

**数据结构实验报告**

**实验报告题目：** 图的创建与遍历

**学生姓名**： 刘凌杰

**专 业： 软件工程**

**学 号**： 201720180411

**指导教师： 邹 国 华**

**2018年 11 月 25 日**

**一. 实验内容：**

* **定义图的邻接表类**
* **图的邻接表创建**
* **图的深度优先遍历**
* **图的广度优先遍历**
* **打印图的邻接表**

**二. 实验算法实现：**

**1.** **定义图的邻接表类**

//记录弧

typedef struct ArcNode

{

int adjvex;

int weight;

struct ArcNode \*next;

}ArcNode;

//记录图节点

typedef struct Vnode

{

DataType data;

ArcNode \*link;

}Vnode;

class photo

{

private:

Vnode\* V;

int len;

public:

photo()

{

V=NULL;

len=0;

}

//图的创建

photo(int n);

//销毁图

~photo();

//深度优先遍历

Status depth\_Tralvel(DataType e);

friend void depth\_node\_travel(photo ph,int n,int \*visit);

//广度优先遍历

Status wide\_Tralvel(DataType e);

friend void wide\_node\_travel(photo ph,int n,int \*\*visit);

int locate(DataType e);

//邻接表打印

void print();

};

**2.构造函数（图的创建）**

//构造函数

photo::photo(int len):len(len)

{

char add;

ArcNode \*Arc;

ArcNode \*Arc1;

V= new Vnode[len];

cout<<"请输入各个顶点的值"<<endl;

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

{

cin>>V[j].data;

V[j].link=NULL;

}

system("cls");

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

{

cout<<"顶点"<<V[i].data <<"是否添加邻接点（0不添加）"<<endl;

cin>>add;

if(add!='0'&&V[i].link==NULL)

{

Arc=new ArcNode;

Arc->next=NULL;

cout<<"请输入邻接点下标和权值："<<endl;

cin>>Arc->adjvex;

cin>>Arc->weight;

V[i].link=Arc;

cout<<"是否继续添加邻接点（0停止添加）"<<endl;

cin>>add;

}

while(add!='0')

{

Arc1=new ArcNode;

Arc1->next=NULL;

cout<<"请输入邻接点下标和权值："<<endl;

cin>>Arc1->adjvex;

cin>>Arc1->weight;

Arc->next=Arc1;

Arc=Arc->next;

cout<<"是否继续添加邻接点（0停止添加）"<<endl;

cin>>add;

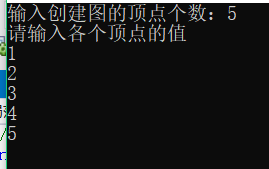
}

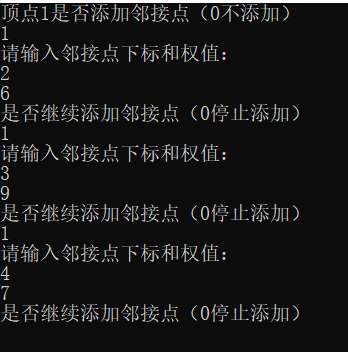
system("cls");

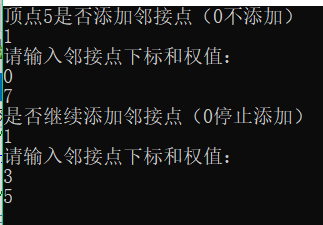
}

cout<<"创建成功！"<<endl;

}







**3.析构函数**

//析构函数

photo::~photo()

{

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

{

ArcNode \*arc1=V[i].link;

ArcNode \*arc2=arc1->next;

while(arc1)

{

delete arc1;

arc1=arc2->next;

arc2=arc2->next;

}

}

delete []V;

V=NULL;

}

4. //根据顶点值获取下标

int photo::locate(DataType e)

{

int i;

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

{

if(V[i].data==e)

{

return i;

}

}

return -1;

}

**5.深度优先遍历**

//全图从指定节点深度优先遍历

Status photo::depth\_Tralvel(DataType e)

{

if(!V)

return ERROR;

int i;

int \*visit=new int[len];

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

{

visit[i]=0;

}

//获取指定节点下标

int flag=this->locate(e);

//节点不存在退出

if(flag==-1)

return ERROR;

//从指定节点开始遍历

depth\_node\_travel(\*this,flag,visit);

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

{

if(visit[i]==0)

depth\_node\_travel(\*this,i,visit);

}

cout<<endl;

delete []visit;

return OK;

}

//子图深度优先遍历

void depth\_node\_travel(photo ph,int n,int \*visit)

