**Slip 11**

Q.1) Write a programs to implement DFS (Depth First Search) and determine the time complexity for the same.

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

// Structure for a node in the adjacency list

struct Node {

    int data;

    struct Node\* next;

};

// Structure for the adjacency list

struct List {

    struct Node\* head;

};

// Structure for the graph

struct Graph {

    int vertices;

    struct List\* array;

};

// Function to create a new node

struct Node\* createNode(int data) {

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

    newNode->data = data;

    newNode->next = NULL;

    return newNode;

}

// Function to create a graph with a given number of vertices

struct Graph\* createGraph(int vertices) {

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

    graph->vertices = vertices;

    graph->array = (struct List\*)malloc(vertices \* sizeof(struct List));

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

        graph->array[i].head = NULL;

    }

    return graph;

}

// Function to add an edge to the graph

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

    struct Node\* newNode = createNode(dest);

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

    graph->array[src].head = newNode;

    // Uncomment the following code to make the graph undirected

    /\*

    newNode = createNode(src);

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

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

    \*/

}

// Function to perform Depth First Search (DFS) from a given vertex

void DFS(struct Graph\* graph, int vertex, bool visited[]) {

    visited[vertex] = true;

    printf("%d ", vertex);

    struct Node\* currentNode = graph->array[vertex].head;

    while (currentNode) {

        int adjacentVertex = currentNode->data;

        if (!visited[adjacentVertex]) {

            DFS(graph, adjacentVertex, visited);

        }

        currentNode = currentNode->next;

    }

}

// Function to perform DFS traversal from a given vertex in a specified order

void DFSTraversal(struct Graph\* graph, int\* order, int orderSize) {

    bool\* visited = (bool\*)malloc(graph->vertices \* sizeof(bool));

    for (int i = 0; i < graph->vertices; i++) {

        visited[i] = false;

    }

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

        if (!visited[order[i]]) {

            DFS(graph, order[i], visited);

        }

    }

    free(visited);

}

int main() {

    int vertices = 4;

    struct Graph\* graph = createGraph(vertices);

    addEdge(graph, 2, 0);

    addEdge(graph, 0, 2);

    addEdge(graph, 1, 2);

    addEdge(graph, 0, 1);

    addEdge(graph, 3, 3);

    addEdge(graph, 1, 3);

    int order[] = {2, 0, 1, 3};

    int orderSize = sizeof(order) / sizeof(order[0]);

    printf("Following is Depth First Traversal (starting from vertex 2):\n");

    DFSTraversal(graph, order, orderSize);

    return 0;

}