```
#include <stdio.h>

#include <stdlib.h>


struct node {

int vertex;

struct node* next;

};

struct node* createNode(int v);

struct Graph {

int numVertices;

int* visited;

// We need int** to store a two dimensional array.

// Similary, we need struct node** to store an array of Linked lists

struct node** adjLists; };


// DFS algo

void DFS(struct Graph* graph, int vertex) {

struct node* adjList = graph->adjLists[vertex];

struct node* temp = adjList;

graph->visited[vertex] = 1;

printf("Visited %d \n", vertex);

while (temp != NULL) {

int connectedVertex = temp->vertex;


if (graph->visited[connectedVertex] == 0) {


DFS(graph, connectedVertex);

}

temp = temp->next;

}

}
```

```
// Create a node
struct node* createNode(int v) {
struct node* newNode = malloc(sizeof(struct node)); newNode->vertex = v;
newNode->next = NULL;
return newNode; }

// Create graph
struct Graph* createGraph(int vertices) { struct Graph* graph = malloc(sizeof(struct Graph));
graph->numVertices = vertices;
graph->adjLists = malloc(vertices * sizeof(struct node*));

graph->visited = malloc(vertices * sizeof(int));
int i;
for (i = 0; i < vertices; i++) {
graph->adjLists[i] = NULL; graph->visited[i] = 0;
}
return graph; }

// Add edge
void addEdge(struct Graph* graph, int src, int dest) { // Add edge from src to dest
struct node* newNode = createNode(dest);
newNode->next = graph->adjLists[src]; graph->adjLists[src] = newNode;

// Add edge from dest to src
newNode = createNode(src);
newNode->next = graph->adjLists[dest];
graph->adjLists[dest] = newNode;
}
// Print the graph
void printGraph(struct Graph* graph) {
int v;
for (v = 0; v < graph->numVertices; v++) {
struct node* temp = graph->adjLists[v];
printf("\n Adjacency list of vertex %d\n ", v); while (temp) {
printf("%d -> ", temp->vertex);
temp = temp->next;
}
printf("\n");
}
}
int main() {
struct Graph* graph = createGraph(4);
addEdge(graph, 0, 1); addEdge(graph, 0, 2);
addEdge(graph, 1, 2);
addEdge(graph, 2, 3);

printGraph(graph);

DFS(graph, 2);
return 0;
}
```

Output:

```
Adjacency list of vertex 0 2 -> 1 ->
 Adjacency list of vertex 1 2 -> 0 ->
 Adjacency list of vertex 2 3 -> 1 -> 0 ->
Adjacency list of vertex 3 2 ->
Visited 2 Visited 3 Visited 1 Visited 0
```