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

#define MAX 100

int adj[MAX][MAX];
int visited[MAX];
int queue[MAX];
int front = -1, rear = -1;

void insert(int v) {
    if (rear == MAX - 1)
        return;
    if (front == -1)
        front = 0;
    queue[++rear] = v;
}

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

void bfs(int start, int n) {
    for (int i = 0; i < n; i++)
        visited[i] = 0;

    insert(start);
    visited[start] = 1;

    while (front <= rear) {
        int node = delete();
        printf("%d ", node);
        for (int i = 0; i < n; i++) {
            if (adj[node][i] == 1 && !visited[i]) {
                insert(i);
                visited[i] = 1;
            }
        }
    }
}

int main() {
    int n = 4;
```

```
    adj[0][1] = 1;
    adj[0][2] = 1;
    adj[1][2] = 1;
    adj[2][0] = 1;
    adj[2][3] = 1;
    adj[3][3] = 1;
    printf("BFS traversal from node 2 = \n");
    bfs(2, n);
    return 0;
}
```

Output:

```
sammj@SAM_LAPTOP MINGW64 ~/DS LAB
$  /usr/bin/env c:\\Users\\sammj\\.vsc
soft-MIEngine-In-avsd042x.u0q --stdout
x35rnmc5.lrl --dbgExe=C:\\msys64\\ucrt
BFS traversal from node 2 =
2 0 3 1
```

PAGE :
DATE : / /

9a. BFS

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

#define MAX 100

int adj[MAX][MAX];
int visited[MAX];
int queue[MAX];
int front = -1, rear = -1;

void visit(int v)
{
    if (rear == MAX-1)
        return;
    if (front == -1)
        front = 0;
    queue[++rear] = v;
}
int delete()
{
    if (front == -1 || front > rear)
    {
        return -1;
    }
    return queue[front++];
}
void bfs(int start, int n)
{
    for (int i=0; i<n; i++)
    {
        visited[i] = 0;
    }
```

```
enqueue
insert(start);
visited[start] = 1;

while(front <= rear)
{
    int node = dequeue(); delete
    printf("%d", node);
    for(int i=0; i<n; i++)
    {
        if(adj[node][i]==1 && !visited[i])
        {
            insert(i);
            visited[i]=1;
        }
    }
}
int main()
{
    int n=4;
    adj[0][1]=1;
    adj[0][2]=1;
    adj[1][2]=1;
    adj[2][0]=1;
    adj[2][3]=1;
    adj[3][3]=1;
    printf("BFS traversal from node 2 = \n");
    bfs(2,n);
    return 0;
}
```

Output:

BFS traversal from node 2 =

2 0 3 1