**Assignment 1**

**Parallel DFS**

#include<bits/stdc++.h>

#include<omp.h>

using namespace std;

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)

{

#pragma omp parallel

// Mark the current node as visited and print it

visited[v] = true;

cout<<v<<" ";

list<int>::iterator i; // Recur for all the vertices adjacent to this vertex

for(i=adj[v].begin();i!=adj[v].end();++i)

{

if(!visited[\*i])

DFS(\*i);

}

}

int main()

{

omp\_set\_num\_threads(4);

int z;

Graph g;

g.addEdge(0,1);

g.addEdge(0,2);

g.addEdge(1,3);

g.addEdge(2,3);

g.addEdge(3,4);

g.addEdge(3,5);

g.addEdge(2,6);

cout<<"Enter the vertex to start the DFS traversal with: "<<endl;

cin>>z;

cout<<"\nDepth First Traversal: \n";

g.DFS(z);

cout<<endl;

return 0;

}

**Output –**

Enter the vertex to start the DFS traversal with:

0

Depth First Traversal:

0 1 3 4 5 2 6