**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и

автоматизированныхсистем

Дисциплина: Дискретная математика

Лабораторная работа №3.2

Тема: «**Транзитивное замыкание отношения**»

Выполнил:

студент группы

Проверил: Рязанов Ю. Д.

Белгород 20

**Цель работы**: изучить способы задания отношений, операции над от ношениями и свойства отношений, научиться программно реализовывать операции и определять свойства отношений.

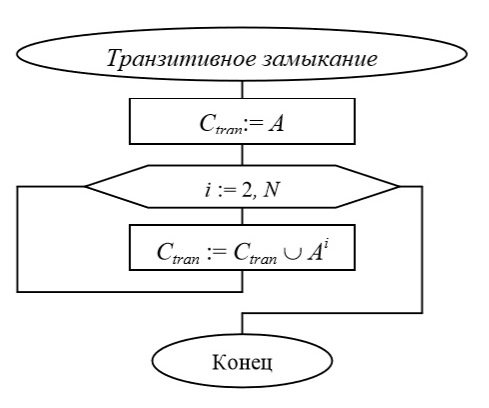
**Задания**

1. Изучить и программно реализовать алгоритмы объединения степеней и Уоршалла для вычисления транзитивного замыкания отношения.

2. Разработать и программно реализовать генератор отношений на множестве мощности N и содержащих заданное число пар.

3. Разработать и написать программу, которая генерирует 1000 отношений на множестве мощности N с заданным числом пар, для каждого отношения вычисляет транзитивное замыкание двумя алгоритмами и определяет время выполнения каждого алгоритма. Время вычисления транзитивного замыкания различных отношений на множестве мощности N с заданным числом пар может быть разным, поэтому программа так же должна определять минимальное и максимальное время вычисления транзитивного замыкания сгенерированных отношений. Выполнить программу при N = 50, 100 и 150. Результат для каждого N представить в виде таблицы.

1.Изучить и программно реализовать алгоритмы объединения степеней и Уоршалла для вычисления транзитивного замыкания отношения.



//объединение

void union\_matr(int \*\*C, int \*\*A, int n)

{

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

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

if(C[x][y] == 0 && A[x][y] == 1)

C[x][y] = 1;

}

int \*\*composition\_matr\_A\_B(int \*\*A, int \*\*B, int n)

{

int \*\*F = NULL;

int i, j, z;

F = (int\*\*) malloc(n \* sizeof(int\*));

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

F[i] = (int\*) malloc(n \* sizeof(int));

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

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

F[i][j] = 0;

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

{

for(z = 0; z < n; z++)

{

if(A[i][z] == 1)

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

{

if(B[z][j] == 1)

F[i][j] = 1;

}

}

}

return F;

}

void transitively\_union(int \*\*C, int \*\*A, int n)

{

int x, y, i = 0;

int \*\*B = NULL;

B = (int\*\*) malloc(n \* sizeof(int\*));

for(y = 0; y < n; y++)

B[y] = (int\*) malloc(n \* sizeof(int));

for(x = 0; x < n; x++)

for(y = 0; y < n; y++)

C[x][y] = A[x][y];

B = composition\_matr\_A\_B(A, A, n);

union\_matr(C, B, n);

for(i = 3; i < n; i++)

{

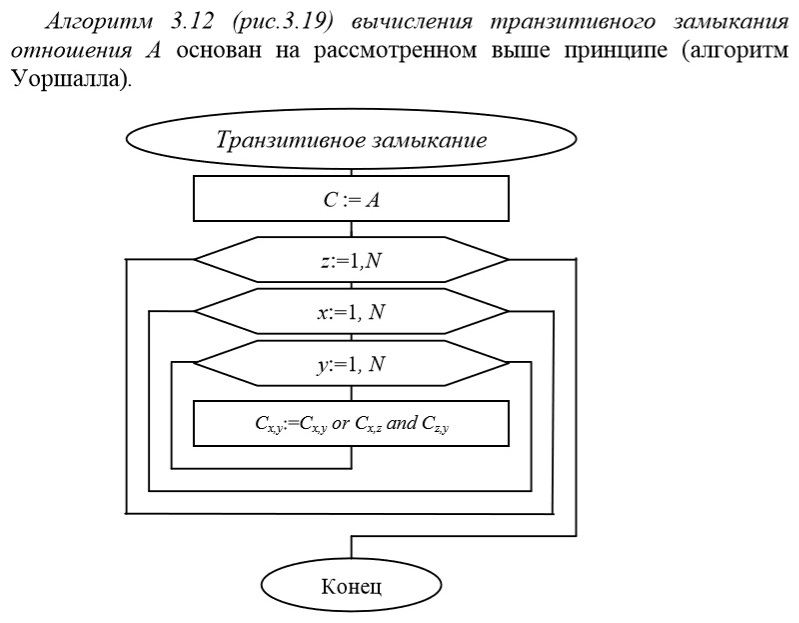
B = composition\_matr\_A\_B(B, A, n);

