

# HPC PRACTICAL NO – 1

## PARALLEL DFS

**Aim: Design and implement Parallel Breadth First Search and Depth First Search based on existing algorithms using OpenMP. Use a Tree or an undirected graph for BFS and DFS .**

```
#include <iostream>

#include <vector>

#include <stack>

#include <omp.h>

using namespace std;

const int MAX = 100000;

vector<int> graph[MAX];

bool visited[MAX];

void dfs(int node) {
    stack<int> s;
    s.push(node);

    while (!s.empty()) {
        int curr_node = s.top();
        s.pop();

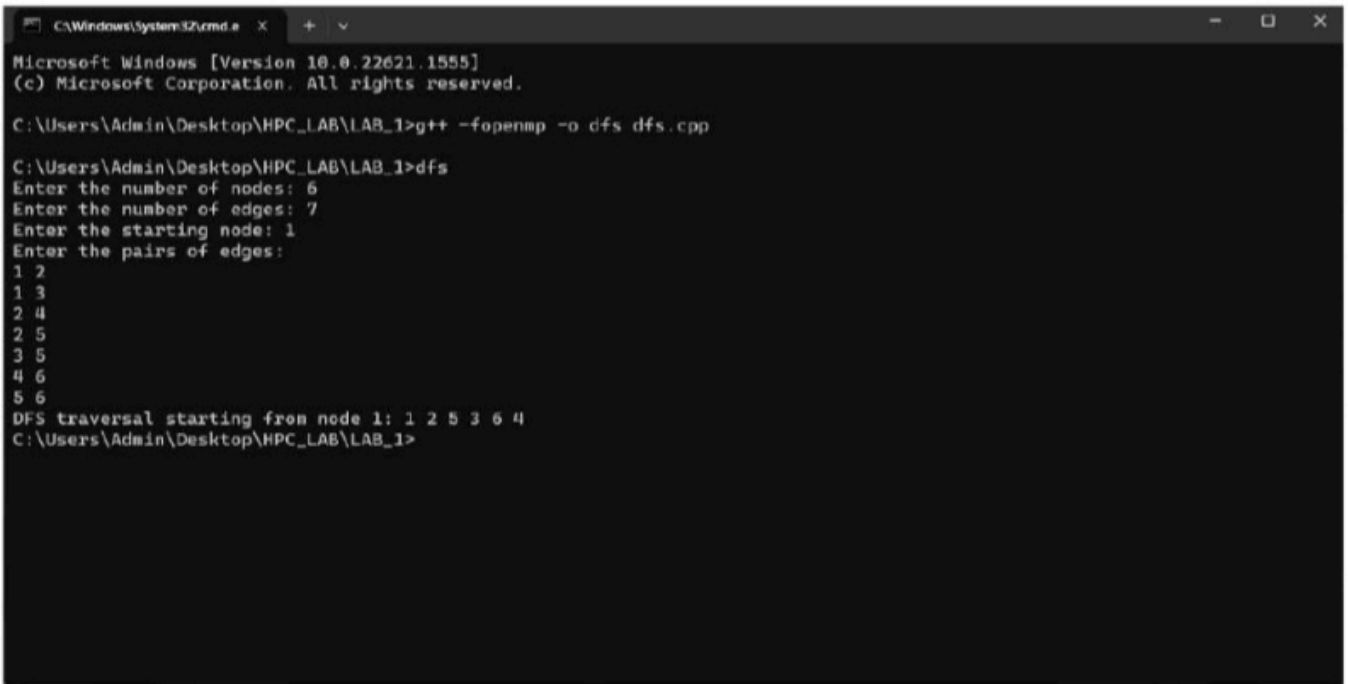
        if (!visited[curr_node]) {
            visited[curr_node] = true;

            if (visited[curr_node]) {
                cout << curr_node << " ";
            }
        }
    }
}
```



```
cout << "DFS traversal starting from node " << start_node << ": ";  
dfs(start_node);  
  
return 0;  
}
```

## Output:



```
C:\Windows\System32\cmd.exe x + v  
Microsoft Windows [Version 10.0.22621.1555]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Users\Admin\Desktop\HPC_LAB\LAB_1>g++ -fopenmp -o dfs dfs.cpp  
  
C:\Users\Admin\Desktop\HPC_LAB\LAB_1>dfs  
Enter the number of nodes: 6  
Enter the number of edges: 7  
Enter the starting node: 1  
Enter the pairs of edges:  
1 2  
1 3  
2 4  
2 5  
3 5  
4 6  
5 6  
DFS traversal starting from node 1: 1 2 5 3 6 4  
C:\Users\Admin\Desktop\HPC_LAB\LAB_1>
```