```
#include <stdio.h>

int Adjmat[4][4] = {
    {0, 0, 1, 0},  // Node 1 connected to Node 3
    {1, 0, 0, 0},  // Node 2 connected to Node 1
    {0, 0, 0, 1},  // Node 3 connected to Node 4
    {0, 1, 0, 0}   // Node 4 connected to Node 2
};
int visit[4] = {0, 0, 0, 0}; // Array to keep track of visited nodes
int queue[4];
int front = -1;
int rear = -1;
int isEmpty() {
    return front == -1;
}
```

```
int isFull() {
    return rear == 4 - 1;
}
void enqueue(int element) {
    if (isFull()) {
        printf("Queue is full\n");
        return;
    }
    if (isEmpty()) {
        front = rear = 0;
    } else {
        rear++;
    }
    queue[rear] = element;
    // printf("%d enqueued to queue\n", element); // Remove this print to clean the output
}
int dequeue() {
    if (isEmpty()) {
        printf("Queue is empty\n");
        return -1;
    }
    int element = queue[front];
    if (front == rear) {
        front = rear = -1;
    } else {
        front++;
    }
    return element;
}
void bfs(int startNode) {
    enqueue(startNode);
    visit[startNode] = 1;
    printf("BFS Traversal: ");
    while (!isEmpty()) {
        int currentNode = dequeue();
        printf("%d ", currentNode + 1); // Printing nodes as 1-based index
        for (int i = 0; i < 4; i++) {
            if (Adjmat[currentNode][i] == 1 && visit[i] == 0) {
                enqueue(i);
                visit[i] = 1;
            }
        }
    }
    printf("\n");
}
int main() {
    bfs(0); // Start BFS from node 1 (0-based index)
    return 0;
}
```