#### 无向图的深度和广度优先遍历 - C++

需要解决的问题：

一个无向图，怎么从深度和广度来遍历这个图，也就是怎么个走法

需要了解和学习的点：

图的邻接矩阵存储法（就是一个二维数组）

回溯 (这里要理解循环能给递归产生回溯的效果）

图的生成树

代码

深度优先遍历

#include <iostream>

#include <climits>

using namespace std;

int book[101] = {0}, sum, n, m, e[101][101] = {0};

void dfs(int cur) {

cout << cur << " ";

sum++;

if (sum == n)return;

for (int i = 1; i <= n; i++) { //循环达到回溯的效果

if (book[i] == 0 && e[cur][i] == 1) {

book[i] = 1;

dfs(i);

}

}

return;

}

int main(int argc, const char \*argv[]) {

cin >> n >> m; //n个节点，m条边

for (int i = 1; i <= n; i++) //初始化

for (int j = 1; j <= n; j++) {

if (i == j)

e[i][j] = 0;

else

e[i][j] = INT\_MAX;

}

int a, b;

for (int k = 1; k <= m; k++) { //矩阵表示

cin >> a >> b;

e[a][b] = 1;

e[b][a] = 1;

}

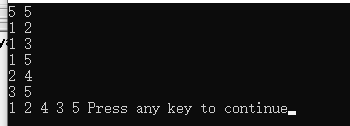
book[1] = 1;

dfs(1);

return 0;

}

运行结果：



广度优先遍历：

#include <iostream>

#include <climits>

using namespace std;

int main() {

int n, m, a, b, cur, book[101] = {0}, e[101][101] = {0};

int que[101], head = 1, tail = 1;

cin >> n >> m; //n个节点，m条边

for (int i = 1; i <= n; i++) //初始化

for (int j = 1; j <= n; j++) {

if (i == j)

e[i][j] = 0;

else

e[i][j] = INT\_MAX;

}

for (int i = 1; i <= m; i++) { //图的矩阵表示

cin >> a >> b;

e[a][b] = 1;

e[b][a] = 1;

}

que[1] = 1;

tail++;

book[1] = 1; //走过的标记为1

while (head < tail) {

cur = que[head];

for (int i = 1; i <= n; i++) { //当前顶点往下的所有可能都存到队列中去

if (book[i] == 0 && e[cur][i] == 1) {

que[tail] = i;

tail++;

}

if (tail > n)

break;

}

head++; //表示当前点遍历结束，下个点

}

for (int i = 1; i <= n; i++) {

cout << que[i] << " ";

}

return 0;

}

运行结果：

