Data Structure and Algorithm (MCA 271)

Lab Practical –

BY

Himanshu Heda (24225013)

SUBMITTED TO

Prof. Vandna Kansal

SCHOOL OF SCIENCES

2024-2025

Program Description:

Code of the program

Output: - Paste the o/p of the program.

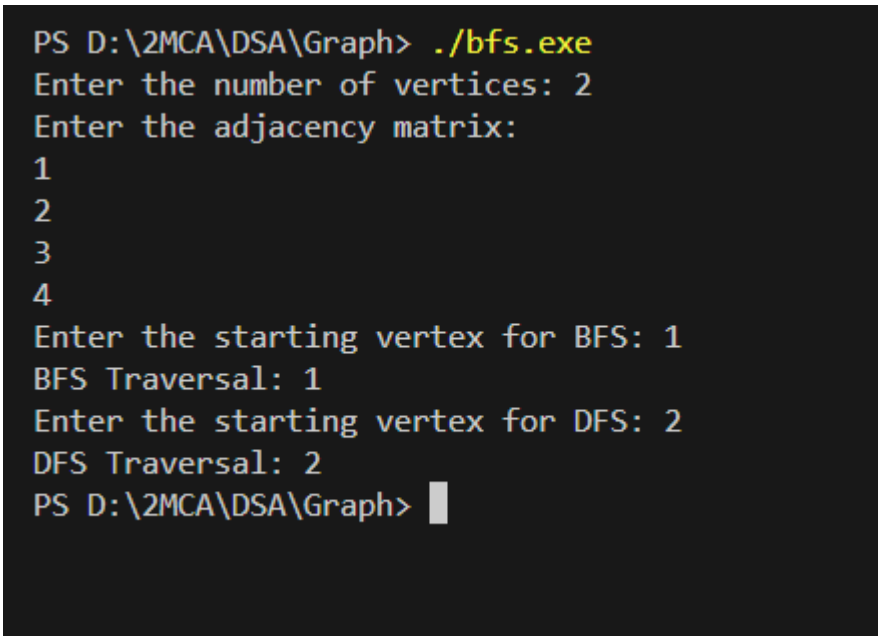
1. BFS & DFS : --

```c
#include <stdio.h>
#include <stdlib.h>

#define MAX 100

int adj[MAX][MAX];
int visited[MAX];
int queue[MAX], front = -1, rear = -1;
int stack[MAX], top = -1;

void enqueue(int vertex) {
    if (rear == MAX - 1)
        printf("\nQueue Overflow\n");
    else {
        if (front == -1)
            front = 0;
        queue[++rear] = vertex;
    }
}

int dequeue() {
    int vertex;
    if (front == -1 || front > rear) {
        printf("\nQueue Underflow\n");
        return -1;
    } else {
        vertex = queue[front++];
        if (front > rear)
            front = rear = -1;
        return vertex;
    }
}

void push(int vertex) {
    if (top == MAX - 1)
        printf("\nStack Overflow\n");
    else
        stack[++top] = vertex;
}
```

```
int pop() {
    if (top == -1) {
        printf("\nStack Underflow\n");
        return -1;
    } else
        return stack[top--];
}

void bfs(int startVertex, int n) {
    int i, vertex;
    for (i = 0; i < n; i++)
        visited[i] = 0;

    enqueue(startVertex);
    visited[startVertex] = 1;

    while (front != -1) {
        vertex = dequeue();
        printf("%d ", vertex);

        for (i = 0; i < n; i++) {
            if (adj[vertex][i] == 1 && !visited[i]) {
                enqueue(i);
                visited[i] = 1;
            }
        }
    }
}

void dfs(int startVertex, int n) {
    int i, vertex;
    for (i = 0; i < n; i++)
        visited[i] = 0;

    push(startVertex);
    visited[startVertex] = 1;
    printf("%d ", startVertex);

    while (top != -1) {
        vertex = stack[top];
```

```
        for (i = 0; i < n; i++) {
            if (adj[vertex][i] == 1 && !visited[i]) {
                push(i);
                visited[i] = 1;
                printf("%d ", i);
                break;
            }
        }
        if (i == n)
            pop();
    }
}

int main() {
    int n, i, j, startVertex;
    printf("Enter the number of vertices: ");
    scanf("%d", &n);

    printf("Enter the adjacency matrix:\n");
    for (i = 0; i < n; i++)
        for (j = 0; j < n; j++)
            scanf("%d", &adj[i][j]);

    printf("Enter the starting vertex for BFS: ");
    scanf("%d", &startVertex);
    printf("BFS Traversal: ");
    bfs(startVertex, n);
    printf("\n");

    printf("Enter the starting vertex for DFS: ");
    scanf("%d", &startVertex);
    printf("DFS Traversal: ");
    dfs(startVertex, n);
    printf("\n");

    return 0;
}
```

OUTPUT : --


PS D:\2MCA\DSA\Graph> ./bfs.exe
Enter the number of vertices: 2
Enter the adjacency matrix:
1
2
3
4
Enter the starting vertex for BFS: 1
BFS Traversal: 1
Enter the starting vertex for DFS: 2
DFS Traversal: 2
PS D:\2MCA\DSA\Graph>