|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования  ***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н.Э. Баумана)*** |

**ФАКУЛЬТЕТ** ***М-КФ «Машиностроительный»***

**КАФЕДРА** \_***М10-КФ «Высшая математика и физика»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

**ЛАБОРАТОРНАЯ РАБОТА №2**

**«Замыкание отношений»**

**ДИСЦИПЛИНА: «Дискретная математика»**

|  |  |  |
| --- | --- | --- |
| Выполнил: студент гр. ИУК4-32Б | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ( Карельский М.К. )  (Подпись) |
| Проверил: доцент кафедры М10-КФ | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ( Булычев В.А. )  (Подпись) |
| Дата сдачи (защиты):  Результаты сдачи (защиты): | | |
|  | - Балльная оценка:  - Оценка: | |

Калуга, 2021

**Цель:** построение алгоритмов для нахождения матриц рефлексивного, симметричного и транзитивного замыканий.

**Задание:**

1. Прочитайте из файла «data.txt» бинарную матрицу отношения для вашего варианта.
2. Найдите матрицу рефлексивного замыкания.
3. Найдите матрицу симметричного замыкания.
4. Найдите матрицу транзитивного замыкания двумя способами:
   * с помощью умножения и сложения матриц;
   * с помощью алгоритма Уоршалла.
5. В качестве контрольного значения выведите сумму мощностей четырёх отношений: исходного отношения и трёх его замыканий.
6. Нарисуйте ориентированные графы четырёх отношений: исходного отношения и трёх его замыканий. Этот пункт задания можно сделать как с помощью компьютера, так и вручную.

**Вариант 10**

**Листинг:**

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

void main()

{

setlocale(LC\_ALL, "Russian");

int matrix[6][6]{};

bool reflexiveMatrix[6][6]{};

bool symmetricMatrix[6][6]{};

bool transitiveMatrix1[6][6]{};

int transitiveMatrix2[6][6]{};

int temporaryMatrix[6][6]{};

int cardinality = 0;

ifstream fileRead("data.txt");

for (uint16\_t i{}; i < 6; ++i)

{

for (uint16\_t j{}; j < 6; ++j)

{

string line{};

fileRead >> line;

matrix[i][j] = (int)line[0] - 48;

reflexiveMatrix[i][j] = (int)line[0] - 48;

symmetricMatrix[i][j] = (int)line[0] - 48;

transitiveMatrix1[i][j] = (int)line[0] - 48;

transitiveMatrix2[i][j] = (int)line[0] - 48;

temporaryMatrix[i][j] = (int)line[0] - 48;

}

}

fileRead.close();

cout << "Исходное отношение:\n";

for (uint16\_t i{}; i < 6; ++i)

{

for (uint16\_t j{}; j < 6; ++j)

{

cout << " " << (bool)matrix[i][j];

}

cout << "\n";

}

cout << "\n";

reflexiveMatrix[0][0] = true;

reflexiveMatrix[1][1] = true;

reflexiveMatrix[2][2] = true;

reflexiveMatrix[3][3] = true;

reflexiveMatrix[4][4] = true;

reflexiveMatrix[5][5] = true;

cout << "Рефлексивное замыкание:\n";

for (uint16\_t i{}; i < 6; ++i)

{

for (uint16\_t j{}; j < 6; ++j)

{

cout << " " << reflexiveMatrix[i][j];

}

cout << "\n";

}

cout << "\n";

for (uint16\_t i{}; i < 6; ++i)

{

for (uint16\_t j{}; j < 6; ++j)

{

if (matrix[i][j])

{

symmetricMatrix[j][i] = true;

}

}

}

cout << "Симметричное замыкание:\n";

for (uint16\_t i{}; i < 6; ++i)

{

for (uint16\_t j{}; j < 6; ++j)

{

cout << " " << symmetricMatrix[i][j];

}

cout << "\n";

}

cout << "\n";

for (uint16\_t k{}; k < 6; ++k)

{

for (uint16\_t i{}; i < 6; ++i)

{

for (uint16\_t j{}; j < 6; ++j)

{

if (transitiveMatrix1[i][k] && transitiveMatrix1[k][j])

{

transitiveMatrix1[i][j] = true;

}

}

}

}

cout << "Транзитивное замыкание (1-й способ):\n";

for (uint16\_t i{}; i < 6; ++i)

{

for (uint16\_t j{}; j < 6; ++j)

{

cout << " " << transitiveMatrix1[i][j];

}

cout << "\n";

}

cout << "\n";

for (uint16\_t k{}; k < 5; ++k)

{

int temp[6][6]{};

for (uint16\_t i{}; i < 6; ++i)

{

for (uint16\_t j{}; j < 6; ++j)

{

temp[i][j] =

temporaryMatrix[i][0] \* matrix[0][j] +

temporaryMatrix[i][1] \* matrix[1][j] +

temporaryMatrix[i][2] \* matrix[2][j] +

temporaryMatrix[i][3] \* matrix[3][j] +

temporaryMatrix[i][4] \* matrix[4][j] +

temporaryMatrix[i][5] \* matrix[5][j];

}

}

for (uint16\_t i{}; i < 6; ++i)

{

for (uint16\_t j{}; j < 6; ++j)

{

temporaryMatrix[i][j] = temp[i][j];

transitiveMatrix2[i][j] += temp[i][j];

}

}

}

cout << "Транзитивное замыкание (2-й способ):\n";

for (uint16\_t i{}; i < 6; ++i)

{

for (uint16\_t j{}; j < 6; ++j)

{

cout << " " << (bool)transitiveMatrix2[i][j];

}

cout << "\n";

}

cout << "\n";

for (uint16\_t i{}; i < 6; ++i)

{

for (uint16\_t j{}; j < 6; ++j)

{

if (matrix[i][j])

{

++cardinality;

}

if (reflexiveMatrix[i][j])

{

++cardinality;

}

if (symmetricMatrix[i][j])

{

++cardinality;

}

if (transitiveMatrix1[i][j])

{

++cardinality;

}

}

}

cout << "Контрольное значение: " << cardinality << "\n";

