Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №7

по курсу «Логика и основы алгоритмизации в ИС»

на тему «Поиск расстояний во взвешенном графе»

Выполнили

студенты группы 20ВВ2:

Тумасов Вадим

Портнов Никита

Принял:

д.т.н Митрохин М.А.

д.т.н Юрова О.В.

Пенза 2021

**Лабораторное задание:**

Задание 1

1. Сгенерируйте (используя генератор случайных чисел) матрицу

смежности для неориентированного взвешенного графа G. Выведите матрицу

на экран.

2. Для сгенерированного графа осуществите процедуру поиска

расстояний, реализованную в соответствии с приведенным выше описанием.

При реализации алгоритма в качестве очереди используйте класс queue из

стандартной библиотеки С++.

3.\* Сгенерируйте (используя генератор случайных чисел) матрицу

смежности для ориентированного взвешенного графа G. Выведите матрицу

на экран и осуществите процедуру поиска расстояний, реализованную в

соответствии с приведенным выше описанием.

Задание 2\*

1. Модернизируйте программу так, чтобы получить возможность запуска

программы с параметрами командной строки (см. описание ниже). В

качестве параметра должны указываться тип графа (взвешенный или нет) и

наличие ориентации его ребер (есть ориентация или нет).

**Листинг:**

#include <iostream>

#include <stdio.h>

#include <stdlib.h>

#include <cstdlib>

#include <windows.h>

#include <conio.h>

#include <ctime>

#include <cmath>

#include <stack>

#include <queue>

using namespace std;

int max\_size = 7, i, j, start, num;

int\* DIST = (int\*)malloc(max\_size \* sizeof(int));

void BSFD(int st, int\*\* b, int num\_of\_elem, int\* vis) {

queue<int> Q;

int t;

Q.push(st);

vis[st] = 0;

while (!Q.empty())

{

t = Q.front();

Q.pop();

printf("%d ", t + 1);

for (int i = 0; i < num\_of\_elem; i++)

{

if (b[t][i] > 0 && vis[i] == 1000)

{

vis[i] = vis[t] + b[t][i];

Q.push(i);

}

}

}

}

void no\_orient\_graph()

{

int\*\* a = new int\* [max\_size];

for (i = 0; i < max\_size; i++)

{

a[i] = new int[max\_size];

}

for (i = 0; i < max\_size; i++)

{

for (j = i + 1; j < max\_size; j++)

{

num = rand() % 7;

if (num <= 3)

{

a[i][j] = 0;

a[j][i] = a[i][j];

}

else

{

a[i][j] = (rand() % 6) + 1;

a[j][i] = a[i][j];

}

}

}

printf("\nНеориентированный взвешенный:\n");

for (i = 0; i < max\_size; i++)

{

DIST[i] = 1000;

for (j = 0; j < max\_size; j++)

{

if (i == j)

a[i][j] = 0;

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

}

printf("\n");

}

printf("\nВведите стартовую вершину - ");

scanf\_s("%d", &start);

BSFD(start - 1, a, max\_size, DIST);

printf("\nПоиск расстояний:\n");

for (i = 0; i < max\_size; i++) {

printf("До %d = %d\n", i + 1, DIST[i]);

DIST[i] = 1000;

}

\_getch();

}

void no\_orient\_graph\_NW()

{

int\*\* a = new int\* [max\_size];

for (i = 0; i < max\_size; i++)

{

a[i] = new int[max\_size];

}

for (i = 0; i < max\_size; i++)

{

for (j = i + 1; j < max\_size; j++)

{

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

a[j][i] = a[i][j];

}

}

printf("\nНеориентированный невзвешенный :\n");

for (i = 0; i < max\_size; i++)

{

DIST[i] = 1000;

for (j = 0; j < max\_size; j++)

{

if (i == j)

a[i][j] = 0;

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

}

printf("\n");

}

printf("\nВведите стартовую вершину - ");

scanf\_s("%d", &start);

BSFD(start - 1, a, max\_size, DIST);

printf("\nПоиск расстояний:\n");

for (i = 0; i < max\_size; i++) {

printf("До %d = %d\n", i + 1, DIST[i]);

DIST[i] = 1000;

}

\_getch();

}

void orient\_graph() {

int\*\* a = new int\* [max\_size];

for (i = 0; i < max\_size; i++)

{

a[i] = new int[max\_size];

}

for (i = 0; i < max\_size; i++)

{

for (j = 0; j < max\_size; j++)

{

num = rand() % 7;

if (num <= 3)

{

a[i][j] = 0;

}

else

{

a[i][j] = (rand() % 6) + 1;

}

}

}

printf("\n Ориентированный взвешенный:\n");

for (i = 0; i < max\_size; i++)

{

DIST[i] = 1000;

for (j = 0; j < max\_size; j++)

{

if (i == j)

a[i][j] = 0;

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

}

printf("\n");

}

printf("\nВведите стартовую вершину - ");

scanf\_s("%d", &start);

BSFD(start - 1, a, max\_size, DIST);

printf("\nПоиск расстояний:\n");

for (i = 0; i < max\_size; i++) {

printf("До %d = %d\n", i + 1, DIST[i]);

DIST[i] = 1000;

}

\_getch();

}

void orient\_grap\_NW() {

int\*\* a = new int\* [max\_size];

for (i = 0; i < max\_size; i++)

{

a[i] = new int[max\_size];

}

for (i = 0; i < max\_size; i++)

{

for (j = 0; j < max\_size; j++)

{

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

}

}

printf("\n Ориентированный невзвешенный:\n");

for (i = 0; i < max\_size; i++)

