```
#include <iostream>
#include <vector>
using namespace std;
void DFS(int node, vector<vector<int>>& adj, vector<bool>& visited) {
    visited[node] = true;
    cout << node << " ";
    for (int neighbor : adj[node]) {
        if (!visited[neighbor]) {
            DFS(neighbor, adj, visited);
        }
    }
}
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;
    cout << "DFS Traversal: ";
    DFS(startNode, adj, visited);
    cout << endl;
    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

Output:

DFS Traversal: 1 2 4 3 5