```cpp
// C++ program to print DFS traversal from
// a given vertex in a given graph
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <cstddef>
#include <omp.h>
#include <bits/stdc++.h>
using namespace std;

// Graph class represents a directed graph
// using adjacency list representation
class Graph {
public:
        map<int, bool> visited;
        map<int, list<int> > adj;

        // function to add an edge to graph
        void addEdge(int v, int w);

        // DFS traversal of the vertices
        // reachable from v
        void DFS(int v);
};

void Graph::addEdge(int v, int w)
{
        adj[v].push_back(w); // Add w to v’s list.
}

void Graph::DFS(int v)
{
        // Mark the current node as visited and
        // print it
        visited[v] = true;
        cout << v << " ";

        // Recur for all the vertices adjacent
        // to this vertex
        list<int>::iterator i;
        for (i = adj[v].begin(); i != adj[v].end(); ++i)
                if (!visited[*i])
                        DFS(*i);
}

// Driver's code
int main()
{
        // Create a graph given in the above diagram
        Graph g;
        g.addEdge(0, 1);
        g.addEdge(0, 2);
        g.addEdge(1, 2);/---
        g.addEdge(2, 0);
        g.addEdge(2, 3);
        g.addEdge(3, 3);

        cout << "Following is Depth First Traversal"
                " (starting from vertex 2) \n";

        // Function call
        g.DFS(2);

        return 0;
}
```

```
// improved by Vishnudev C
```