Usn:1BM21CS222

CODE:

#include <stdio.h>

#define MAX\_QUEUE\_SIZE 100

int graph[MAX\_QUEUE\_SIZE][MAX\_QUEUE\_SIZE];

int visited[MAX\_QUEUE\_SIZE];

int queue[MAX\_QUEUE\_SIZE];

int front = -1;

int rear = -1;

int numVertices;

void enqueue(int vertex) {

if (rear == MAX\_QUEUE\_SIZE - 1) {

printf("Queue is full. Overflow condition.\n");

} else {

if (front == -1) {

front = 0;

}

rear++;

queue[rear] = vertex;

}

}

int dequeue() {

int vertex;

if (front == -1 || front > rear) {

printf("Queue is empty. Underflow condition.\n");

return -1;

} else {

vertex = queue[front];

front++;

return vertex;

}

}

void bfs(int startVertex) {

int i, currentVertex;

enqueue(startVertex);

visited[startVertex] = 1;

printf("BFS Traversal: ");

while (front != -1 && front <= rear) {

currentVertex = dequeue();

printf("%d ", currentVertex);

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

if (graph[currentVertex][i] == 1 && visited[i] == 0) {

enqueue(i);

visited[i] = 1;

}

}

}

}

int main() {

int i, j, startVertex;

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

scanf("%d", &numVertices);

printf("Enter the adjacency matrix:\n");

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

for (j = 0; j < numVertices; j++) {

scanf("%d", &graph[i][j]);

}

}

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

visited[i] = 0;

}

printf("Enter the starting vertex: ");

scanf("%d", &startVertex);

bfs(startVertex);

return 0;

}

OUTPUT:

