数据结构实习报告

**题目：有向图的遍历**

|  |  |
| --- | --- |
| **学生：** | 计科141 20141514110 杨亚东 |
| 计科141 20141514130 刘宛秋 |
| 计科141 20141514126 张秀芳 |
| **日期：** | 2015年12月31日 |

1. 问题分析

有向图的遍历主要目的在于更深一步的了解图的遍历的问题，图的DFS,BFS的递归和非递归算法的是实现，用有向图和无向图分别实现图的遍历，广度优先遍历和深度优先遍历的实现，用邻接表和邻接矩阵等多种结构存储图，在这次课程设计中主要解决的是给出一个图如何用多种方法完成图的遍历的问题，也包括如何创建一个图，深度优先遍历和广度优先遍历一个图，递归和非递归的方法实现的遍历。程序最终调试运行，初步实现了设计目标。

1. 数据结构
   1. 逻辑结构

图结构：有向图十字链表

线性结构：队列、数组

* 1. 物理结构（物理结构要给出所用到的类型定义，如结点类型、线性表类型等）

顺序存储结构：顶点的存储

链式存储结构：弧的存储，队列结点的存储

1. 算法（不是程序，里面的局部变量就不要定义了，算法内部第一行写算法说明，不用写主函数）

十字链表创建

void create(OLGraph &OL){

//十字链表的创建

while(on\_off){

cout<<"请输入有向图的顶点数和弧数:";

cin>>vexnum>>arcnum;

if(arcnum > vexnum\*(vexnum-1)){

cout<<"顶点数和弧数不符逻辑！再次输入"<<endl;

}else{

on\_off = false;

}

}

OL.vexnum=vexnum;

OL.arcnum=arcnum;

OL.vex = new VexNode[OL.vexnum];

if(!OL.vex){

cout<<"创建失败！"<<endl;

return;

}

for(i = 0; i < OL.vexnum; i++){

cout<<"输入第"<<i+1<<"个顶点的值:";

cin >> data;

for(j = 0;j < OL.vexnum;j++){

if(data == OL.vex[j].data) n = j;

}

if(n==-1){

OL.vex[i].data = data;

OL.vex[i].ver=i;

OL.vex[i].firstin = OL.vex[i].firstout = NULL;

}else{

cout<<"节点已存在！再次输入:"<<endl;

i--;

}

n=-1;

}

for(i = 0;i < OL.arcnum;i++){

cout << "输入第"<<i+1<<"边的弧尾与弧头结点及权值:" << endl;

cin >> v1 >> v2 >> w;

for(j = 0;j < OL.vexnum;j++){

if(v1 == OL.vex[j].data) m = j;

if(v2 == OL.vex[j].data) n = j;

}

if(m==-1){

cout<<"弧尾节点不存在！再次输入:"<<endl;

i--;

}else if(n==-1){

cout<<"弧头节点不存在！再次输入:"<<endl;

i--;

}else{

p = new ArcBox;

p->tailvex = m;

p->headvex = n;

p->weight = w;

p->hlink = OL.vex[n].firstin;

OL.vex[n].firstin = p;

p->tlink = OL.vex[m].firstout;

OL.vex[m].firstout = p;

}

m=n=-1; //初始化弧头弧尾位置坐标

}

}

深度优先遍历

void depthFirst(OLGraph OL,int v){

//深度优先遍历

PolyNomial p;

visited[v]=true;

cout<<OL.vex[v].data<<" ";

p=OL.vex[v].firstout;

while (p!=NULL){

if (!visited[p->headvex])

depthFirst (OL, p->headvex);

p=p->tlink;

}

}

广度优先遍历

void breadthFirst(OLGraph OL){

//广度优先遍历

cout<<endl<<"广度优先遍历"<<endl;

for(int v = 0; v < OL.vexnum; v++){

visited[v] = false;

}

createlist();

for(v = 0; v < OL.vexnum; v++){

add(OL.vex[v]);

if(!visited[v]){

cout<<OL.vex[v].data<<" ";

visited[v] = true;

num++;

if(num == OL.vexnum){

return;

}

}

while(L.head!=L.foot){

node = L.head->next->data;

if(!visited[node.ver]){

cout<<node.data<<" ";

visited[node.ver] = true;

num++;

if(num == OL.vexnum){

return;

}

}

p = node.firstout;

while(p){

add(OL.vex[p->headvex]);

if(!visited[OL.vex[p->headvex].ver]){

cout<<OL.vex[p->headvex].data<<" ";

visited[p->headvex] = true;

num++;

if(num == OL.vexnum){

return;

}

}

p = p->tlink;

}

pop();

}

}

cout<<endl<<"广度优先遍历结束"<<endl;

}

1. 函数调用关系（所有函数间的调用关系，包括main函数）

创建十字链表：Main()->create();

深度优先遍历：initializeOL(初始化遍历条件)->depthFirst(递归遍历);

广度优先遍历：breadthFirst(遍历)->createlist(初始化队列)->add(入队)->pop(出对);

1. 上机调试

顶点信息重复问题：

结果：造成链表混乱

处理：加是否纯在判断若纯在则从新输入；

弧首，弧尾不纯在问题：

结果：造成链表无法构建

处理：加是否纯在判断若不纯在则从新输入；

广度优先遍历非连通：

结果：造成链表遍历不完全

处理：加for循环把未遍历执行遍历方法

1. 总结

经过一个星期的课程实践，收获颇多。也明白了自己在数据结构上存在的一些问题，也解决了一些一直困扰我的问题。以前自己觉得程序很简单，不放在心上。经过课程实践才使我明白做程序需要的东西是很多的，既要懂得程序又要有耐心和恒心。课程设计是对我的一次全面的测试，它既锻炼了我的动手能力也磨练了我的心理素质。这次课程设计是两个人一起完成的，团队的合作是非常重要的，我也在同伴身上学到了很多东西。所以，课本上的知识还是不够的，我应该扩展自己的课外知识，多多阅读课外的相关知识，这样才能对java更加熟悉。所以我要谢谢帮助我解决困难的老师和同学们，没有他们的帮助，我很难完成这个课程设计。这次课程设计时间有限，而且我对知识的理解也不是很清楚，会有很多不足之处，请老师给予指导，我会不断改正，不断完善。

注意：

上机调试要写调试过程中出现了哪些错误，如何解决的，

总结部分要写明组内分工，各完成哪部分，有什么心得体会

每组一个文件夹，命名为：姓名1-姓名2-姓名3-题目名，里面包括：实习报告和源程序

实习报告电子稿命名为：姓名1-姓名2-姓名3-题目名.docx

源程序存为cpp文件