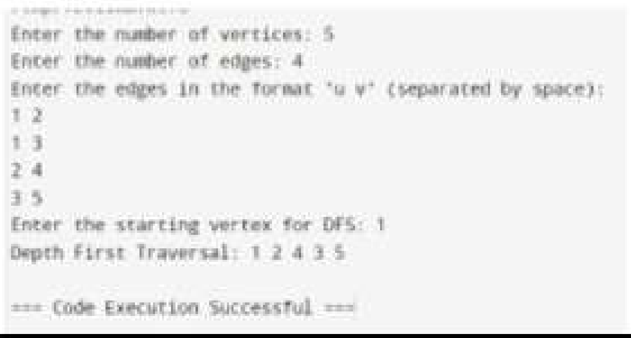
```cpp
#include <iostream>

#include <vector>

#include <unordered_set>

#include <unordered_map> // Added for
unordered_map

using namespace std;

class Graph {

   unordered_map<int, vector<int>>
adjList; // Added unordered_map
declaration

public:

   void addEdge(int u, int v) {

      adjList[u].push_back(v);

      adjList[v].push_back(u); // Assuming
it's an undirected graph

   }

   void dfsUtil(int v, unordered_set<int>&
visited) {

      visited.insert(v);

      cout << v << " ";


      for (int neighbor : adjList[v]) {

         if (visited.find(neighbor) ==
visited.end()) {

            dfsUtil(neighbor, visited);

         }

      }

   }

   void dfs(int start) {

      unordered_set<int> visited;

      dfsUtil(start, visited);

   }

};

int main() {

   Graph g;

   int V, E;

   cout << "Enter the number of vertices:
";

   cin >> V;

   cout << "Enter the number of edges: ";

   cin >> E;

   cout << "Enter the edges in the format
'u v' (separated by space):" << endl;

   for (int i = 0; i < E; ++i) {

      int u, v;

      cin >> u >> v;

      g.addEdge(u, v);}

   int startVertex;

   cout << "Enter the starting vertex for
DFS: ";

   cin >> startVertex;

   cout << "Depth First Traversal: ";

   g.dfs(startVertex);


   return 0;

}
```



```
Enter the number of vertices: 5
Enter the number of edges: 4
Enter the edges in the format 'u v' (separated by space):
1 2
1 3
2 4
3 5
Enter the starting vertex for DFS: 1
Depth First Traversal: 1 2 4 3 5

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