Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №3

# по курсу «Логика и основы алгоритмизации в инженерных задачах»

## на тему «Унарные и бинарные операции над графами»

**Выполнил:**

студент группы 20ВВ3

Халилов Алмаз

**Приняли:**

Юрова О.В.

Митрохин М.А.

Пенза 2021

# **Название -** Унарные и бинарные операции над графами

**Цель работы** – Изучение унарных и бинарных операций. Реализация алгоритмов этих операций.

**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.
2. \* Для указанных графов преобразуйте представление матриц смежности в списки смежности. Выведите полученные списки 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.

1. \* Для представления графов в виде списков смежности 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.

**Задание 4 \***

1. Для матричной формы представления графов выполните операцию декартова произведения графов *G = G*1X *G*2.

Результат выполнения операции выведите на экран.

**Ход работы**

**Листинг**

#define \_CRT\_NONSTDC\_NO\_WARNINGS

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <queue>

#include <stdio.h>

#include <locale.h>

#include <stdlib.h>

#include <time.h>

#include <stack>

void matrix(int\*\* g1, int\*\* g2, int size)

{

//1мат

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

{

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

{

g1[i][j] = rand() % 2;

}

}

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

{

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

{

if (i == j)

g1[i][j] = 0;

else

g1[i][j] = g1[j][i];

}

}

//2мат

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

{

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

{

g2[i][j] = rand() % 2;

}

}

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

{

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

{

if (i == j)

g2[i][j] = 0;

else

g2[i][j] = g2[j][i];

}

}

}

void print(int\*\* g1, int\*\* g2, int size) {

printf("\nМатрица 1: \n");

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

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

printf("% d ", g1[i][j]);

}

printf("\n");

}

printf("\nМатрица 2: \n");

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

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

printf("% d ", g2[i][j]);

}

printf("\n");

}

printf("\n");

}

int\*\* toj(int\*\* g, int size)

{

int n1, n2;

int a = 0;

int\*\* num = (int\*\*)malloc(size \* sizeof(int\*));

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

{

num[i] = (int\*)malloc(size \* sizeof(int));

}

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

{

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

num[i][j] = g[i][j];

}

printf("Вершина 1:");

scanf("%d", &n1);

printf("Вершина 2:");

scanf("%d", &n2);

if (num[n1][n2] == 1)

a = 1;

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

{

if (num[n2][i] == 1)

{

num[n1][i] = num[n2][i];

num[i][n1] = num[i][n2];

}

}

for (int i = n2; i < size - 1; i++)

{

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

num[i][j] = num[i + 1][j];

}

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

{

for (int j = n2; j < size - 1; j++)

num[i][j] = num[i][j + 1];

}

size--;

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

num[i] = (int\*)realloc(num[i], size \* sizeof(int));

free(num[size]);

if (a == 1)

num[n1][n1] = 1;

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

{

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

{

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

}

printf("\n");

}

return(num);

}

int\*\* styag(int\*\* g, int size)

{

int n1, n2;

int\*\* num = (int\*\*)malloc(size \* sizeof(int\*));

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

num[i] = (int\*)malloc(size \* sizeof(int));

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

{

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

num[i][j] = g[i][j];

}

printf("Вершина 1:");

scanf("%d", &n1);

printf("Вершина 2:");

scanf("%d", &n2);

if (num[n1][n2] != 1) {

return(num);

}

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

{

if (num[n2][i] == 1)

{

num[n1][i] = num[n2][i];

num[i][n1] = num[i][n2];

}

}

for (int i = n2; i < size - 1; i++)

{

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

num[i][j] = num[i + 1][j];

}

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

{

for (int j = n2; j < size - 1; j++)

num[i][j] = num[i][j + 1];

}

size--;

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

num[i] = (int\*)realloc(num[i], size \* sizeof(int));

free(num[size]);

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

{

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

{

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

}

printf("\n");

}

return(num);

}

int\*\* ras(int\*\* g, int size)

{

int n;

int\*\* num = (int\*\*)calloc((size + 1), sizeof(int\*));

for (int i = 0; i < (size + 1); i++)

num[i] = (int\*)calloc((size + 1), sizeof(int));

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

{

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

num[i][j] = g[i][j];

}

printf("Вершины: ");

scanf("%d", &n);

for (int j = 0; j < size / 2; j++)

{

num[j][size] = num[n][j];

num[size][j] = num[j][n];

num[n][j] = 0;

num[j][n] = 0;

}

for (int i = 0; i < (size + 1); i++)