{

DIST[i] = 1000;

for (j = 0; j < max\_size; j++)

{

if (i == j)

a[i][j] = 0;

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

}

printf("\n");

}

printf("\nВведите стартовую вершину - ");

scanf\_s("%d", &start);

BSFD(start - 1, a, max\_size, DIST);

printf("\nПоиск расстояний:\n");

for (i = 0; i < max\_size; i++) {

printf("До %d = %d\n", i + 1, DIST[i]);

DIST[i] = 1000;

}

\_getch();

}

int main(int argc, char\* argv[]) {

setlocale(LC\_ALL, "Rus");

srand(time(NULL));

int counter\_argc = 1;

if (argc > 1)

{

while (counter\_argc <= argc)

{

if (strcmp(argv[counter\_argc-1], "NW") == 0)

no\_orient\_graph();

else if (strcmp(argv[counter\_argc-1], "OW") == 0)

orient\_graph();

else if (strcmp(argv[counter\_argc-1], "NN") == 0)

no\_orient\_graph\_NW();

else if (strcmp(argv[counter\_argc-1], "ON") == 0)

orient\_grap\_NW();

counter\_argc++;

}

}

else

{

no\_orient\_graph();

orient\_graph();

no\_orient\_graph\_NW();

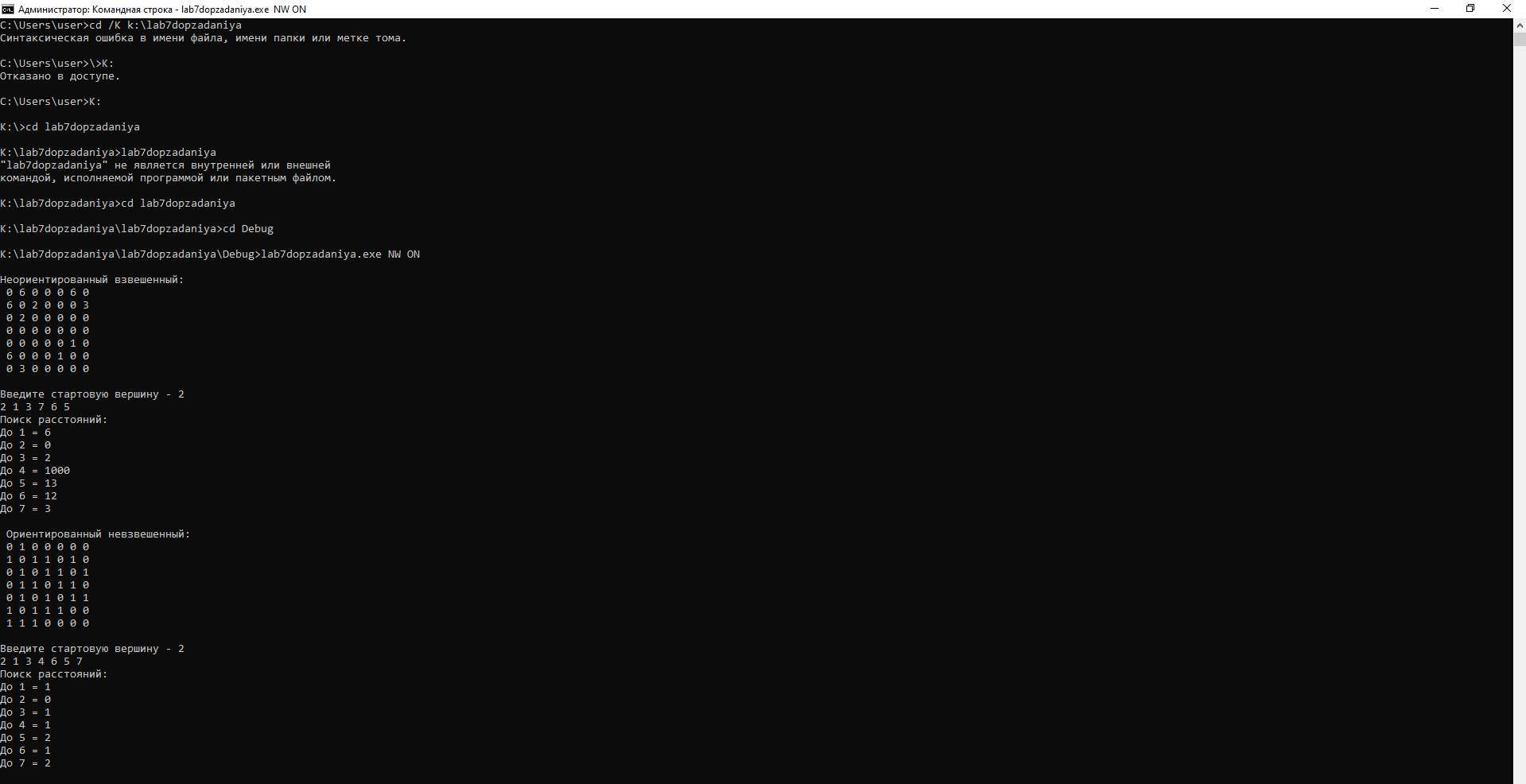
orient\_grap\_NW();

}

}

**Результат работы программы:**

С помощью командной строки запустили exe-файл нашей лабораторной работы и ввели два аргумента,проверили работоспособность нашей программы в таких условиях.



**Вывод**:в ходе выполнения данной лабораторной работы мы научились работать с ориентированным графом(взвешенным и невзвешенным),так же изучили команды командной строки чтобы перемещаться по дискам и папкам,чтобы запускать программу с помощью нее.