 1; j <= n; j++) {

if (a[i][j] == 1) {

sc[j] = sc[j] - 1;

}

}

i = 0;

}

}

printf("\n");

}

//---------------------------------------------------------------------------------------

int main()

{

int a[100][100], n, i, j, sc[100],ch;

printf("Enter number of vertices: ");

scanf("%d", &n);

printf("Enter adjacency matrix:\n");

for (i = 1; i <= n; i++)

for (j = 1; j <= n; j++)

scanf("%d", &a[i][j]);

printf("1:Topological order using DFS 2: Source Removal method:\n");

scanf("%d",&ch);

switch(ch)

{

case 1: printf("Topological order using DFS:\n");

topologicalDFS(n, a);

break;

case 2: for (i = 1; i <= n; i++)

sc[i] = 0;

for (i = 1; i <= n; i++)

for (j = 1; j <= n; j++)

if(a[i][j] == 1)

{

sc[j] = sc[j] + 1;

}

printf("Topological order using Source Removal:\n");

topologicalSourceRemoval(n, a, sc);

break;

Enter number of vertices: 7

Enter adjacency matrix:

0 1 1 0 0 0 0

0 0 0 0 1 0 1

0 0 0 0 0 1 0

1 1 1 0 0 1 1

0 0 0 0 0 0 0

0 0 0 0 0 0 0

0 0 0 0 1 1 0

1:Topological order using DFS 2: Source Removal method:

2

Topological order using Source Removal:

4 1 2 3 7 5 6

default: exit(0);

}

}

Enter number of vertices: 7

Enter adjacency matrix:

0 1 1 0 0 0 0

0 0 0 0 1 0 1

0 0 0 0 0 1 0

1 1 1 0 0 1 1

0 0 0 0 0 0 0

0 0 0 0 0 0 0

0 0 0 0 1 1 0

1:Topological order using DFS 2: Source Removal method:

1

Topological order using DFS:

--> 4 --> 1 --> 3 --> 2 --> 7 --> 6 --> 5

// Program 4a: Write and execute a program to arrange nodes in topological order using source removal technique.

#include<stdio.h>

// Topology Function

void topology(int n, int a[10][10], int sc[10]) {

int i, j;

for(i = 1; i <= n; i++) {

if(sc[i] == 0) {

printf("%d\t",i);

sc[i] = -1;

for(j = 1; j <= n; j++)

if(a[i][j] == 1)

sc[j] = sc[j] - 1;

i = 0;

}

}

}

// Main Program

int main() {

int a[10][10], n, i, j, sc[10];

printf("Enter the number of nodes:\n");

scanf("%d", &n);

for(i = 1; i <= n; i++)

sc[i] = 0;

printf("Enter the adjacency matrix:\n");

for(i = 1; i <= n; i++) {

for(j = 1; j <= n; j++) {

scanf("%d", &a[i][j]);

if(a[i][j] == 1)

sc[j] = sc[j] + 1;

}

}

printf("The Topological Order of graph is:\n");

topology(n, a, sc);

}

Enter number of vertices: 7

Enter adjacency matrix:

0 1 1 0 0 0 0

0 0 0 0 1 0 1

0 0 0 0 0 1 0

1 1 1 0 0 1 1

0 0 0 0 0 0 0

0 0 0 0 0 0 0

0 0 0 0 1 1 0

1:Topological order using DFS 2: Source Removal method:

1

Topological order using DFS:

--> 4 --> 1 --> 3 --> 2 --> 7 --> 6 --> 5