Program 9a

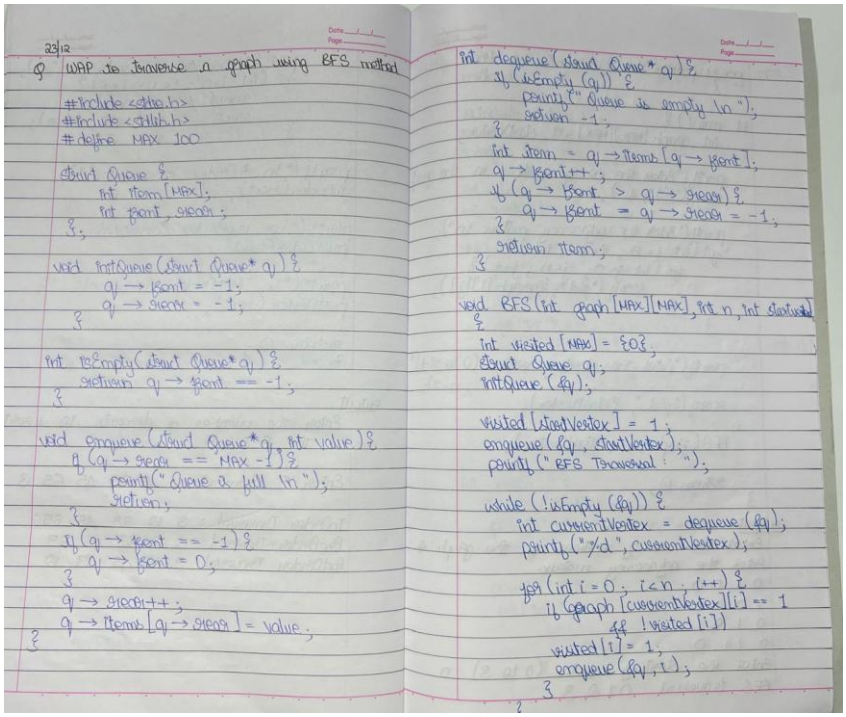
Write a program to traverse a graph using BFS method.

Code:

```
#include <stdio.h>
#include <stdlib.h>

#define MAX 100

struct Queue {
    int items[MAX];
    int front, rear;
};

void initQueue(struct Queue* q) {
    q->front = -1;
    q->rear = -1;
}

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");
        return;
    }

    if (q->front == -1) {
        q->front = 0;
    }
    q->rear++;
    q->items[q->rear] = value;
}

int dequeue(struct 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;
}
```

```
void BFS(int graph[MAX][MAX], int n, int startVertex) {
    int visited[MAX] = {0};
    struct Queue q;
    initQueue(&q);

    visited[startVertex] = 1;
    enqueue(&q, startVertex);

    printf("BFS Traversal: ");

    while (!isEmpty(&q)) {
        int currentVertex = dequeue(&q);
        printf("%d ", currentVertex);

        for (int i = 0; i < n; i++) {
            if (graph[currentVertex][i] == 1 && !visited[i]) {
                visited[i] = 1;
                enqueue(&q, i);
            }
        }
    }
    printf("\n");
}

int main() {
    int graph[MAX][MAX], n, startVertex;

    printf("Enter the number of vertices in the graph: ");
    scanf("%d", &n);

    printf("Enter the adjacency matrix of the graph:\n");
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            scanf("%d", &graph[i][j]);
        }
    }

    printf("Enter the starting vertex (0 to %d): ", n - 1);
    scanf("%d", &startVertex);

    BFS(graph, n, startVertex);

    return 0;
}
```

```
Enter the number of vertices in the graph: 4
Enter the adjacency matrix of the graph:
0 1 0 0
1 0 1 1
0 1 0 1
0 1 1 0
Enter the starting vertex (0 to 3): 0
BFS Traversal: 0 1 2 3
```



```
#include <stdio.h>
#include <stdlib.h>
#define MAX 100

struct Queue {
    int items[MAX];
    int front, rear;
};

void initQueue(struct Queue* q) {
    q->front = -1;
    q->rear = -1;
}

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");
        return;
    }
    if (q->front == -1) {
        q->front = 0;
    }
    q->rear++;
    q->items[q->rear] = value;
}

int dequeue(struct 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;
}

void BFS(int graph[MAX][MAX], int n, int startVertex) {
    int visited[MAX] = {0};
    struct Queue q;
    initQueue(&q);

    visited[startVertex] = 1;
    enqueue(&q, startVertex);
    printf("BFS Traversal: ");

    while (!isEmpty(&q)) {
        int currentVertex = dequeue(&q);
        printf("%d ", currentVertex);

        for (int i = 0; i < n; i++) {
            if (graph[currentVertex][i] == 1 &&
                !visited[i]) {
                visited[i] = 1;
                enqueue(&q, i);
            }
        }
    }
    printf("\n");
}

int main() {
    int graph[MAX][MAX], n, startVertex;

    printf("Enter the no of vertices in the graph: ");
    scanf("%d", &n);

    printf("Enter the adjacency matrix\n");
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            scanf("%d", &graph[i][j]);
        }
    }

    printf("Enter the starting vertex (0 to %d): ",
           n-1);
    scanf("%d", &startVertex);

    BFS(graph, n, startVertex);

    return 0;
}
```

output

```
Enter the no of vertices in the graph: 4
Enter the adjacency matrix
0 1 0 0
1 0 1 1
0 1 0 1
0 1 1 0
Enter the starting vertex (0 to 3): 0
BFS traversal: 0 1 2 3
```