//Graph(BFS)

#include <stdio.h>

#include <stdlib.h>

#define MAX 100

struct Queue {

int visited[MAX];

int front, rear;

};

struct Queue\* createQueue() {

struct Queue\* q = (struct Queue\*)malloc(sizeof(struct Queue));

q->front = -1;

q->rear = -1;

return q;

}

int isEmpty(struct Queue\* q) {

return q->front == -1;

}

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

if (q->rear == MAX - 1) {

printf("Queue is full\n");

} else {

if (q->front == -1) {

q->front = 0;

}

q->visited[++q->rear] = value;

}

}

int dequeue(struct Queue\* q) {

int item;

if (isEmpty(q)) {

printf("Queue is empty\n");

return -1;

} else {

item = q->visited[q->front];

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

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

} else {

q->front++;

}

return item;

}

}

void bfs(int graph[MAX][MAX], int startVertex, int n) {

int visited[MAX] = {0};

struct Queue\* q = createQueue();

visited[startVertex] = 1;

enqueue(q, startVertex);

printf("BFS Traversal: ");

while (!isEmpty(q)) {

int currentVertex = dequeue(q);

printf("%d ", currentVertex);

for (int i = 1; i <= n; i++) {

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

visited[i] = 1;

enqueue(q, i);

}

}

}

printf("\n");

}

int main() {

int n, startVertex;

int graph[MAX][MAX];

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

scanf("%d", &n);

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

for (int i = 1; i <= n; i++) {

for (int j = 1; j <= n; j++) {

printf("Vertex %d, %d : ",i,j);

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

}

}

printf("Enter the starting vertex: ");

scanf("%d", &startVertex);

bfs(graph, startVertex, n);

return 0;

}

Output:

