南京信息工程大学 数据结构 实验(实习)报告

实验(实习)名称 实验(实习)日期 得分 指导老师

系专业 班级 姓名 学号

一、实验目的

1、使学生可以巩固所学的有关图的基本知识；

2、熟练掌握图的存储结构；

3、熟练掌握图的两种遍历算法。

1. 实验内容与步骤

#include<iostream>

#include<malloc.h>

using namespace std;

#define MAX 20

int visited[MAX];

typedef struct Anode{

int adjvex;

struct Anode \*next;

}ALnode;

typedef struct vexnode{

char data;

ALnode \*firstal;

}VexHeadNode;

typedef struct{

VexHeadNode adjlist[MAX];

int n;

int e;

}Graph;

void createGraph(Graph &G){

int i,j,k;

ALnode \*p;

cout<<"输入图的顶点数:";

cin>>G.n;

cout<<"输入图的边数:";

cin>>G.e;

cout<<endl;

cout<<"输入图的各顶点(存储序号从0开始)："<<endl;

for(i=0;i<G.n;i++){

cout<<"第"<<i<<"个顶点信息：";

cin>>G.adjlist[i].data;

G.adjlist[i].firstal=NULL;

}

cout<<endl;

cout<<"输入图中的边，顶点序号从0开始:"<<endl;

for(k=0;k<G.e;k++){

cout<<endl;

cout<<"输入第"<<k+1<<"条边:"<<endl;

cout<<"输入出发顶点的序号：";

cin>>i;

cout<<"输入指向顶点的序号:";

cin>>j;

p=(ALnode \*)malloc(sizeof(ALnode));

p->adjvex=j;

p->next=G.adjlist[i].firstal;

G.adjlist[i].firstal=p;

}

}

void printGraph(Graph G){

int i;

ALnode \*p;

for(i=0;i<G.n;i++){

cout<<i<<' '<<G.adjlist[i].data;

p=G.adjlist[i].firstal;

while(p!=NULL){

cout<<"--->"<<p->adjvex<<' ';

p=p->next;

}

cout<<endl;

}

}

void DFSTraverse(Graph G,int v){

ALnode \*p;

cout<<"("<<v<<","<<G.adjlist[v].data<<")"<<' ';

visited[v] = 1;

p=G.adjlist[v].firstal;

while(p!=NULL){

if(visited[p->adjvex]==0){

DFSTraverse(G,p->adjvex);

}

p=p->next;

}

}

void BFSTraverse(Graph G,int v){

int i,j,visited[MAX];

ALnode \*p;

int queue[MAX],front=0,rear=0;

for(i=0;i<G.n;i++){

visited[i]=0;

}

cout<<"("<<v<<","<<G.adjlist[v].data<<")"<<' ';

visited[v]=1;

rear=(rear+1)%MAX;

queue[rear]=v;

while(front!=rear){

front=(front+1)%MAX;

j=queue[front];

p=G.adjlist[j].firstal;

while(p!=NULL){

if(visited[p->adjvex]==0){

visited[p->adjvex]=1;

cout<<"("<<p->adjvex<<","<<G.adjlist[p->adjvex].data<<")"<<' ';

rear=(rear+1)%MAX;

queue[rear]=p->adjvex;

}

p=p->next;

}

}

}

int main(){

Graph G;

int v1,v2,choose;

cin>>choose;

while(choose!=0){

switch(choose){

case 1:{

createGraph(G);

printGraph(G);

break;

}

case 2:{

cin>>v1;

DFSTraverse(G,v1);

for(int i=0;i<G.n;i++){

visited[i]=0;

}

cout<<endl;

break;

}

case 3:{

cin>>v2;

BFSTraverse(G,v2);

cout<<endl;

break;

}

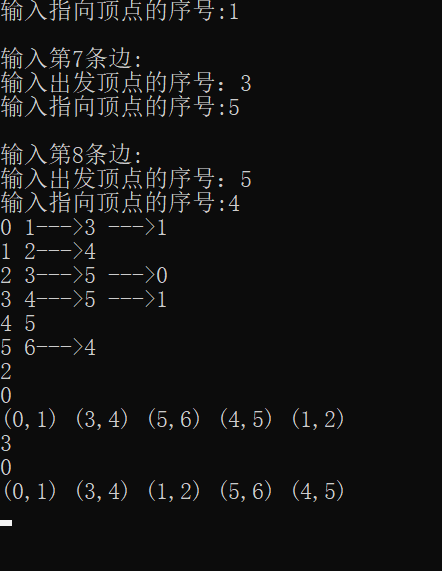
default:cout<<"输入错误，请重新选择！"<<endl;

}

cin>>choose;

}

}



三、体会和总结

深度优先遍历和广度优先遍历结果有很大区别