22.Write a C program to Graph traversal using Breadth First Search

```
#include <stdio.h>
#define MAX 20
int queue[MAX];
int front = -1, rear = -1;
int visited[MAX];
int adj[MAX][MAX];
int n;
void enqueue(int v) {
   if (rear == MAX - 1)
      printf("Queue Overflow\n");
   else {
      if (front == -1)
         front = 0;
      rear++;
      queue[rear] = v;
   }
}
int dequeue() {
   int v;
   if (front == -1 || front > rear)
      return -1;
   v = queue[front];
   front++;
   return v;
}
void bfs(int start) {
   int i, current;
   for (i = 0; i < n; i++)
      visited[i] = 0;
```

```
    enqueue(start);
    visited[start] = 1;
    printf("BFS Traversal: ");
    while (front <= rear) {
        current = dequeue();
        printf("%d ", current);
        for (i = 0; i < n; i++) {
            if (adj[current][i] == 1 && visited[i] == 0) {
                enqueue(i);
                visited[i] = 1;
            }
        }
    }
    printf("\n");
}
int main() {
    int i, j, start;
    printf("Enter the number of vertices: ");
    scanf("%d", &n);
    printf("Enter the adjacency matrix:\n");
    for (i = 0; i < n; i++) {
        for (j = 0; j < n; j++) {
            scanf("%d", &adj[i][j]);
        }
    }
    printf("Enter the starting vertex (0 to %d): ", n - 1);
    scanf("%d", &start);
    bfs(start);
    return 0;
}
```

main.c

Share Run

```c
#include <stdio.h>
#define MAX 20
int queue[MAX];
int front = -1, rear = -1;
int visited[MAX];
int adj[MAX][MAX];
int n;

void enqueue(int v) {
    if (rear == MAX - 1)
        printf("Queue Overflow\n");
    else {
        if (front == -1)
            front = 0;
        rear++;
        queue[rear] = v;
    }
}

int dequeue() {
    int v;
    if (front == -1 || front > rear)
        return -1;
    v = queue[front];
    front++;
    return v;
}
```

Output

```
Enter the number of vertices: 5
Enter the adjacency matrix:
1 0 2 1 4
1 5 6 2 3
4 5 2 3 8
7 8 5 6 4
2 3 5 4 6
Enter the starting vertex (0 to 4): 3
BFS Traversal: 3


=== Code Execution Successful ===
```