 实 验 报 告

课程名称： 数据结构（C++版）

题 目： 医院选址问题

专业班级： 18空间

姓 名： 邓珊珊

学 号： 1812001107

指导老师： 袁莹

实验时间： 2020.01.10

第一次实验报告

班级：18空间 姓名：邓珊珊 学号：1812001107

1. 问题描述

n个村庄之间的交通图可以用有向图来表示，图中边<vi，vj>上的权重表示从村庄i到村庄j的道路长度。现在要从这n个村庄中选择一个村庄新建一所医院，问这所医院应建在哪个村庄，才能使所有的村庄离医院比较近？

1. 基本要求
2. 建立数据模型，设计存储结构；
3. 设计算法完成问题求解；
4. 分析算法的时间复杂度。
5. 设计思想

医院选址问题实际上是求有向图的中心点。首先定义顶点的偏心度。

1. 伪代码：
2. 对带权有向图，调用Floyd算法，求每对顶点间最短路径长度的矩阵；
3. 对最短路径长度矩阵的每列求大值，即得到各顶点的偏心度；
4. 具有最小偏心度的顶点即为所求。
5. 实验内容

（代码）

头文件：Hospital.h

#ifndef Hospital\_h

#define Hospital\_h

const int MaxValue = 100; //定义const常量 100

const int N = 5; //假设图中有5个顶点

class Hospital

{

public:

Hospital(char v[N],int e[N][N], int n); //构造函数

~Hospital( ); //析构函数

void Floyd(int dist[N][N]); //求任意两个顶点之间的最短路径

int Center( ); //求图的中心点，返回中心点的下标

private:

char vertex[N] ; //一维数组存放顶点信息

int edge[N][N]; //二维数组存放邻接关系

int vertexNum; //顶点个数

};

#endif /\* Hospital\_h \*/

Hospital.cpp

#include <iostream>

using namespace std;

#include "Hospital.h"

Hospital :: Hospital(char v[N], int e[N][N], int n)

{

for (int i = 0; i < n; i++) //初始化顶点信息

vertex[i] = v[i];

for(int i = 0; i < n; i++) //初始化邻接矩阵

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

edge[i][j] = e[i][j] ;

vertexNum = n;

}

Hospital :: ~Hospital( )

{

}

void Hospital :: Floyd(int dist[N][N])

{

int i, j, k;

for (i = 0; i < vertexNum; i++) //初始化结果矩阵dist

for (j = 0; j < vertexNum; j++)

dist[i][j] = edge[i][j];

for (k = 0; k < vertexNum; k++) //进行N次迭代

for (i = 0; i < vertexNum; i++) //顶点i和j之间是否经过顶点k

for (j = 0; j < vertexNum; j++)

if (dist[i][k] + dist[k][j] < dist[i][j])

dist[i][j] = dist[i][k] + dist[k][j];

}

int Hospital :: Center( )

{

int wayCost, minCost = MaxValue;

int i, k, index, dist[N][N]; //矩阵dist存储顶点之间的最短路径长度

Floyd(dist);

for (k = 0; k < vertexNum; k++) //依次求每个顶点的往返代价

{

wayCost = 0; //往返代价初始化为0

for (i = 0; i < vertexNum; i++) //顶点i到其他顶点的路径长度之和

wayCost = wayCost + dist[i][k];

for (i = 0; i < vertexNum; i++) //其他顶点到顶点i的路径长度之和

wayCost = wayCost + dist[k][i];

if (wayCost < minCost) {

minCost = wayCost;

index = k; //顶点k为当前的中心点

}

}

return index; //返回中心点的下标

}

Hospital\_main.cpp:

#include <iostream>

using namespace std;

#include "Hospital.cpp"

int main( )

{

char vertex[N] = {'A','B','C','D','E'};

int edge[N][N] = {{0,13,MaxValue,4,MaxValue},

{13,0,15,MaxValue,5},

{MaxValue,MaxValue,0,12,MaxValue},

{4,MaxValue,12,0,MaxValue},

{MaxValue,MaxValue,6,3,0}};

int minPoint;

Hospital H{vertex, edge, 5}; //定义对象H

minPoint = H.Center( ); //调用函数Center求中心点

cout << "应该设在点:" << vertex[minPoint] << endl;

return 0;

}