离散二组大作业上机报告

组员：（实验1）张帅豪 赵宇轩 王辰毓 田峥 邱志博

（实验2）王柏涵 冯晴 李懿哲 李有庆 孙宇琪 吴琦

### 实验基本信息

小组选题内容：求等价类（三）、判断图的连通性（十）

操作系统：win10

实验语言：c（实验1）和c++（试验2）

### 实验1：求等价类

实验题目

例题P106

**1、实验类型：**设计性

**2、实验目的**

通过算法设计并编程实现对给定集合上的关系是否为自反的、对称的和传递关系的判断，加深学生对等价关系和等价类的理解，掌握求等价类的方法。

**3、实验内容**

给定一个集合{1，2，…..，n}，及其上的一个等价关系R，求R上的等价类。

**4、实验原理**

给定任意关系，欲判断R是否为等价关系，可用实验八所给出的程序。但是，如果给定一个等价关系，那么它的关系矩阵必为对称矩阵。为了节省存储空间，只要存放关系矩阵的一半就可以了（另一半与这一半相同）。我们可以用一维数组来存放关系矩阵的下三角矩阵（包括主对角线在内），具体对应关系如下：

根据等价类的定义可知，等价类内的各元素之间均有R关系，所以在构造等价类时，只要依据所给关系矩阵，把所有相互有R关系的各元素归为一类就可以了。在输出时，把每一类显示在同一行上。

**5、实验仪器设备或软件环境及工具**

运行Windows 或Linux操作系统的PC机，具有gcc(Linux)、Turboc、Vc(Windows)等C语言的编译环境。

**6、实验要求**

复习等价关系和等价类的定义，实验由几个人一组完成。所编程序能够通过编译，并能够实现求出给定集合上等价关系的等价类。

**7、实验报告要求**

（1）写出实验过程中遇到的问题及其解决过程。

（2）写出类c的算法并编写一个程序求出给定集合上的等价关系的等价类。

（3）写出实验结束时的程序清单及运行结果及实验总结。

报告内容

等价类的前提是我们先定一个集合和二元关系。因为二元关系的不确定性，有两种方案；

**for**(i=1;i<=n;i++)

{

**for**(j=1;j<=n;j++)

{

//scanf("%d",&c[i][j]);//为手动输入

**if**((a[i]%3)==(b[j]%3))

{

c[i][j]=1;

}

**else**

{

c[i][j]=0;

}

}

}

//定义二元关系，并建立关系矩阵

//1.直接输入0 or 1 代表，有点可以脱离具体的题，适应更大

//2.在程序内写入关系式，计算机判断赋值0 or 1

构建二元关系矩阵后，判断该关系是否等价

从自反性、对称性、传递性三个角度进行判断，只有三个都成立，该二元关系才等价。

**for**(i=1,j=1;j<=n;i++,j++)

{

**if**(c[i][j]!=1)

{

judge=0;

**break**;

}

}

//判断二元关系的自反性

**for**(i=1;i<=n;i++)

{

**for**(j=1;j<=n;j++)

{

**if**(c[i][j]^c[j][i])

{

judge=0;

**break**;

}

}

}

//判断二元关系的对称性

**for**(i=1;i<=n;i++)

{

**for**(j=1;j<=n;j++)

{

**if**(c[i][j]==1)

{

**for**(t=1;t<=n;t++)

{

**if**(c[j][t]==1)

{

**if**(c[i][t]==0)

{

judge=0;

**break**;

}

}

}

}

}

}

//判断二元关系的传递性

如果为等价关系，我们接下来求等价类。设R是定义在集合A上的等价关系。那么R的等价类构成S的划分。反过来，给定集合S的划分{https://gss3.bdstatic.com/-Po3dSag_xI4khGkpoWK1HF6hhy/baike/s%3D17/sign=507f52ebf6039245a5b5e50886947dff/622762d0f703918f335d99ec5b3d269759eec40c.jpg|i∈I},则存在一个等价关系R，它以集合https://gss2.bdstatic.com/-fo3dSag_xI4khGkpoWK1HF6hhy/baike/s%3D61/sign=30e8004edb09b3deefbfe769cdbf04e8/8326cffc1e178a82dfa7d886fc03738da977e879.jpg作为它的等价类。

**if**(judge==1)//由自反性、对称性、传递性 判断该二元关系是否为等价关系

{

printf("等价类如下\n") ;

**for**(i=1;i<=n;i++)

{

**for**(j=1;j<=n;j++)

{

**if**(c[i][j]==1)

{

printf("%d ",b[j]);