union\_matr(C, B, n);

}

free\_matr(B, n);

}



void transitively(int \*\*X, int \*\*A, int n)

{

int x, z, y;

for(y = 0; y < n; y++)

for(x = 0; x < n; x++)

X[x][y] = A[x][y];

for(z = 0; z < n; z++)

{

for(x = 0; x < n; x++)

{

for(y = 0; y < n; y++)

{

if((X[x][y] == 1) || ((X[x][z] == 1) && (X[z][y] == 1)))

X[x][y] = 1;

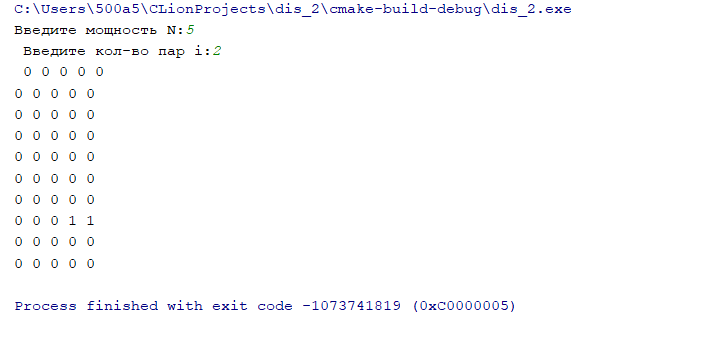
}

}

}

}

2.Разработать и программно реализовать генератор отношений на множестве мощности N и содержащих заданное число пар.



int \*\*generate\_rand\_matr(int \*\*A, int n, int i)

{

int x, y, j = 0;

while (j != i)

{

x = rand() % n;

y = rand() % n;

if(A[x][y] == 0)

{

A[x][y] = 1;

j++;

}

}

return A;

}

3.Разработать и написать программу, которая генерирует 1000 отношений на множестве мощности N с заданным числом пар, для каждого отношения вычисляет транзитивное замыкание двумя алгоритмами и определяет время выполнения каждого алгоритма. Время вычисления транзитивного замыкания различных отношений на множестве мощности N с заданным числом пар может быть разным, поэтому программа так же должна определять минимальное и максимальное время вычисления транзитивного замыкания сгенерированных отношений. Выполнить программу при N = 50, 100 и 150. Результат для каждого N представить в виде таблицы.

int main()

{

setlocale(LC\_ALL, "Rus");

srand(time(NULL));

int \*\*A = NULL;

int \*\*X = NULL;

int n, i;

printf("Введите мощность N: ");

scanf("%d", &n);

getmem\_matr(X, n);

getmem\_matr(A, n);

printf("Введите кол-во пар i: ");

scanf("%d", &i);

clock\_t start;

clock\_t finish;

double min;

double max;

double s;

A = generate\_rand\_matr(A, n, i);

start = clock();

transitively(X, A, n);

//transitively\_union(X, A, n);

finish = clock();

s = finish - start;

//s = double(finish - start) / CLOCKS\_PER\_SEC;

min = s;

max = s;

for(int t = 1; t < 1000; t++)

{

A = generate\_rand\_matr(A, n, i);

start = clock();

transitively(X, A, n);

//transitively\_union(X, A, n);

finish = clock();

s = finish - start;

//s = double(finish - start) / CLOCKS\_PER\_SEC;

if(s > max)

max = s;

if(s < min)

min = s;

}

printf("Min: %f\n", min);

printf("Max: %f\n\n", max);

free\_matr(A, n);

free\_matr(X, n);

return 0;

}

