#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

struct AdjListNode {

int dest;

struct AdjListNode\* next;

};

struct AdjList {

struct AdjListNode\* head;

};

struct Graph {

int V;

struct AdjList\* array;

};

struct AdjListNode\* newAdjListNode(int dest) {

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

newNode->dest = dest;

newNode->next = NULL;

return newNode;

}

struct Graph\* createGraph(int V) {

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

graph->V = V;

graph->array = (struct AdjList\*)malloc(V \* sizeof(struct AdjList));

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

graph->array[i].head = NULL;

}

return graph;

}

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

struct AdjListNode\* newNode = newAdjListNode(dest);

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

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

newNode = newAdjListNode(src);

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

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

}

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

visited[vertex] = true;

printf("%d ", vertex);

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

while (currentNode != NULL) {

int adjVertex = currentNode->dest;

if (!visited[adjVertex]) {

DFSUtil(graph, adjVertex, visited);

}

currentNode = currentNode->next; } }

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

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

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

visited[i] = false;

}

printf("Depth-First Traversal starting from vertex %d: ", src);

DFSUtil(graph, src, visited);

printf("\n");

free(visited);

}

int main() {

int V = 6;

struct Graph\* graph = createGraph(V);

addEdge(graph, 0, 1);

addEdge(graph, 0, 3);

addEdge(graph, 1, 3);

addEdge(graph, 1, 4);

addEdge(graph, 2, 4);

addEdge(graph, 2, 5);

int sourceVertex = 0;

DFS(graph, sourceVertex);

return 0; }