**实验报告（第 6 次）**

**实验名称：图的遍历操作演示程序的设计与实现**

**实验时间： 2021.5.12**

1. **实验目的**

1．掌握图的定义和构造方法；

2．掌握图的存储及遍历方法及应用。

1. **实验内容**

编程实现：

1、建立一个以邻接表形式存储的图；

2、建立一个以邻接矩阵形式存储的图；

3、以邻接表形式存储的图的深度优先遍历；

4、以邻接表形式存储的图的广度优先遍历；

5、以邻接矩阵形式存储的图的深度优先遍历；（选做）

6、以邻接矩阵形式存储的图的广度优先遍历；（选做）

7、退出系统。

1. **源程序及主要算法说明**

**/\*\***

**\*202001010193 刘欢**

**\*/**

**#include<iostream>**

**#define MVNum 100 //最大顶点数**

**#define OK 1**

**using namespace std;**

**typedef int Status;**

**typedef char VerTexType; //假设顶点的数据类型为字符型**

**typedef int OtherInfo; //假设边的信息类型为整型**

**typedef struct ArcNode{ //边结点**

**int adjvex; //该边所指向的顶点的位置**

**struct ArcNode \*nextarc; //指向下一条边的指针**

**OtherInfo info; //和边相关的信息**

**}ArcNode;**

**typedef struct VNode{ //顶点信息**

**VerTexType data;**

**ArcNode \*firstarc; //指向第一条依附该顶点的边的指针**

**}VNode,AdjList[MVNum]; //AdjList表示邻接表类型**

**typedef struct{ //邻接表**

**AdjList vertices;**

**int vexnum,arcnum; //图当前的顶点数和边数**

**}ALGraph;**

**Status LocateVex(ALGraph G,char v){//确定位置，即顶点在G.vertices中的序号**

**int i;**

**for(i=1;i<=G.vexnum;++i){**

**if(v==G.vertices[i].data) return i;**

**}**

**}**

**Status CreateUDG(ALGraph &G){ //采用邻接表表示法，创建无向网G**

**int i,k,j;**

**char v1,v2;**

**ArcNode \*p1,\*p2;**

**cout<<"请输入总顶点数：";**

**cin>>G.vexnum;**

**cout<<"请输入总边数：";**

**cin>>G.arcnum;**

**cout<<"请依次输入各个顶点的数据："<<endl;**

**for( i=1;i<=G.vexnum;++i){ //输入各点，构造表头结点表**

**cout<<"顶点"<<i<<":";**

**cin>>G.vertices[i].data;**

**G.vertices[i].firstarc=NULL;//初始化表头结点的指针域为NULL**

**}**

**for(k=1;k<=G.arcnum;++k){ //输入各边，构造邻接表**

**cout<<"请输入第"<<k<<"条边的两个顶点：";**

**cin>>v1>>v2;**

**i=LocateVex(G,v1); //确定v1在G中的位置，即顶点在G.vertices中的序号**

**j=LocateVex(G,v2); //确定v1在G中的位置，即顶点在G.vertices中的序号**

**p1=new ArcNode; //生成一个新的边结点\*p1**

**p1->adjvex=j; //领结点序号为j**

**p1->nextarc=G.vertices[i].firstarc;**

**G.vertices[i].firstarc=p1;//将新结点\*p1插入顶点vi的边表头部**

**p2=new ArcNode; //生成另一个对称的新的边结点\*p2**

**p2->adjvex=i; //领结点序号为i**

**p2->nextarc=G.vertices[j].firstarc;**

**G.vertices[j].firstarc=p2;//将新结点\*p2插入顶点vj的边表头部**

**}**

**return OK;**

**}**

**void position(ALGraph G){ //打印邻接表**

**int i;**

**ArcNode p;**

**for(i=1;i<=G.vexnum;i++){**

**ArcNode p;**

**p.adjvex=G.vertices[i].firstarc->adjvex;**

**p.nextarc=G.vertices[i].firstarc->nextarc;**

**cout<<i<<"->";**

**while(p.nextarc!