}

**Результат:**

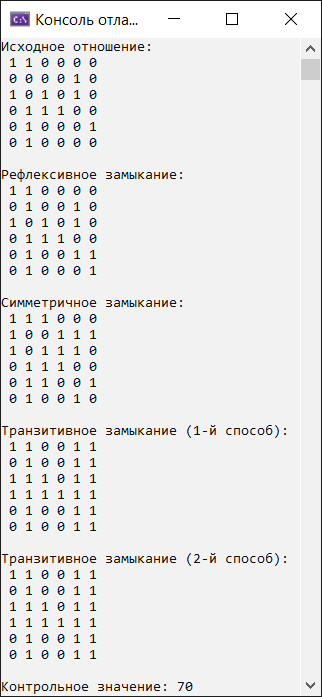


Рис. 1. Результат работы

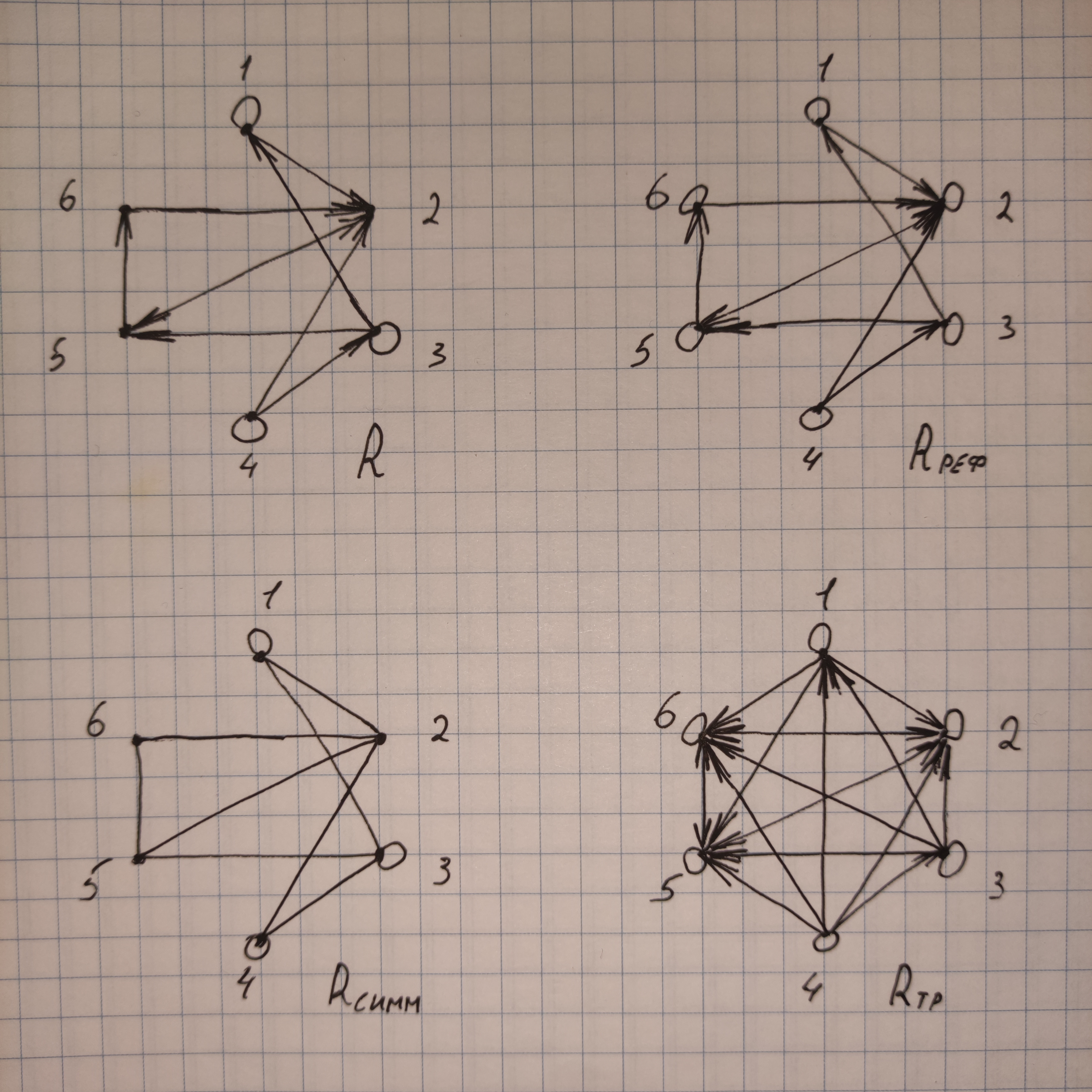


Рис. 2. Ориентированные графы

**Вывод:** в ходе выполнения лабораторной работы составлена программа, вычисляющая матрицы рефлексивного, симметричного и транзитивного замыканий заданного бинарного отношения; нарисованы их графы.

**Контрольные вопросы:**

1. Дайте определения рефлексивного, симметричного, транзитивного отношений.

Отношение называется рефлексивным, если для любого a выполняется aRa.

Отношение называется симметрическим, если для любых a и b из aRb следует bRa.

Отношение называется транзитивным, если для любых a,b и c из aRb и bRс следует aRc.

2. Что такое замыкание отношения по заданному свойству?

Замыканием R' отношения R по свойству *С* называется такое отношение, что R⊆R', R' обладает свойством *С*, а также R' - «наименьшее» из всех отношений, обладающих этими двумя свойствами.

3. Опишите коротко алгоритмы, которые вы использовали для нахождения рефлексивного, симметричного и транзитивного замыканий.

Рефлексивное замыкание – на главной диагонали все элементы становятся единицами.

Симметрическое замыкание – для каждого элемента, для которого выполняется ai j = 1, обратный aj i становится единицей.

Транзитивное замыкание (умножением) – для матрицы находится матрица достижимости: матрица транзитивного замыкания будет равна объединению от 1 до n (количество элементов, в данном случае – 6) степени исходной матрицы R.

Транзитивное замыкание – использован алгоритм Уоршелла:

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

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

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

if (R[i][k] == 1 && R[k][j] == 1) {

R[i][j] = 1

}

}

}

}