山东大学 软件 学院

数据结构 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：201900302030 | 姓名：邵嘉明 | | 班级： 数字媒体技术 |
| 实验题目：实验七 图的操作 | | | |
| 实验学时：2 | | 实验日期：2020.12.5 | |
| 实验目的：  掌握无向图的创建、遍历方法。 | | | |
| 硬件环境：PC | | | |
| 软件环境：Visual Studio 2019 | | | |
| 实验步骤与内容：   1. 创建图类，存储结构使用邻接矩阵。 2. 输入图的节点数n（小于10个）、边数m，节点分别用1-n代表。 3. 采用“起始节点，终止节点，权值”输入图的m条边，创建图。 4. 输出从节点1开始的BFS遍历，在遍历过程中，如有多个可以选择的节点，则优先选择编号较小的节点。 5. 输出从节点1开始的DFS遍历，在遍历过程中，如有多个可以选择的节点，则优先选择编号较小的节点。 6. 输出从第1节点到第n节点最短路径的长度，如果没有路经，输出0。 | | | |
| 结论分析与体会：  掌握了无向图的创建、遍历方法。 | | | |

**本次实验主要代码：**

#include <iostream>

#include <queue>

#include <cstdio>

#include <stdlib.h>

#include <string.h>

#include <string>

#include <vector>

#include <algorithm>

using namespace std;

int num\_01,num\_02,array\_01[11][11],array\_02[11],require[11],bfs\_require[11],dfs\_require[11];

int check\_count{0};

void bfs\_func(int valueIn)

{

int count{1};

queue<int> queue\_00;

bfs\_require[valueIn]=1;

queue\_00.push(valueIn);

cout<<'1'<<",";

while(!queue\_00.empty())

{

int w=queue\_00.front();

queue\_00.pop();

for(int u{1};u<=num\_01;++u)

if(array\_01[w][u]!=0&&bfs\_require[u]==0)

{

queue\_00.push(u);

++count;

if(count==check\_count)cout<<u<<endl;

else cout<<u<<',';

bfs\_require[u]=1;

}

}

}

void dfs\_func(int valueIn)

{

dfs\_require[valueIn]=1;

++check\_count;

for(int k{1};k<=num\_01;++k)

{

if(array\_01[valueIn][k]!=0&&dfs\_require[k]==0)dfs\_func(k);

}

}

void min\_path()

{

for(int i{2};i<=num\_01;++i)

{

if(array\_01[1][i]!=0)array\_02[i]=array\_01[1][i];

else array\_02[i]=99999;

}

require[1]=1;

for(int i{1};i<=num\_01;++i)

{

int temp{99999},u{1};

for(int j{1};j<=num\_01;++j)

{

if(require[j]==0&&array\_02[j]<temp)

{

temp=array\_02[j];

u=j;

}

}

require[u]=1;

for(int j=1;j<=num\_01;j++)

if(require[j]==0&&array\_01[u][j]!=0)

if(array\_02[j]>array\_02[u]+array\_01[u][j])

array\_02[j]=array\_02[u]+array\_01[u][j];

}

cout<<array\_02[num\_01]<<endl;

}

int main(int argc, char \*argv[])

{

cout<<"Input"<<endl;

char char\_00;

cin>>num\_01>>char\_00>>num\_02;

int tempA,tempB,tempC;

for(int i=1;i<=num\_02;i++)

{

cin>>tempA>>char\_00>>tempB>>char\_00>>tempC;

array\_01[tempA][tempB]=array\_01[tempB][tempA]=tempC;

}

cout<<"Output"<<endl;

dfs\_func(1);

bfs\_func(1);

for(int i=1,j=1;i<=num\_01;i++){

if(dfs\_require[i]!=0){

if(j==check\_count)

{

cout<<i<<endl;

break;

}

else cout<<i<<",";

++j;

}

}

if(bfs\_require[num\_01]==0)cout<<'0'<<endl;

else min\_path();

cout<<"End"<<endl;

return 0;

}