}

}

printf("\n");

}

}

//分割等价类

**else**

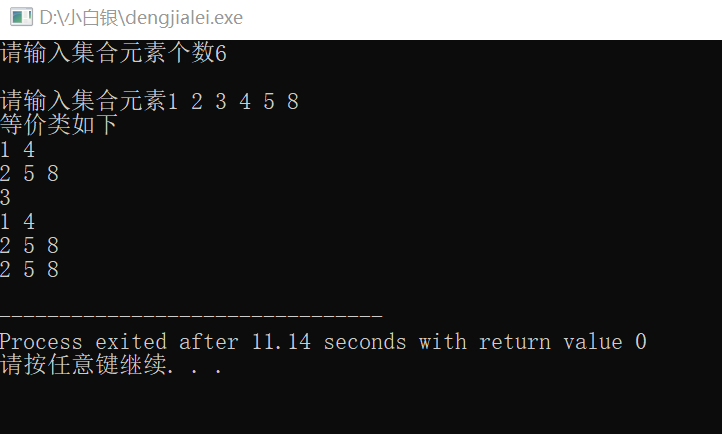
{

printf("该二元关系非等价关系\n");

}

从而求出等价类。但有缺点，会重复。考虑过用链表，这样可以剔除出现过的元素，不会引起重复，也能减少运算，但昨晚看了一堆，有点迷。没有写出来。总体的时间复杂度为n+n\*n+n+n\*n+n\*n\*n+n\*n=2n+3n\*n+n\*n\*n。运算时间过长，算法不够简洁，仍需优化。

测试结果，给定一个集合{1，2，3，4，5，8}，二元关系为模3同余



### 实验2：图的连通性

实验题目

**1、实验类型：**设计性

**2、实验目的**

通过算法设计并编程实现，使学生掌握利用计算机语言判别图的连通性的基本方法。

**3、实验内容**

给定n个结点的有向图的邻接矩阵，可判断该图是否为强连通的，单向连通的，或弱连通的。

**4、实验原理**

对于给定的邻接矩阵A，我们可以用前面给出的可达矩阵Warshall算法求出A所表示的图的可达矩阵P。对于可达矩阵P来说，如果P的所有元素均为1，则所给的有向图是强连通的；对于P的所有元素（除主对角线元素外）Pij来说，均有：Pij+Pji>0，则所给有向图是单向连通的。当所给有向图既不是强连通的，又不是单向连通的时候，我们改造邻接矩阵为：对于矩阵A中所有的元素（除主对角线的元素外）aij，若aij=1或aji=1，则1aij且1aji。对于这样改造之后所得到的新的矩阵A’（A’相当于原有向图忽略方向之后所得到的无向图的邻接矩阵），再用前面所述的方法进行判断，当P’的所有元素（除主对角线的元素外）均为1时，原有向图是弱连通图；否则，原有向图是不连通的。

**5、实验仪器设备或软件环境及工具**

运行Windows 或Linux操作系统的PC机，具有gcc(Linux)、Turboc、Vc(Windows)等C语言的编译环境。

**6、实验要求**

复习图的强连通、单向连通和弱连通的定义，实验由几人一组完成。所编程序能够通过编译，并能够对给定n个结点的有向图的邻接矩阵，判断该图是否为强连通的，单向连通的，或弱连通的。

