DFS

#include <stdio.h>

#include <stdlib.h>

#define MAX\_VERTICES 100

struct Node {

int vertex;

struct Node\* next;

};

struct Graph {

int numVertices;

struct Node\* adjList[MAX\_VERTICES];

};

struct Graph\* createGraph(int numVertices) {

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

graph->numVertices = numVertices;

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

graph->adjList[i] = NULL;

}

return graph;

}

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

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

newNode->vertex = dest;

newNode->next = graph->adjList[src];

graph->adjList[src] = newNode;

}

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

visited[vertex] = 1;

printf("%d ", vertex);

struct Node\* temp = graph->adjList[vertex];

while (temp != NULL) {

int adjVertex = temp->vertex;

if (!visited[adjVertex]) {

DFS(graph, adjVertex, visited);

}

temp = temp->next;

}

}

int main() {

int numVertices = 4;

struct Graph\* graph = createGraph(numVertices);

addEdge(graph, 0, 1);

addEdge(graph, 0, 2);

addEdge(graph, 1, 2);

addEdge(graph, 2, 0);

addEdge(graph, 2, 3);

addEdge(graph, 3, 3);

int visited[MAX\_VERTICES] = {0};

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

DFS(graph, 2, visited);

return 0;

}