```
#include <iostream>
#include <vector>
#include <queue>
using namespace std;
void BFS(int start, vector<vector<int>>& adj, vector<bool>& visited) {
    queue<int> q;
    visited[start] = true;
    q.push(start);
    cout << "BFS Traversal: ";
    while (!q.empty()) {
        int node = q.front();
        q.pop();
        cout << node << " ";
        // Visit all adjacent nodes
        for (int neighbor : adj[node]) {
            if (!visited[neighbor]) {
                visited[neighbor] = true;
                q.push(neighbor);
            }
        }
    }
    cout << endl;
}
int main() {
    int nodes, edges;
    cout << "Enter number of nodes and edges: ";
    cin >> nodes >> edges;
    vector<vector<int>> adj(nodes + 1); // adjacency list
    vector<bool> visited(nodes + 1, false);
    cout << "Enter edges (u v):" << endl;
    for (int i = 0; i < edges; ++i) {
        int u, v;
        cin >> u >> v;
        adj[u].push_back(v);
        adj[v].push_back(u); // For undirected graph
    }
    int startNode;
    cout << "Enter starting node: ";
    cin >> startNode;
    BFS(startNode, adj, visited);
    return 0;
}
```

OUTPUT:

Sample Input:

```
Enter number of nodes and edges: 5 4
Enter edges (u v):
1 2
1 3
2 4
3 5
Enter starting node: 1
BFS Traversal: 1 2 3 4 5
```