{

for (int j = 0; j < (size + 1); j++)

{

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

}

printf("\n");

}

return(num);

}

int\*\* obed(int\*\* g1, int\*\* g2, int size)

{

int\*\* num = (int\*\*)malloc(size \* sizeof(int\*));

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

num[i] = (int\*)malloc(size \* sizeof(int));

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

{

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

{

num[i][j] = g1[i][j];

}

}

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

{

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

{

if (g2[i][j] == 1)

num[i][j] = g2[i][j];

}

}

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

{

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

{

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

}

printf("\n");

}

return(num);

}

int\*\* peresech(int\*\* g1, int\*\* g2, int size)

{

int\*\* num = (int\*\*)calloc(size, sizeof(int\*));

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

num[i] = (int\*)calloc(size, sizeof(int));

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

{

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

{

if ((g1[i][j] == 1) && (g2[i][j] == 1))

num[i][j] = g1[i][j];

}

}

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

{

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

{

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

}

printf("\n");

}

return(num);

}

int\*\* sum(int\*\* g1, int\*\* g2, int size)

{

int\*\* num = (int\*\*)calloc(size, sizeof(int\*));

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

num[i] = (int\*)calloc(size, sizeof(int));

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

{

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

{

if ((g1[i][j] == 1) && (g2[i][j] == !1))

num[i][j] = g1[i][j];

else

if ((g2[i][j] == 1) && (g1[i][j] == !1))

num[i][j] = g2[i][j];

}

}

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

{

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

{

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

}

printf("\n");

}

return(num);

}

int main()

{

int size,i;

int\*\* g2 = 0;

int\*\* g1 = 0;

setlocale(LC\_ALL, "RUS");

srand(time(NULL));

printf("Размер матриц: ");

scanf("%d", &size);

g1 = (int\*\*)malloc(size \* sizeof(int\*));

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

{

g1[i] = (int\*)malloc(size \* sizeof(int));

};

g2 = (int\*\*)malloc(size \* sizeof(int\*));

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

{

g2[i] = (int\*)malloc(size \* sizeof(int));

};

matrix(g1, g2, size);

print(g1, g2, size);

printf("Отождествление матрица 1: \n");

toj(g1, size);

printf("Отождествление матрица 2: \n");

toj(g2, size);

print(g1, g2, size);

printf("Стягивание матрица 1: \n");

styag(g1, size);

printf("Стягивание матрица 2: \n");

styag(g2, size);

print(g1, g2, size);

printf("Расщепление матрица 1: \n");

ras(g1, size);

printf("Расщипление матрица 2: \n");

ras(g2, size);

print(g1, g2, size);

printf("Объединение матриц: \n");

obed(g1, g2, size);

printf("Пересечение матриц: \n");

peresech(g1, g2, size);

printf("Кольцевая сумма матриц: \n");

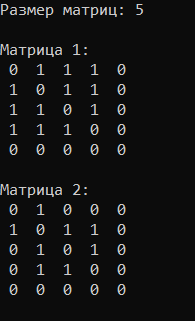
sum(g1, g2, size);

system("PAUSE");

}

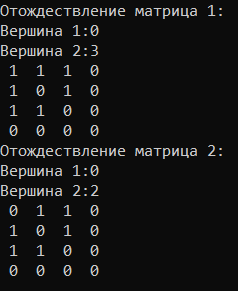
**Задание 1:**

1. Сгенерировали матрицы

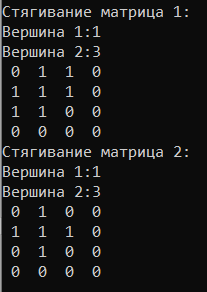


**Задание 2:**

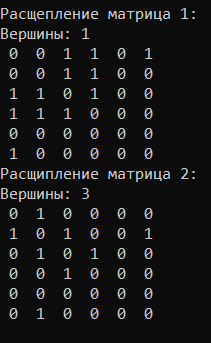
1. Отождествление матриц



1. Стягивание ребра

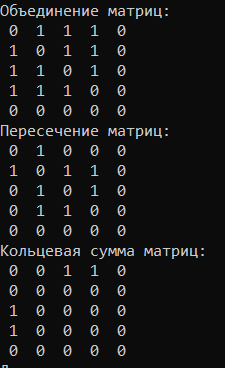


1. Расщепление вершины



**Задание 3:**

1. Объединение, пересечения, кольцевая сумма.



**Вывод:** Изучили унарные и бинарные операции над графами, реализовали алгоритмы унарных и бинарных операций на языке Си.