 实 验 报 告

课程名称： 数据结构（C++版）

题 目： 邻接矩阵的实现

专业班级： 18空间

姓 名： 邓珊珊

学 号： 1812001107

指导老师： 袁莹

实验时间： 2019.12.13

第一次实验报告

班级：18空间 姓名：邓珊珊 学号：1812001107

1. 实验目的及原理

（选择公式的原理，及文字说明，所选的公式需要输入什么，计算出什么，如何计算，目的：通过实际问题学会用函数编程实现）

1. 实验目的
2. 掌握图的逻辑结构；
3. 掌握图的邻接矩阵存储结构；
4. 验证图的邻接矩阵存储及其遍历操作的实现。
5. 实验内容
6. 建立无向图的邻接矩阵存储；
7. 对建立的无向图，进行深度优先遍历；
8. 对建立的无向图，进行广度优先遍历。
9. 实验提示

定义邻接矩阵存储的无向图类MGraph，包括题目要求的建立、深度优先遍历、广度优先遍历等基本操作。

1. 实验内容

（代码）

头文件Mgraph.h

#ifndef MGraph\_H //避免包含Mgraph.h头文件

#define MGraph\_H

const int MaxSize = 10; //图中最多顶点个数

//以下是类Mgraph的声明

template <class DataType>

class MGraph

{

public:

MGraph(DataType a[],int n, int e);

//构造函数，建立具有n个顶点e条边的图

~MGraph(){}

//析构函数为空

void DFSTraverse(int v);

//深度优先遍历图

void BFSTraverse(int v);

//广度优先遍历图

private:

DataType vertex[MaxSize];

//存放图中顶点的数组

int arc[MaxSize][MaxSize];

//存放图中边的数组

int vertexNum,arcNum;

//图的顶点数和边数

};

#endif /\* MGraph\_h \*/

Mgraph.cpp

#include <iostream>

//引入输入输出流

using namespace std;

#include "MGraph.h"

//引入类MGraph的声明

//以下是类MGraph的成员函数定义

template < class DataType >

MGraph<DataType>::MGraph(DataType a[],int n,int e)

{

int i, j, k;

vertexNum = n; arcNum = e;

for (i = 0; i < vertexNum; i++)

//存储图的顶点信息

vertex[i] = a[i];

for (i = 0; i < vertexNum; i++)

//初始化图的邻接矩阵

for (j = 0; j < vertexNum; j++)

arc[i][j] = 0;

for (k = 0; k < arcNum; k++)

//存储图的边信息

{

cout<<"请输入边的两个顶点的序号:";

cin>>i>>j;

arc[i][j] = 1; arc[j][i] = 1;

}

}

template < class DataType >

void MGraph<DataType>::DFSTraverse(int v)

//深度优先遍历图

{

cout<<vertex[v]; visited[v] = 1;

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

if (arc[v][j] == 1 && visited[j] == 0)

DFSTraverse(j);

}

template < class DataType >

void MGraph<DataType>::BFSTraverse(int v)

//广度优先遍历图

{

int Q[MaxSize];

//假设队列采用顺序存储且不会发生溢出

int front = -1,rear = -1;

//初始化队列

cout<<vertex[v]; visited[v] = 1; Q[ ++rear] = v;

//被访问顶点入队

while (front != rear)

//当队列非空时

{

v = Q[++front];

//将队头元素出队并送到v中

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

if (arc[v][j] == 1 && visited[j] == 0){

cout<<vertex[j]; visited[j] = 1; Q[ ++rear] = j;

}

}

}

MGraph\_main.cpp：

#include <iostream>

//引入输入输出流

using namespace std;

#include "MGraph.cpp"

//引入类MGraph的成员函数定义

int visited[MaxSize] = {0};

//全局数组变量 visited 初始化

//以下是主函数

int main()

{

char ch[] = {'A','B','C','D','E'};

//顶点信息

MGraph<char> MG(ch,5,6);

//图中顶点的类型为char型

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

//初始化图中所有顶点均为被访问

visited[i] = 0;

cout<<"深度优先遍历是:";

MG.BFSTraverse(0);

//从顶点0出发深度优先遍历图

cout<<endl;

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

//初始化图中所有顶点均未被访问

visited[i] = 0;

cout<<"广度优先遍历序列是:";

//从顶点0出发广度优先遍历图

MG.BFSTraverse(0);

cout<<endl;

return 0;

}