**Name**: Darshan Deepak Bhokare

**Branch**: AI&DS **Div:** A **Batch:** 2

**Subject**: Advanced Data Structures

**Roll No**. 26 **PRN**. 12210089

**Lab 5**

**Graph Traversal-**

**1. BFS**

**2. DFS**

**3. Exit**

**Code:**

#include <stdio.h>

#include <stdlib.h>

#define MAX\_VERTICES 100

typedef struct Node

{

    int vertex;

    struct Node \*next;

} Node;

typedef struct AdjList

{

    Node \*head;

} AdjList;

typedef struct Graph

{

    int numVertices;

    AdjList \*array[MAX\_VERTICES];

} Graph;

Graph \*createGraph(int numVertices)

{

    Graph \*graph = (Graph \*)malloc(sizeof(Graph));

    graph->numVertices = numVertices;

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

    {

        graph->array[i] = (AdjList \*)malloc(sizeof(AdjList));

        graph->array[i]->head = NULL;

    }

    return graph;

}

void addEdge(Graph \*graph, int src, int dest)

{

    Node \*newNode = (Node \*)malloc(sizeof(Node));

    newNode->vertex = dest;

    newNode->next = graph->array[src]->head;

    graph->array[src]->head = newNode;

    newNode = (Node \*)malloc(sizeof(Node));

    newNode->vertex = src;

    newNode->next = graph->array[dest]->head;

    graph->array[dest]->head = newNode;

}

void BFS(Graph \*graph, int startVertex)

{

    int visited[MAX\_VERTICES] = {0};

    int queue[MAX\_VERTICES];

    int front = -1, rear = -1;

    visited[startVertex] = 1;

    queue[++rear] = startVertex;

    while (front != rear)

    {

        int currentVertex = queue[++front];

        printf("%d ", currentVertex);

        Node \*temp = graph->array[currentVertex]->head;

        while (temp)

        {

            int adjVertex = temp->vertex;

            if (!visited[adjVertex])

            {

                visited[adjVertex] = 1;

                queue[++rear] = adjVertex;

            }

            temp = temp->next;

        }

    }

}

void DFSUtil(Graph \*graph, int vertex, int visited[])

{

    visited[vertex] = 1;

    printf("%d ", vertex);

    Node \*temp = graph->array[vertex]->head;

    while (temp)

    {

        int adjVertex = temp->vertex;

        if (!visited[adjVertex])

        {

            DFSUtil(graph, adjVertex, visited);

        }

        temp = temp->next;

    }

}

void DFS(Graph \*graph, int startVertex)

{

    int visited[MAX\_VERTICES] = {0};

    DFSUtil(graph, startVertex, visited);

}

int main()

{

    int choice, vertices, edges, src, dest, startVertex;

    printf("Enter the number of vertices: ");

    scanf("%d", &vertices);

    Graph \*graph = createGraph(vertices);

    printf("Enter the number of edges: ");

    scanf("%d", &edges);

    for (int i = 0; i < edges; i++)

    {

        printf("Enter edge %d (S D): ", i + 1);

        scanf("%d %d", &src, &dest);

        addEdge(graph, src, dest);

    }

    while (1)

    {

        printf("\nMenu:\n");

        printf("1. BFS\n");

        printf("2. DFS\n");

        printf("3. Exit\n");

        printf("Enter choice: ");

        scanf("%d", &choice);

        switch (choice)

        {

        case 1:

            printf("Enter the start vertex (BFS): ");

            scanf("%d", &startVertex);

            printf("BFS Traversal: ");

            BFS(graph, startVertex);

            printf("\n");

            break;

        case 2:

            printf("Enter the start vertex (DFS): ");

            scanf("%d", &startVertex);

            printf("DFS Traversal: ");

            DFS(graph, startVertex);

            printf("\n");

            break;

        case 3:

            printf("Exit\n");

            exit(0);

        default:

            printf("Invalid choice.\n");

        }

    }

    return 0;

}

**Output:**

  
