**实验六 无向连通图的建立和遍历**

应用技术 学院 计算机科学与技术 专业 1 班

学号 202033050040 姓名 李鹏程

一．实习目的

1.掌握无向连通图的建立；

2.掌握无向连通图的深度优先遍历递归算法；

3. 掌握无向连通图的广度优先遍历算法；

二．实习内容

1. 输入无向连通图的顶点和边，在计算机中存储无向图的顶点信息和邻接矩阵；

2. 编写图的深度优先遍历程序，完成对无向连通图的深度优先遍历，并输出深度优先遍历序列；

3. 编写图的广度优先遍历程序，完成对无向连通图的广度优先遍历，并输出广度优先遍历序列；

三、实验步骤

1. 无向图的顶点表定义；

2. 无向图的邻接矩阵定义；

3. 创建无向图并存储相应的顶点表和邻接矩阵；

4. 使用深度优先遍历算法对创建的无向图进行深度优先遍历，并输出结果；

5. 使用广度优先遍历算法对创建的无向图进行广度优先遍历，并输出结果；

四．程序运行

1.程序代码

#include "stdio.h"

#include <string.h>

#include "stdlib.h"

#include "Linkqueue.h" //将队列操作的基本函数文件加入

#include <iostream>

using namespace std;

// 函数结果状态代码

#define TRUE 1

#define FALSE 0

#define MAX\_NAME 5 // 顶点字符串的最大长度+1

#define MAX\_INFO 20 // 相关信息字符串的最大长度+1

typedef int VRType;

typedef char VertexType[MAX\_NAME];

#define MAX\_VERTEX\_NUM 26 // 最大顶点个数

enum GraphKind{DG,DN,UDG,UDN}; // {有向图,有向网,无向图,无向网}

typedef struct

{

VRType adj; // 顶点关系类型。对无向图，用1(是)或0(否)表示相邻否；

}ArcCell,AdjMatrix[MAX\_VERTEX\_NUM][MAX\_VERTEX\_NUM]; // 二维数组

struct MGraph

{

VertexType vexs[MAX\_VERTEX\_NUM]; // 顶点数组

AdjMatrix arcs; // 邻接矩阵

int vexnum,arcnum; // 图的当前顶点数和弧数

GraphKind kind; // 图的种类标志

};

int visited[MAX\_VERTEX\_NUM]={0};

int LocateVex(MGraph G, VertexType u){

// 初始条件：图G存在，u和G中顶点有相同特征

// 操作结果：若G中存在顶点u，则返回该顶点在图中位置；否则返回-1

int i;

for(i=0;i<G.vexnum;++i)

if(strcmp(u,G.vexs[i])==0)

return i;

return -1;

}

void CreateUDG(MGraph &G)

{ // 采用数组(邻接矩阵)表示法，构造无向图

int i,j,k,w,IncInfo;

char s[MAX\_INFO];

VertexType va, vb; //va,vb字符数组

printf("请输入无向图G的顶点数,边数: ");

scanf("%d%d",&G.vexnum,&G.arcnum);

printf("请输入%d个顶点的值(<%d个字符):\n",G.vexnum,MAX\_NAME);

for(i=0;i<G.vexnum;++i) // 构造顶点数组

scanf("%s",G.vexs[i]);

for(i=0;i<G.vexnum;++i) // 初始化邻接矩阵

for(j=0;j<G.vexnum;++j)

{

G.arcs[i][j].adj=0; //矩阵初始化为0

}

printf("请输入%d条边的(以空格作为间隔如v1 v2 ): \n",G.arcnum);

for(k=0;k<G.arcnum;++k)

{

scanf("%s%s", va,vb);

i=LocateVex(G,va);

j=LocateVex(G,vb);

G.arcs[i][j].adj=1;

G.arcs[j][i].adj=1; // 无向图 对称矩阵

}

G.kind=UDG;

}

void Display(MGraph G)

{ // 输出邻接矩阵存储表示的图G

int i,j;

char s[10];

switch(G.kind)

{

case DG: strcpy(s,"有向图");

break;

case DN: strcpy(s,"有向网");

break;

case UDG:strcpy(s,"无向图");

break;

case UDN:strcpy(s,"无向网");

}

printf("%d个顶点%d条边的%s。顶点依次是: ",G.vexnum,G.arcnum,s);

for(i=0;i<G.vexnum;++i) // 输出G.vexs

printf("%s ", G.vexs[i] );

printf("\nG.arcs:\n"); // 输出G.arcs.adj

for(i=0;i<G.vexnum;i++)

{

for(j=0;j<G.vexnum;j++)

printf("%5d",G.arcs[i][j].adj);

printf("\n");

