ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

КАФЕДРА ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

Лабораторная работа по дисциплине

«Л и ОА в ИТ»

на тему

## «Унарные и бинарные операции над графами»

Выполнил студент группы 19ВВ1:

Петухов М. Д.

Принял:

Митрохин М. А.

Пенза 2020

**Теоретическая часть:**

Все унарные операции над графами можно объединить в две группы. Первую группу составляют операции, с помощью которых из исходного графа *G*1*,* можно построить граф *G*2 с меньшим числом элементов. В группу входят операции удаления ребра или вершины, отождествления вершин, стягивание ребра. Вторую группу составляют операции, позволяющие строить графы с большим числом элементов. В группу входят операции расщепления вершин, добавления ребра.

*Отождествление вершин.* В графе *G*1 выделяются вершины *и,v.* Определяют окружение *Q*1 вершины *u*,и окружение *Q*2 вершины *v,* вычисляют их объединение *Q* = *Q1* * Q2.* Затем над графом *G*1 выполняются следующие преобразования:

* из графа *G*1 удаляют вершины *u,* *v (H*1 *= G*1 *- u - v);*
* к графу *Н*1присоединяют новую вершину *z (H*1 *= H*1 *+z);*
* вершину *z* соединяют ребром с каждой из вершин *w*1*Q*

*(G*2 *= H*1 *+ zwi*, *i =* 1,2,3*,…).*

*Стягивание ребра.* Данная операция является операцией отождествления смежных вершин *и, v* в графе *G*1.

Наиболее важными бинарными операциями являются: объединение, пересечение, декартово произведение и кольцевая сумма.

*Объединение.* Граф *G* называется объединением или наложением графов *G*1 и *G*2, если *VG = V*1*V*2*; UG = U*1* U*2 (рис. 1).

**U**

*v*1

*v*2

*v*3

*v*4

*v*3

*v*4

*v*5

*v*2

*v*1

*v*3

*v*4

*v*5

Рис. 1. Объединение графов *G*1, *G*2

Объединение графов *G*1 и *G*2 называется дизъюнктным, если *V*1*V*2 *= *. При дизъюнктном объединении никакие два из объединяемых графов не должны иметь общих вершин.

*Пересечение.* Граф *G* называется пересечением графов *G*1, *G*2,если *VG = V*1*V*2и *UG = U*1*U*2 (риc.2). Операция "пересечения" записывается следующим образом: *G = G*1*G*2*.*

**∩**

*v*1

*v*2

*v*3

*v*5

*v*3

*v*4

*v*6

*v*2

*v*1

*v*6

*v*4

*v*5

*v*1

*v*4

*v*6

*v*5

*v*3

*v*2

Рис.2. Пересечение графов *G*1, *G*2*.*

*Декартово произведение.* Граф *G* называется декартовым произведением графов *G*1 и *G*2 если *VG* = *V*1*V*2 —декартово произведение множеств вершин графов *G*1, *G*2, а множество ребер *U*c задается следующим образом: вершины (*zi*, *vk*) и (*zj*, *vl*) смежны в графе *G* тогда и только тогда, когда *zi* = *zj*(*i* = *j*), a *v*k и *vl* смежны в *G*2 или *vk* = *vl*(*k* = *l*), смежны в графе *G*1 (см. рис.3).

**X**

*z*1

*z*2

*v*1

*v*3

*v*2

*z*1*v*1

*z*1*v*2

*z*1*v*3

*z*2*v*1

*z*2*v*2

*z*2*v*3

Рис. 3. Декартово произведение графов *G*1, *G*2

*Кольцевая сумма* графов представляет граф, который не имеет изолированных вершин и состоит из ребер, присутствующих либо в первом исходном графе, либо во втором. Кольцевая сумма определяется следующим соотношением: *G* = *G*1  *G*2 (рис.4).

**⊕**

*v*1

*v*2

*v*3

*v*5

*v*3

*v*4

*v*6

*v*2

*v*1

*v*4

*v*5

*v*1

*v*4

*v*6

*v*5

*v*3

*v*2

Рис.4. Кольцевая сумма графов *G1, G2*

**Лабораторное задание:**

**Задание 1:**

1. Сгенерируйте (используя генератор случайных чисел) две матрицы *M*1*, М*2 смежности неориентированных помеченных графов *G*1, *G*2. Выведите сгенерированные матрицы на экран.

**Задание 2:**

1. Для матричной формы представления графов выполните операцию:

а) отождествления вершин

б) стягивания ребра

в) расщепления вершины

Номера выбираемых для выполнения операции вершин ввести с клавиатуры.

Результат выполнения операции выведите на экран.

**Задание 3:**

1. Для матричной формы представления графов выполните операцию:

а) объединения *G* = *G*1  *G*2

б) пересечения *G* = *G*1  *G*2

в) кольцевой суммы *G* = *G*1  *G*2

Результат выполнения операции выведите на экран.