**7、实验步骤及注意事项**

（1）输入邻接矩阵A（n，n）。

（2）A（n，n）P（n，n）。

（3）调用求可达矩阵子程序求出可达矩阵P。

（4）调用强连通或单向连通子程序。

（5）若为强连通或单向连通的，则输出其标志，转结束；否则转（6）。

（6）改造A阵为 A’，且A’ P（n,n）。

（7）调用求可达矩阵子程序。

（8）调用判断连通或单向连通子程序。

（9）若为强连通的，则输出原有向图是弱连通的；否则输出原有向图是非连通的。

（10）结束。

**8、实验报告要求**

（1）写出实验过程中遇到的问题及其解决过程。

（2）写出类c的算法，并编写一个程序求出给定n个结点的有向图的邻接矩阵，据此判断该图是否为强连通的，单向连通的，或弱连通的。

（3）写出实验结束时的程序清单及运行结果及实验总结。

报告内容

（1）实验中遇到的问题：

1. 第一次编写时，忘记将基图再次进行Warshall函数求可达矩阵，结果弱联通的情况都被判定为不连通；

解决过程：在判定完不是单向联通过后，将邻接矩阵A（Matirx）转化为AvA^T,再利用Warshall算法求出其可达矩阵，再对它进行判定是否为弱联通。

1. 后来检查时，发现直接将邻接矩阵A的可达矩阵P转化为PvP^T,使得判定过程有误；

解决过程：之前的Warshall函数和judge都没有传参，而是直接利用全局变量进行，现在改为传参进行；而且额外创建一个数组matrix储存起来最开始输入的邻接矩阵Matirx的元素，在利用Matirx判定完是否为单向联通过后，，再利用matrix将邻接矩阵A （matrix）转化为AvA^T，在利用Warshall算法求出其可达矩阵，再对它进行判定是否为弱联通。

（2）算法思路如下：

首先我们得明白如何利用邻接矩阵A和可达矩阵P判断图的连通性：

有向线图是强连通图，当且仅当它的可达矩阵P的所有元素均为1；

有向线图是单向连通图，当且仅当PVP^T的所有元素均为1；

有向线图是弱连通图，当且仅当以AVA^T作为邻接矩阵求得的可达矩阵P‘中的所有元素均为1；

然后再来写具体思路：

先根据输入的邻接矩阵，调用warshall函数转化为可达性矩阵，再调动判断（Judge）函数判断是否为强连通，若为强连通，

则输出“强连通” 并结束程序，若不是强连通，则判断是否是单向联通，若是结束程序，若不是单向连通则进行矩阵转换为无向，

并将转换过的矩阵运用Warshall函数转化为无向图（基图）的可达性矩阵，判断若基图的是连通的，则该矩阵为弱联通矩阵，

反之则为不连通。

（3）程序代码：

首先我们先编写Judge函数：判断传入的矩阵中是否有为1的元素，如果有judge变为1，如果没有judge仍为1。

#define max 500

**int** judge,n;

**void** Judge(**int** M[max][max]) //判断是否是强连通和弱联通的函数

{

judge = 0;

**for**(**int** i = 0;i<n;i++)

{

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

{

**if**(M[i][j] == 0) judge = 1;

}

}

}

编写Warshall函数，首先了解Warshall算法的原理,Warshall算法其实就是求关系的传递闭包的一个有效算法，具体过程是：

（1）置新矩阵A=M;

（2）置k=1;

（3）对所有i如果A[i,k]=1，则对j=1..n执行：

A[i,j]←A[i,j]∨A[k,j];

（4）k增1;

（5）如果k≤n，则转到步骤（3），否则停止。

所得的矩阵A即为关系R的传递闭包t(R)的关系矩阵。

Warshall算法时间复杂度为O（n^3）,但已经是非常高效的求解传递闭包的算法，

**void** Warshall(**int** M[max][max])

{

**for**(**int** j = 0;j < n;j++)//将邻接矩阵转化为可达性矩阵 （Warshall算法 ）

{

**for**(**int** i = 0;i < n;i++)

{

**if**(M[i][j]==1)

**for**(**int** k = 0;k<n;k++)

{

M[i][k] = M[i][k]||M[j][k];

}

}

M[j][j] = 1;

}

}

以下则是之前描述过的过程代码的具体实现：

**int** Matrix[max][max];//矩阵最大可容纳500个节点

**int** matrix[max][max];

**int** i,j;

cout<<"请输入节点个数"<<endl;

cin>>n;

cout<<"请输入邻接矩阵"<<endl;

**for**(i = 0;i < n;i++) //输入邻接矩阵

{

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

{

cin>>Matrix[i][j];

matrix[i][j] = Matrix[i][j];

}

}

Warshall(Matrix);

Judge(Matrix); //判断是否为强连通，根据是judge的值

**if**(judge == 0)

{

cout<<"强连通"<<endl;

**return** 0;

}

judge = 0;

**for**(i = 0;i<n;i++)//判断是否为单向连通

{

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

{

**if**(Matrix[i][j]+Matrix[j][i]<=0) judge = 1;

}

}

**if**(judge==0)

{

cout<<"单向联通"<<endl;

**return** 0;

}

**for**(i =0;i<n;i++) //将邻接矩阵转化为与其转置矩阵的析取

{

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

{

**if**(matrix[i][j]==1) matrix[j][i]=1;

}

}

Warshall(matrix);

Judge(matrix); //判断是否为弱联通，通过judge的值进行判断

**if**(judge == 0)

{

cout<<"弱联通"<<endl;

**return** 0;

}

**else**{

cout<<"不连通"<<endl;

