《数据结构》课程实验报告

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 专业名称 | 计算机 | 年级 | 2017 | 班级 | 2 |
| 学生姓名 | 王汝芸 | 学号 | 201711010202 | 指导教师 | 郑志华 |
| 实验题目 | 实验8 用floyd算法求每对顶点间的最短路 | | | 提交时间 | 2018．5.21 |

一、实验目的和要求

掌握floyd算法

二、实验环境

Visual Studio2017

三、实验内容及实施

用二维数组的比较赋值即可完成

**实验8：**用floyd算法求每对顶点间的最短路

**实验要求：**

**1、写出floyd 算法的算法思想**

通过一个图的权值[矩阵](https://baike.baidu.com/item/%E7%9F%A9%E9%98%B5" \t "_blank)求出它的每两点间的[最短路径](https://baike.baidu.com/item/%E6%9C%80%E7%9F%AD%E8%B7%AF%E5%BE%84)矩阵。

从图的带权[邻接矩阵](https://baike.baidu.com/item/%E9%82%BB%E6%8E%A5%E7%9F%A9%E9%98%B5" \t "_blank)A=[a(i,j)] n×n开始，[递归](https://baike.baidu.com/item/%E9%80%92%E5%BD%92" \t "_blank)地进行n次更新，即由矩阵D(0)=A，按一个公式，构造出矩阵D(1)；又用同样地公式由D(1)构造出D(2)；……；最后又用同样的公式由D(n-1)构造出矩阵D(n)。矩阵D(n)的i行j列元素便是i号顶点到j号顶点的最短路径长度，称D(n)为图的[距离矩阵](https://baike.baidu.com/item/%E8%B7%9D%E7%A6%BB%E7%9F%A9%E9%98%B5" \t "_blank)，同时还可引入一个后继节点矩阵path来记录两点间的最短路径。

**2、画出测试用例的费用矩阵对应的图，其他要求不变**

10

1

2

15

6

3

8

1000

4

4

3

2

**模块图**

建立N\*N二维数组

查找每次开放点

比较原路径与开放路径长短

取短路径

输出

1、写出floyd 算法的算法思想

2、画出测试用例的费用矩阵对应的图，其他要求不变

**【源程序】**

/\*用floyd算法求每对顶点间的最短路\*/

#include<stdio.h>

#include<stdlib.h>

#define N 4

//赋值函数

void assignment(int a[][4],int b[][4])

{

int i, j;

for (i = 0; i < N; i++)

{

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

{

b[i][j] = a[i][j];

}

}

}

void print(int a[][4])

{

int i, j;

for (i = 0; i < N; i++)

{

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

{

printf("%d\t", a[i][j]);

}

printf("\n");

}

}

//求最短路径函数,k为开放点

void min(int a[][4],int b[][4],int k)

{

int i, j;

for (i = 0; i < N; i++)

{

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

{

if (a[i][j] > a[i][k-1] + a[k-1][j])

{

b[i][j] = a[j][k-1] + a[k-1][j];

}

}

}

}

int main()

{

int i, j, k;

int a[N][N] = {0,10,32767,32767,15,0,6,32767,3,1000,0,4,32767,8,2,0};

printf("如果没有路用无穷（32767）表示\n");

printf("初始路径为：\n");

print(a);

putchar('\n');

int dist1[N][N],dist2[N][N],dist3[N][N],dist4[N][N];

//1

assignment(a, dist1);//给N赋值

min(a, dist1, 1);

printf("开放第一个节点的dist1为：\n");

print(dist1);

putchar('\n');

//2

assignment(dist1, dist2);

min(dist1, dist2, 2);

printf("开放第二个节点的dist2为：\n");

print(dist2);

putchar('\n');

//3

assignment(dist2, dist3);

min(dist2, dist3, 3);

printf("开放第三个节点的dist3为：\n");

print(dist3);

putchar('\n');

//4

assignment(dist3, dist4);

min(dist3, dist4, 4);

printf("开放第四个节点的dist4为：\n");

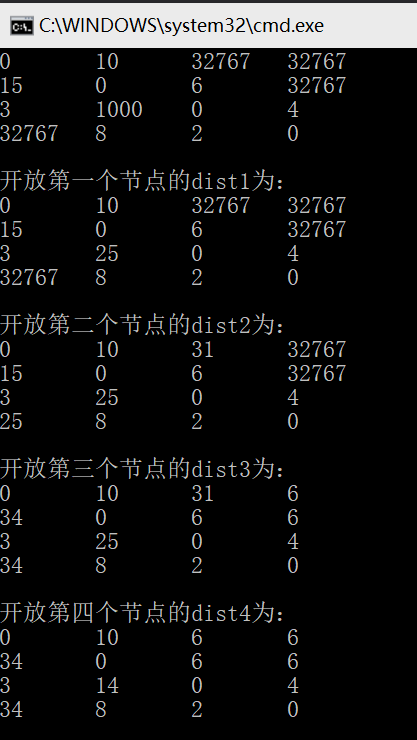
print(dist4);

putchar('\n');

return 0;

}

**四、实验结果 (程序的执行结果)**



**五、实验讨论（可选）**

无