#include <iostream>

#include <vector>

#include <queue>

#include <omp.h>

using namespace std;

class ParallelBFS {

private:

vector<vector<int>> adjMatrix;

vector<int> visited;

int n;

public:

void input() {

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

cin >> n;

adjMatrix.resize(n, vector<int>(n, 0));

visited.resize(n, 0);

cout << "Enter the adjacency matrix:\n";

for (int i = 0; i < n; i++)

for (int j = 0; j < n; j++)

cin >> adjMatrix[i][j];

}

void display() {

cout << "Adjacency Matrix:\n";

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

for (int j = 0; j < n; j++)

cout << adjMatrix[i][j] << " ";

cout << endl;

}

}

void bfs\_sequential(int start) {

queue<int> q;

fill(visited.begin(), visited.end(), 0);

q.push(start);

visited[start] = 1;

cout << "Sequential BFS Order: ";

while (!q.empty()) {

int current = q.front();

q.pop();

cout << current << " ";

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

if (adjMatrix[current][i] && !visited[i]) {

q.push(i);

visited[i] = 1;

}

}

}

cout << endl;

}

void bfs\_parallel(int start) {

queue<int> q;

fill(visited.begin(), visited.end(), 0);

#pragma omp parallel

{

#pragma omp single

{

q.push(start);

visited[start] = 1;

cout << "Parallel BFS Order: ";

while (!q.empty()) {

int current = q.front();

q.pop();

cout << current << " ";

#pragma omp task firstprivate(current)

{

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

if (adjMatrix[current][i] && !visited[i]) {

#pragma omp critical

{

if (!visited[i]) {

q.push(i);

visited[i] = 1;

}

}

}

}

}

}

cout << endl;

}

}

}

};

int main() {

ParallelBFS bfs;

bfs.input();

bfs.display();

int startVertex;

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

cin >> startVertex;

double start, end;

start = omp\_get\_wtime();

bfs.bfs\_sequential(startVertex);

end = omp\_get\_wtime();

cout << "Time taken by Sequential BFS: " << end - start << " seconds\n";

start = omp\_get\_wtime();

bfs.bfs\_parallel(startVertex);

end = omp\_get\_wtime();

cout << "Time taken by Parallel BFS: " << end - start << " seconds\n";

return 0;

}