**return** 0;

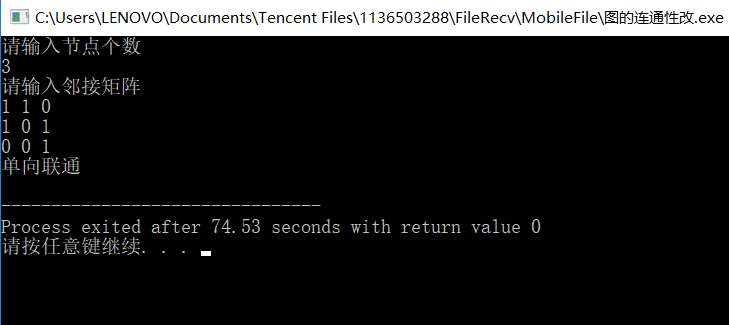
}

}

试运行了几个例子：①当输入邻接矩阵【1，1，0】，判断出来为单向联通；

【0，0，1】

【0，0，1】



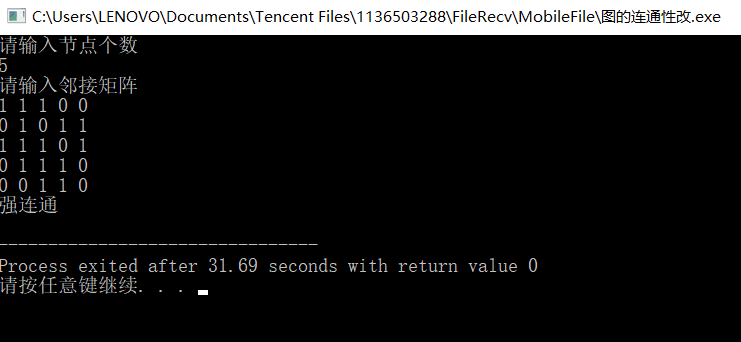
1. 当输入邻接矩阵 【1，1，1，0，0】，判断出来为强联通；

【0，1，0，1，1】

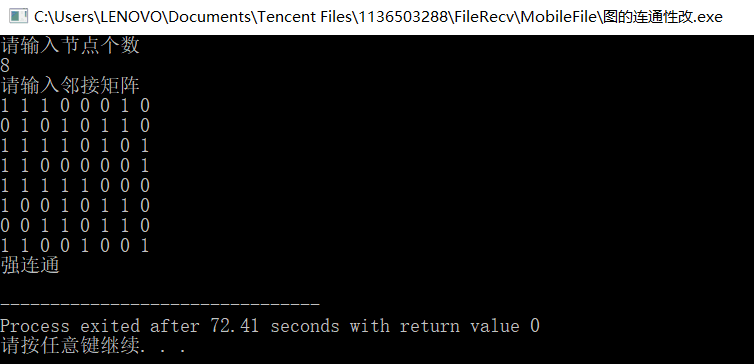
【1，1，1，0，1】

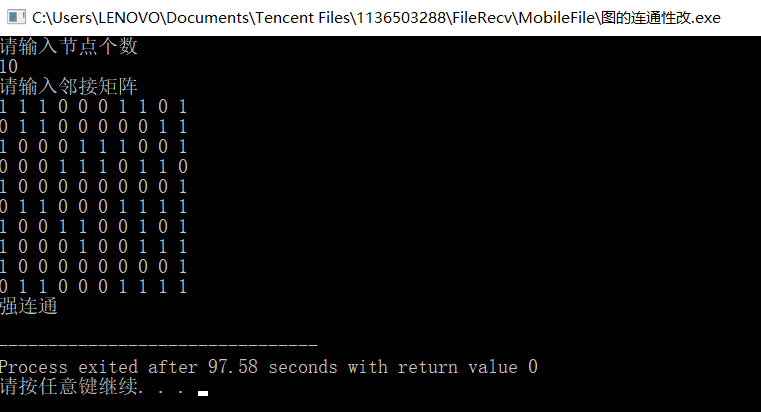
【0，1，1，1，0】

【0，0，1，1，0】



更多如：





（4）实验总结：

①在利用Warshall算法的时候将邻接矩阵看作图的结点集的关系矩阵，由此就可以算出可达矩阵，从而可以反映出个节点间是否有路。

②采用矩阵表示图，既便于计算机储存和处理图的信息，也便于运用代数的方法（比如说线性代数）研究图的性质，例如，可以通过矩阵计算结果判定图的连通性，可达性等问题。