数据结构作业12

Code:

#include <iostream>  
#include <vector>  
#include <algorithm>  
#include <stack>  
#include <queue>  
using namespace std;  
*//图的dfs*const int N = 1e5 + 10;  
vector<vector<int>> g;  
vector<bool> visited;  
*//图的dfs：传入正在访问的节点u*void dfs(int u) {  
 visited[u] = true;  
 cout << u << " ";  
 for (auto i : g[u]) {  
 int v = g[u][i];  
 if (!visited[v]) {  
 dfs(v);  
 }  
 }  
} *//递归算法，时间复杂度为O(n + m)*void dfs2(int u) {  
 stack<int> s; *//利用了std::stack* s.push(u); *//将起始点入栈* visited[u] = true; *//标记访问* cout << u << " ";  
 while (!s.empty()) {  
 int u = s.top(); *//取栈顶元素* s.pop();  
 for (auto i : g[u]) {  
 int v = g[u][i];  
 if (!visited[v]) {  
 s.push(v);  
 visited[v] = true;  
 cout << v << " ";  
 }  
 }  
 }  
} *//非递归算法  
  
//图的bfs*void bfs(int u) {  
 queue<int> q; *//利用了std::queue* q.push(u); *//将起始点入队* visited[u] = true; *//标记已经访问过* while (!q.empty()) {  
 int u = q.front();  
 q.pop();  
 cout << u << " ";  
 for (auto i : g[u]) {  
 int v = g[u][i];  
 if (!visited[v]) {  
 q.push(v);  
 visited[v] = true;  
 }  
 }  
 }  
}  
int main(){  
 ios::sync\_with\_stdio(0);  
 cin.tie(0);  
 cout.tie(0);  
 int n, m;  
 cin >> n >> m;  
 g.resize(n + 1);  
 visited.resize(n + 1);  
 for (int i = 0; i < m; i++) {  
 int u, v;  
 cin >> u >> v;  
 g[u].emplace\_back(v);  
 g[v].emplace\_back(u);  
 }  
 for (int i = 1; i <= n; i++) {  
 if (!visited[i]) {  
 dfs(i);  
 }  
 }  
 return 0;  
}