{

cout<<ph.V[n].data<<" ";

visit[n]=1;

ArcNode \*nextarc=ph.V[n].link;

while(nextarc)

{

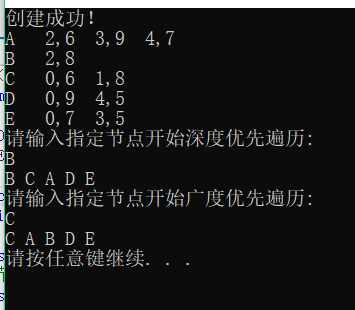
if(visit[nextarc->adjvex]==0)

depth\_node\_travel(ph,nextarc->adjvex,visit);

nextarc=nextarc->next;

}

}



**6.广度优先遍历**

//全图广度优先遍历

Status photo::wide\_Tralvel(DataType e)

{

if(!V)

return ERROR;

int i;

//二维数组表示状态，visit[0][]表示该顶点是否读取过周围顶点，visit[1][]顶点表示是否遍历过

int \*\*visit=new int\*[2];

for(i=0;i<2;i++)

{

visit[i]=new int[len];

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

{

visit[i][j]=0;

}

}

//获取指定节点下标

int flag=this->locate(e);

//节点不存在退出

if(flag==-1)

return ERROR;

//从指定节点开始遍历

wide\_node\_travel(\*this,flag,visit);

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

{

if(visit[0][i]==0)

wide\_node\_travel(\*this,i,visit);

}

cout<<endl;

//释放

delete visit[1];

delete visit[0];

delete visit;

visit=NULL;

return OK;

}

//子图广度优先遍历;

void wide\_node\_travel(photo ph,int n,int \*\*visit)

{

if(visit[1][n]==0)

{

cout<<ph.V[n].data<<" ";

visit[1][n]=1;

}

visit[0][n]=1;

ArcNode\* nextArc=ph.V[n].link;

//将该顶点周围的顶点遍历

while(nextArc)

{

if(visit[1][nextArc->adjvex]==0) //判断是否遍历过该顶点

{

cout<<ph.V[nextArc->adjvex].data<<" ";

visit[1][nextArc->adjvex]=1;

}

nextArc=nextArc->next;

}

nextArc=ph.V[n].link;

while(nextArc)

{

if(visit[0][nextArc->adjvex]==0) //判断是否遍历过周围顶点

wide\_node\_travel(ph,nextArc->adjvex,visit);

nextArc=nextArc->next;

}

}

**7.打印邻接表**

//打印邻接表

void photo::print()

{

ArcNode \*arc;

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

{

cout<<V[i].data<<" ";

arc=V[i].link;

while(arc)

{

cout<<arc->adjvex<<","<<arc->weight<<" ";

arc=arc->next;

}

cout<<endl;

}

}

**8.main()函数**

void main()

{

DataType e;

int i;

int \*visit=new int[5];

for(i=0;i<5;i++)

{

visit[i]=0;

}

cout<<"输入创建图的顶点个数：";

cin>>i;

photo ph(i);

ph.print();

cout<<"请输入指定节点开始深度优先遍历:"<<endl;

cin>>e;

if(ph.depth\_Tralvel(e)==ERROR)

cout<<"不存在该元素或为空图！"<<endl;

cout<<"请输入指定节点开始广度优先遍历:"<<endl;

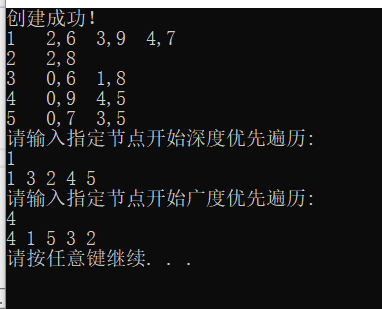
cin>>e;

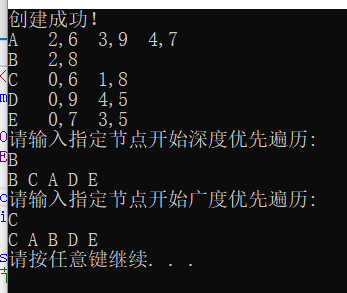
if(ph.wide\_Tralvel(e)==ERROR)

cout<<"不存在该元素或为空图！"<<endl;

system("pause");

}

****

****

**三．实验总结：**

经过一段时间的学习，图的算法的实现基本掌握了，就是代码打得还不太熟练，还有待提高，有些功能不太熟要问同学或查资料，尤其在做遍历运算时。以后在这方面我会多花点时间，让自己更加熟练。