代码：（截图再下代码下面）

#include <iostream>

#include <queue>

using namespace std;

typedef enum

{

UDG , UDN , DG , DN

}MapType;

class Linjie

{

private:

int \_Node\_num; //顶点数

int \_Side\_num; //边数

MapType \_Kind; //图类型

int\*\* \_Map; //图

int\* \_v; //顶点的顺序集

int \*visited;

void Find\_DFS(int number);

public:

Linjie(int node\_num = 0 , int side\_num = 0 , MapType k = (MapType)0); //默认为无向无权图

void Create(); //顶点数、边数

void Gain\_node(); //获取图的信息

void Print\_map(); //打印邻接矩阵

void DFS();

void BFS();

};

Linjie::Linjie(int node\_num , int side\_num, MapType k)

{

this->\_Node\_num = node\_num;

this->\_Side\_num = side\_num;

this->\_Kind = k;

}

void Linjie::Create()

{

this->\_Map = new int\*[this->\_Node\_num];

for(int i = 0 ; i < this->\_Node\_num ; ++i)

\_Map[i] = new int[this->\_Node\_num];

this->\_v = new int[this->\_Node\_num];

for(int i = 0 ; i < this->\_Node\_num ; ++i)

{

this->\_v[i] = i+1;

}

visited = new int[this->\_Node\_num];

for(int i = 0 ; i < this->\_Node\_num ; ++i)

{

visited[i] = 0;

}

}

void Linjie::Gain\_node()

{

//初始矩阵元素

for(int i = 0 ; i < this->\_Node\_num ; ++i)

{

for(int j = 0 ; j < this->\_Node\_num ; ++j)

{

\_Map[i][j] = 0;

}

}

int i = 0 , j = 0 , w = 0;

while(this->\_Side\_num--)

{

if(this->\_Kind == UDG || this->\_Kind == DG) //无向无权、有向无权

{

cout << "输入两个节点代号表示它们之间存在边,空格隔开" << endl;

cin >> i >> j;

\_Map[i-1][j-1] = 1;

if(this->\_Kind == UDG)

\_Map[j-1][i-1] = 1;

}

else if(this->\_Kind == UDN || this->\_Kind == DN)

{

cout << "输入两个节点代号表示它们之间存在边,输入边的权重,空格隔开" << endl;

cin >> i >> j >> w;

\_Map[i-1][j-1] = w;

if(this->\_Kind == UDN)

\_Map[j-1][i-1] = w;

}

}

}

void Linjie::Print\_map()

{

cout << "\t";

for(int i = 0 ; i < this->\_Node\_num ; ++i)

cout << "V" << (i+1) << "\t";

cout << endl;

for(int i = 0 ; i < this->\_Node\_num ; ++i)

{

cout << "V" << (i+1) << "\t";

for(int j = 0 ; j < this->\_Node\_num ; ++j)

{

cout << \_Map[i][j] << "\t";

}

cout << endl;

}

}

void Linjie::Find\_DFS(int number)

{

this->visited[number] = 1;

cout << this->\_v[number] << "\t";

for(int j = 0 ; j < this->\_Node\_num ; ++j)

{

if(this->\_Map[number][j] != 0 && this->visited[j] == 0)

{

Find\_DFS(j);

}

}

}

void Linjie::DFS()

{

for(int i = 0 ; i < this->\_Node\_num ; ++i)

{

visited[i] = 0;

}

for(int i = 0 ; i < this->\_Node\_num ; ++i)

{

if(this->visited[i] == 0)

{

Find\_DFS(i);

}

}

}

void Linjie::BFS()

{

queue<int> Que;

for(int i = 0 ; i < this->\_Node\_num ; ++i)

{

visited[i] = 0;

}

for(int i = 0 ; i < this->\_Node\_num ; ++i)

{

if(visited[i] == 0)

{

visited[i] = 1;

cout << this->\_v[i] << "\t";

Que.push(i);

while(Que.empty() == true)

{

int index = 0;

index = Que.front();

for(int i = 0 ; i < this->\_Node\_num ; ++i)

{

if(this->\_Map[index][i] != 0 && visited[i] == 0)

{

visited[i] = 1;

cout << \_Map[index][i] << "\t";

Que.push(i);

}

}

}

}

}

}

int main()

{

Linjie a(6 , 6 , (MapType)0);

a.Create();

a.Gain\_node();

a.Print\_map();

cout << "深度优先遍历" << endl;

a.DFS();

cout << endl;

cout << "广度优先遍历" << endl;

a.BFS();

}截图：

