**Parallel BFS**

#include <iostream>

#include <vector>

#include <queue>

#include <omp.h>

using namespace std;

vector<int> graph[100000];

bool visited[100000];

void parallelBFS(int start) {

queue<int> q;

q.push(start);

visited[start] = true;

cout << "Parallel BFS Traversal: ";

while (!q.empty()) {

int node = q.front();

q.pop();

cout << node << " ";

#pragma omp parallel for

for (int i = 0; i < graph[node].size(); i++) {

int neighbor = graph[node][i];

if (!visited[neighbor]) {

#pragma omp critical

visited[neighbor] = true, q.push(neighbor);

}

}

}

}

int main() {

int n, m, start;

cout << "Enter No of Nodes, Edges, and Start Node: ";

cin >> n >> m >> start;

cout << "Enter Pairs of Edges:\n";

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

int u, v; cin >> u >> v;

graph[u].push\_back(v); graph[v].push\_back(u);

}

fill(visited, visited + n, false);

parallelBFS(start);

return 0;

}