}

}

int FirstAdjVex(MGraph G,int v) //返回v的第一个邻接点

{int i,j;

for(j=0;j<G.vexnum;j++)

if(G.arcs[v][j].adj==1)

{return j;

}

return -1;

}

int NextAdjVex(MGraph G,int v, int w) //返回v相对于w的下一个邻接点

{int i,j;

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

if(G.arcs[v][j].adj==1&&visited[j]==0)

return j;

return -1;

}

void DFS(MGraph G, int v) { // 算法7.5

// 从第v个顶点出发递归地深度优先遍历图G。

int w;

visited[v] = true;

printf("%s ",G.vexs[v]);

for (w=FirstAdjVex(G,v); w!=-1; w=NextAdjVex(G, v, w))

if (!visited[w]) // 对v的尚未访问的邻接顶点w递归调用DFS

DFS(G,w);

}

void BFS(MGraph G, Status (\*Visit)(int v)) {// 算法7.6

// 按广度优先非递归遍历图G。使用辅助队列Q和访问标志数组visited。

int v,w;

LinkQueue Q;

int u;

for (v=0; v<G.vexnum;++v) visited[v] = FALSE;

InitQueue(Q); // 置空的辅助队列Q

for (v=0; v<G.vexnum;++v)

if(!visited[v]){

visited[v] = TRUE;

Visit(w);

EnQueue(Q,v);

while(!QueueEmpty(Q)){

DeQueue(Q,u);

for(w = FirstAdjVex(G,u);w != -1;w = NextAdjVex(G,u,w))

if(!visited[w]){

visited[w] = TRUE;

Visit(w);

EnQueue(Q,w);

}

}

}//if

}

int printc(int v){

printf("v%d ",v+1 );

return 0;

}

int main()

{

int i,j;

MGraph g;

CreateUDG(g); // 构造无向图

Display(g); // 输出无向网

printf("无向连通图深度优先遍历的序列为：\n");

DFS(g,0); //以g中位置为0的顶点为源点，遍历

printf("\n无向连通图广度优先遍历的序列为：\n");

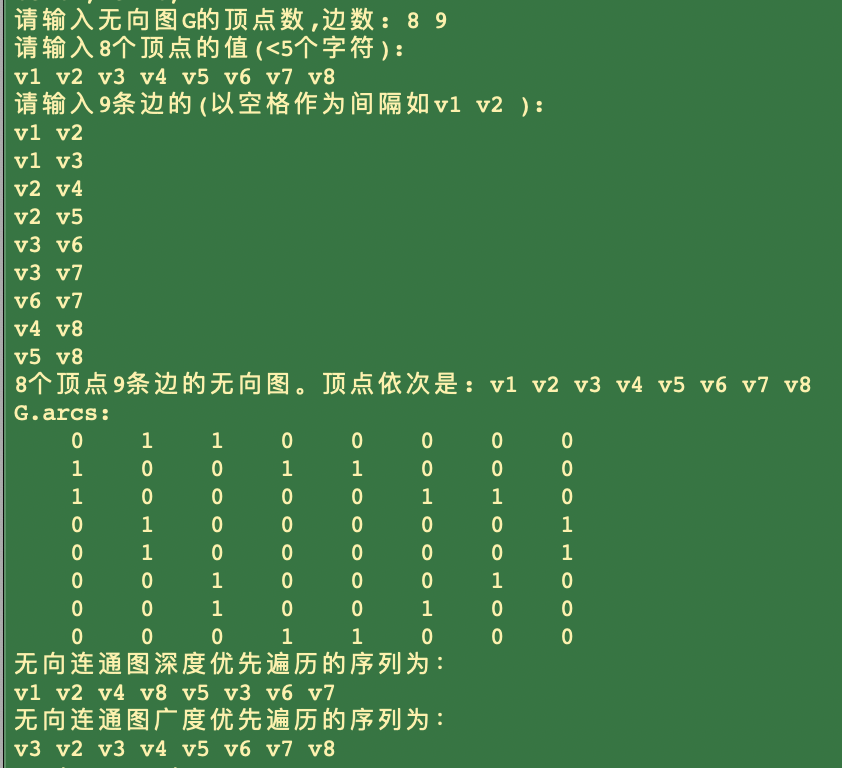
BFS(g,printc); //广度遍历

return 0;

}

1. 运行程序

3.运行结果（截图）



五．实验小结

通过本次实习，我掌握了图的各种类型，例如，无向图(网)，有向图(网)等。并了解了相关概念的基本性质，通过性质的学习，掌握了无向图连通图的创建以及建立其邻接矩阵，并利用深度优先和广度优先来进行遍历，从而完成本次实验目标。