Experimment-21 : Breadth First Search

21.Write a C Program To Graph Traversal Using Breadth First Search.

Code :

#include <stdio.h>

#include <stdlib.h>

#define MAX\_NODES 100

// Queue structure for BFS

typedef struct {

int items[MAX\_NODES];

int front;

int rear;

} Queue;

// Initialize queue

void initQueue(Queue \*q) {

q->front = -1;

q->rear = -1;

}

// Check if queue is empty

int isEmpty(Queue \*q) {

return q->rear == -1;

}

// Check if queue is full

int isFull(Queue \*q) {

return q->rear == MAX\_NODES - 1;

}

// Add element to queue

void enqueue(Queue \*q, int value) {

if (isFull(q)) {

printf("Queue is full!\n");

return;

}

if (isEmpty(q))

q->front = 0;

q->rear++;

q->items[q->rear] = value;

}

// Remove element from queue

int dequeue(Queue \*q) {

if (isEmpty(q)) {

printf("Queue is empty!\n");

return -1;

}

int item = q->items[q->front];

q->front++;

if (q->front > q->rear) {

q->front = q->rear = -1;

}

return item;

}

// Graph structure

typedef struct {

int vertices;

int \*\*adjMatrix;

int \*visited;

} Graph;

// Create graph

Graph \*createGraph(int vertices) {

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

graph->vertices = vertices;

graph->adjMatrix = (int \*\*)malloc(vertices \* sizeof(int \*));

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

graph->adjMatrix[i] = (int \*)calloc(vertices, sizeof(int));

}

graph->visited = (int \*)calloc(vertices, sizeof(int));

return graph;

}

// Add edge to graph

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

graph->adjMatrix[src][dest] = 1;

// For undirected graph uncomment the line below

// graph->adjMatrix[dest][src] = 1;

}

// BFS traversal

void BFS(Graph \*graph, int startNode) {

Queue q;

initQueue(&q);

graph->visited[startNode] = 1;

enqueue(&q, startNode);

printf("Breadth First Search starting from node %d: ", startNode);

while (!isEmpty(&q)) {

int currentVertex = dequeue(&q);

printf("%d ", currentVertex);

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

if (graph->adjMatrix[currentVertex][i] == 1 && !graph->visited[i]) {

graph->visited[i] = 1;

enqueue(&q, i);

}

}

}

printf("\n");

}

int main() {

int vertices, edges, startNode;

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

scanf("%d", &vertices);

Graph \*graph = createGraph(vertices);

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

scanf("%d", &edges);

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

int src, dest;

printf("Enter edge %d (source destination): ", i + 1);

scanf("%d%d", &src, &dest);

addEdge(graph, src, dest);

}

printf("Enter the starting node for BFS: ");

scanf("%d", &startNode);

BFS(graph, startNode);

return 0;

}

Output :