**Практическая часть:**

**Листинг:**

#include "stdafx.h"

#include <conio.h>

#include <stdlib.h>

int main()

{

int M1[6][6], M2[6][6], Z[7][7];

int i,j,u,v,k1=0,k2=0;

for (i=0;i<6;i++)//Задание 1

{

for (j=i;j<6;j++)

{

M1[i][j]=rand()%2;

M1[j][i]=M1[i][j];

M2[i][j]=rand()%2;

M2[j][i]=M2[i][j];

}

M1[i][i]=0;

M2[i][i]=0;

}

for (i=0;i<6;i++)

{

for (j=0;j<6;j++)

printf("%2d",M1[i][j]);

printf("\n");

}

printf("\n");

for (i=0;i<6;i++)

{

for (j=0;j<6;j++)

printf("%2d",M2[i][j]);

printf("\n");

}

printf("\n");//Задание 1

scanf("%d %d",&u,&v);//Задание 2

for (i=0;i<6;i++)

{

if (i==u || i==v) {i++;k1++;}

for (j=0;j<6;j++)

{

if (j==u || j==v) {j++;k2++;}

Z[i-k1][j-k2]=M1[i][j];

}

k2=0;

}

k1=0; k2=0;

for (i=0;i<6;i++)

{

if (i==v){i++;k1++;}

if (M1[i][u]==1 || M1[i][v]==1) Z[i-k1][4]=1;

else Z[i-k1][4]=0;

Z[4][i-k1]=Z[i-k1][4];

}

k1=0;

for (i=0;i<5;i++)

{

for (j=0;j<5;j++)

{

printf("%2d",Z[i][j]);

Z[i][j]=0;

}

printf("\n");

}

printf("\n");

for (i=0;i<6;i++)

{

if (i==u) {i++;k1++;}

for (j=0;j<6;j++)

{

if (j==u) {j++;k2++;}

Z[i-k1][j-k2]=M1[i][j];

}

k2=0;

}

k1=0;

for (i=0;i<7;i++)

{

if (i==u){i++;k1++;}

if (M1[i][u]==1){Z[i-k1][5]=1;k2++;}

else Z[i-k1][5]=0;

if (k2 % 2==1) Z[i-k1][5]=0;

Z[5][i-k1]=Z[i-k1][5];

}

k1=0;

for (i=0;i<8;i++)

{

if (i==u){i++;k1++;}

if (M1[i][u]==1) {Z[i-k1][6]=1;k2++;}

else Z[i-k1][6]=0;

if (k2 % 2==0) Z[i-k1][6]=0;

Z[6][i-k1]=Z[i-k1][6];

}

Z[5][6]=1;

Z[6][5]=1;

for (i=0;i<7;i++)

{

for (j=0;j<7;j++)

{

printf("%2d",Z[i][j]);

Z[i][j]=0;

}

printf("\n");

}

printf("\n");//Задание 2

for (i=0;i<6;i++)//Задание 3

{

for (j=0;j<6;j++)

{

Z[i][j]=M1[i][j];

if (M2[i][j]==1) Z[i][j]=1;

printf("%2d",Z[i][j]);

Z[i][j]=0;

}

printf("\n");

}

printf("\n");

for (i=0;i<6;i++)

{

for (j=0;j<6;j++)

{

if (M1[i][j]==1 && M2[i][j]==1) Z[i][j]=1;

printf("%2d",Z[i][j]);

Z[i][j]=0;

}

printf("\n");

}

printf("\n");

for (i=0;i<6;i++)

{

for (j=0;j<6;j++)

{

if (i!=j && (i==j ||((M1[i][j]==1 & M2[i][j]==0)||(M1[i][j]==0 & M2[i][j]==1)))) Z[i][j]=1;

printf("%2d",Z[i][j]);

Z[i][j]=0;

}

printf("\n");

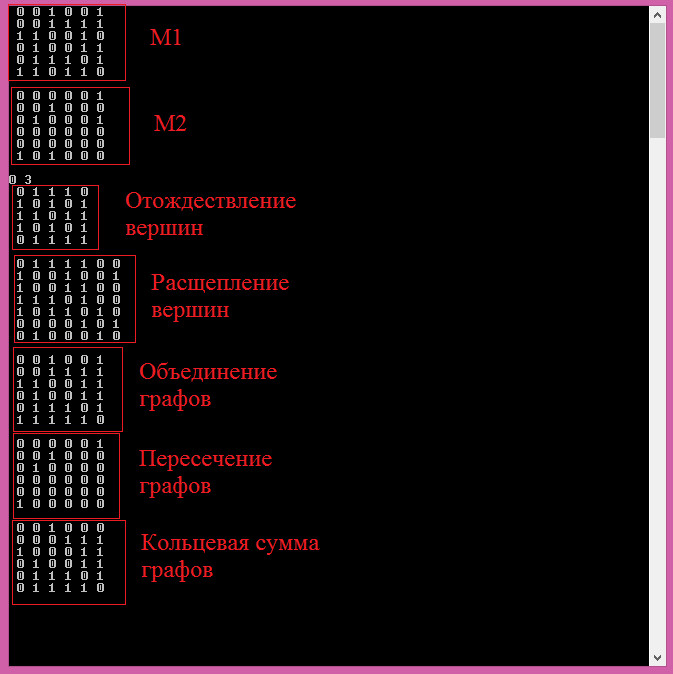
}//Задание 3

getch();

return 0;

}

**Задания:**

****

Вывод: Я научился производить унарные и бинарные операции над графами.