=NULL){**

**cout<<p.adjvex<<"->";**

**p=\*p.nextarc;**

**}**

**cout<<p.adjvex<<endl;**

**}**

**}**

**int main(){ //主函数**

**ALGraph G;**

**CreateUDG(G);**

**cout<<"邻接表："<<endl;**

**position(G);**

**return 0;**

**}**

**/\*\***

**\*202001010193 刘欢**

**\*/**

**#include<iostream>**

**#define MaxInt 0 //表示极大值**

**#define MVNum 100 //表示顶点数**

**#define OK 1**

**using namespace std;**

**typedef int Status;**

**typedef char VerTexType; //假设顶点的数据类型为字符型**

**typedef int ArcType; //假设边的权值类型为整型**

**typedef struct{**

**VerTexType vexs[MVNum]; //顶点表**

**ArcType arcs [MVNum][MVNum]; //邻接矩阵**

**int vexnum,arcnum; //图的当前点数和边数**

**}AMGraph;**

**Status LocateVex(AMGraph G,VerTexType v){//确定位置，即顶点在G.vertices中的序号**

**int i;**

**for(i=1;i<=G.vexnum;i++){**

**if(v==G.vexs[i]){**

**return i;**

**}**

**}**

**}**

**Status CreateURN(AMGraph &G) //采用邻接矩阵表示法，创建无向网G**

**{**

**int i,j,k,w;**

**VerTexType v1,v2;**

**cout<<"请输入总顶点数：";**

**cin>>G.vexnum;**

**cout<<"请输入总边数：";**

**cin>>G.arcnum;**

**cout<<"请依次输入各个顶点的数据："<<endl;**

**for(i=1;i<=G.vexnum;++i){**

**cout<<"顶点"<<i<<":";**

**cin>>G.vexs[i];**

**}**

**for(i=1;i<=G.vexnum;++i){ //初始化邻接矩阵，边的权值均设置为极大值MaxInt**

**for(j=1;j<=G.vexnum;++j){**

**G.arcs[i][j]=MaxInt;**

**}**

**}**

**cout<<"请依次输入边的两个顶点及权值"<<endl;**

**for(k=1;k<=G.arcnum;++k){ //构造邻接矩阵**

**cout<<"请输入第"<<k<<"条边的两个顶点：";**

**cin>>v1>>v2;**

**cout<<"请输入第"<<k<<"条边的权值：";**

**cin>>w;**

**i=LocateVex(G,v1); //确定v1在G中的位置，即顶点数组的下标**

**j=LocateVex(G,v2); //确定v1在G中的位置，即顶点数组的下标**

**G.arcs[i][j]=w; //边<v1,v2>的权值设置为w**

**G.arcs[i][j]=G.arcs[i][j]; //设置<v1,v2>的对称边<v2,v1>的权值为w**

**}**

**return OK;**

**}**

**int main() //主函数**

**{**

**int i,j;**

**AMGraph G;**

**CreateURN(G);**

**puts("邻接矩阵：");**

**for(i=1;i<=G.vexnum;i++){**

**for(j=1;j<=G.vexnum;j++){**

**cout<<G.arcs[i][j];**

**cout<<" ";**

**}**

**cout<<endl;**

**}**

**return 0;**

**}**

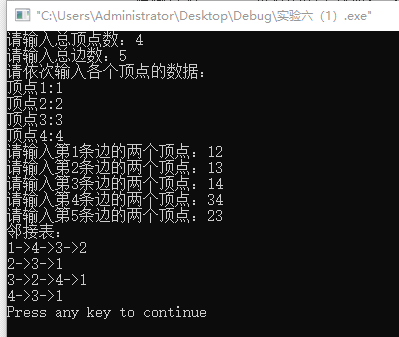
**四．主要问题和解决方案**

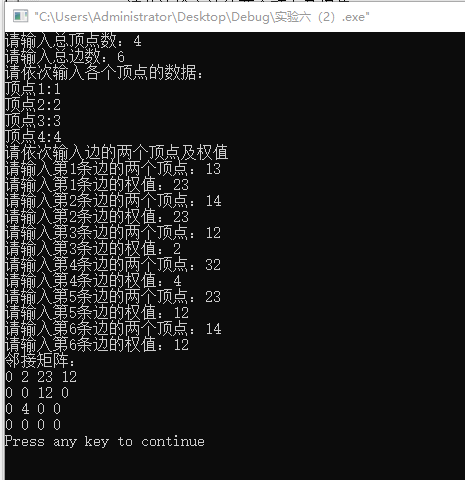
“初始化邻接矩阵的时候，边的权值均要设置成极大值MaxInt”；

“确定位置”不太熟悉；

在结束的时候忘了要加上主函数。

1. **测试数据及结果**

****

****

1. **心得体会与自我评价**

自己仔细看了一遍整个代码，梳理了一下还是能懂的。但是有看书还有老师上课讲的记录下来的代码，挖空比较简单，自己独立完成全部还是有点困难，要继续提升一下。

1. **教师评分**