****

项目说明文档

**离散数学课程实验报告**

**——求关系的自反、对称和传递闭包**

**培养单位：软件学院**

**本 科 生：蓝 笙 聆**

**学 号：1951096**

**指导老师：唐 剑 锋**

二○二○年十二月

目录

[一、实验目的 1](#_Toc60012021)

[二、实验内容 1](#_Toc60012022)

[三、实验原理 1](#_Toc60012023)

[1、关系的闭包 1](#_Toc60012024)

[1、自反闭包 1](#_Toc60012025)

[2、对称闭包 1](#_Toc60012026)

[3、传递闭包 2](#_Toc60012027)

[四、实验过程 2](#_Toc60012028)

[1、实验代码 2](#_Toc60012029)

[2、实验截屏 5](#_Toc60012030)

[五、实验小结 6](#_Toc60012031)

[1、解题思路 6](#_Toc60012032)

## 一、实验目的

熟悉Warshall算法，掌握求关系的自反闭包、对称闭包和传递闭包的方法。

## 二、实验内容

给定一个关系矩阵，求出他的自反闭包、对称闭包和传递闭包。

## 三、实验原理

### 1、关系的闭包

设R是集合A上的关系，P是给定的某种性质（如： 自反、对称、传递），满足下列所有条件的关系R1 称为R的关于P的闭包:

R⊆R1

R1 满足性质P

如果存在集合A上的关系R`，R`满足性质P 并包含R，则 R1⊆R`

自反闭包r(R)、对称闭包s(R)、传递闭包t(R)。

### 1、自反闭包

### 2、对称闭包

**对称闭包s(R):**

### 3、传递闭包

**传递闭包t(R):**

## 四、实验过程

### 1、实验代码

根据题意，写出实验代码如下所示：

#include <iostream>

#include <string>

#include <vector>

using namespace std;

void output(vector<vector<int>> arr) {

    cout << "The resulting relationship matrix is:" << endl;

    for (int i = 0; i < arr.size(); i++) {

        for (int j = 0; j < arr[0].size(); j++) cout << arr[i][j] << " ";

        cout << endl;

    }

}

void reflexive(vector<vector<int>> arr) {

    for (int i = 0; i < arr.size(); i++) arr[i][i] = 1;

    output(arr);

}

void symmetric(vector<vector<int>> arr) {

    vector<vector<int>> s1 = arr;

    for (int i = 0; i < arr.size(); i++)

        for (int j = 0; j < arr[0].size(); j++) {

            arr[i][j] = arr[i][j] + s1[j][i];

            if (arr[i][j] > 1) arr[i][j] = 1;

        }

    output(arr);

}

void transitive(vector<vector<int>> arr) {

    //使用warshall算法实现，具体原理见第6题和文档

    for (int i = 0; i < arr.size(); i++)

        for (int j = 0; j < arr.size(); j++)

            if (arr[j][i])

                for (int k = 0; k < arr.size(); k++)

                    arr[j][k] = arr[j][k] | arr[i][k];  //逻辑加

    output(arr);

}

void opening(vector<vector<int>>& arr) {

    int n, d;

    cout << "Reflexive, symmetric, and transitive closures of relationships. "

         << endl;

    cout << "Please enter the number of rows of the matrix." << endl;

    cin >> n;

    arr.resize(n);

    cout << "Please enter the number of columns of the matrix." << endl;

    cin >> d;

    for (auto&& i : arr) i.resize(d);

    cout << "Please enter the relationship matrix" << endl;

    for (int i = 0; i < n; i++) {

        for (int j = 0; j < d; j++) cin >> arr[i][j];

    }

}

bool loop(vector<vector<int>>& arr) {

    cout << "Please enter a number to select an algorithm" << endl

         << "l:Reflexive closure." << endl

         << "2:Symmetric closure." << endl

         << "3:Transitive closure." << endl

         << "4:Quit" << endl;

    string str;

    cin >> str;

    switch (str[0]) {

        case '1':

            reflexive(arr);

            return true;

        case '2':

            symmetric(arr);

            return true;

        case '3':

            transitive(arr);

            return true;

        default:

            return false;

    }

}

int main() {

    vector<vector<int>> arr;

    opening(arr);

    while (loop(arr))

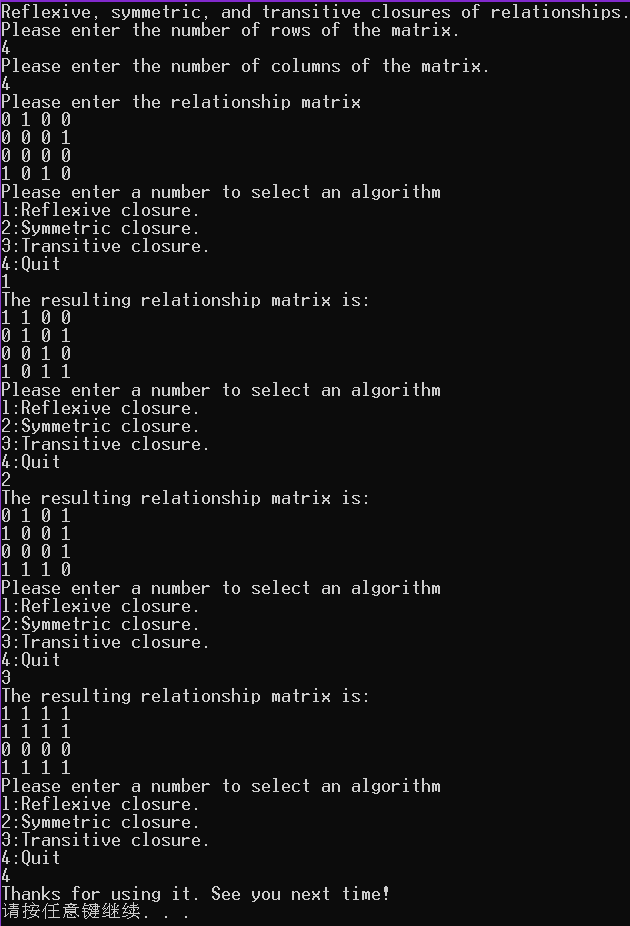
        ;

    std::cout << "Thanks for using it. See you next time! " << std::endl;

    return 0;

}

### 2、实验截屏



## 五、实验小结

### 1、解题思路

**自反闭包**

将关系矩阵的主对角线全部置为1，即为自反闭包。

**对称闭包**

关系矩阵与其转置矩阵进行逻辑或运算，即为对称闭包

**传递闭包**

传递闭包Warshall方法简要介绍：

1. 在集合X上的二元关系R的传递闭包是包含R的X上的最小的传递关系。R的传递闭包在数字图像处理的图像和视觉基础、图的连通性描述等方面都是基本概念。一般用B表示定义在具有n个元素的集合X上关系R的n×n二值矩阵，则传递闭包的矩阵B+可如下计算： B+ = B + B2 + B3 + ……+ (B)n
2. 式中矩阵运算时所有乘法都用逻辑与代替，所有加法都用逻辑或代替。上式中的操作次序为B，B(B)，B(BB)，B(BBB)，……，所以在运算的每一步我们只需简单地把现有结果乘以B。

其具体过程如下，设在n个元素的有限集上关系R的关系矩阵为M：

（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到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。