Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №7

по курсу «Логика и основы алгоритмизации в инженерных задачах»

## на тему «Поиск расстояний во взвешенном графе»

Выполнил:

студент группы 19ВВ2

Юдин Д.А.

Приняли:

Митрохин М. А.

Юрова О.В.

Пенза 2020

Цель работы

Реализовать алгоритм поиска расстояний во взвешенном графе.

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.

### **Листинг**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <iostream>

#include <stdlib.h>

#include <queue>

#include <time.h>

using namespace std;

void bfsd(int\*\* mas1, int n, int\* dist, int v)

{

queue<int> q;

q.push(v);

dist[v] = 0;

while (!q.empty())

{

v = q.front();

q.pop();

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

{

if (mas1[v][i] > 0 && dist[i] > dist[v] + mas1[v][i])

{

q.push(i);

dist[i] = dist[v] + mas1[v][i];

}

}

}

}

void main()

{

setlocale(LC\_ALL, "RUS");

srand(time(NULL));

int n, num, v; //N-кол-во вершин, num-вершина для ввода, v-вершина для ввода

int\*\* mas1, \*\*mas2;

printf("\nУкажите размер матрицы N\*N: ");

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

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

mas1 = (int\*\*)malloc(n \* sizeof(int\*));

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

{

mas1[i] = (int\*)malloc(n \* sizeof(int));

}

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

{

mas1[i][i] = 0;

for (int j = i + 1; j < n; j++)

{

mas1[i][j] = rand() % 11;

mas1[j][i] = mas1[i][j];

}

}

printf("\n\nВывод матрицы смежности:\n ");

for (int i = 0; i < n; i++) { // вывод матриц смежности

printf("\n");

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

printf("%d ", mas1[i][j]);

}

}

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

{

dist[i] = 1000;

}

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

bfsd(mas1, n, dist, i);

printf("\nРасстояние от вершины %d до остальных:\n", i + 1);

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

{

printf("%d ", dist[j]);

}

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

{

dist[k] = 1000;

}

}

mas2 = (int\*\*)malloc(n \* sizeof(int\*));

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

{

mas2[i] = (int\*)malloc(n \* sizeof(int));

}

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

{

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

{

mas2[i][j] = rand() % 11;

}

mas2[i][i] = 0;

}

printf("\n\nВывод матрицы смежности ориентированного графа:\n ");

for (int i = 0; i < n; i++) { // вывод матриц смежности

printf("\n");

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

printf("%d ", mas2[i][j]);

}

}

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

{

dist[i] = 1000;

}

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

bfsd(mas2, n, dist, i);

printf("\nРасстояние от вершины %d до остальных:\n", i + 1);

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

{

printf("%d ", dist[j]);

}

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

{

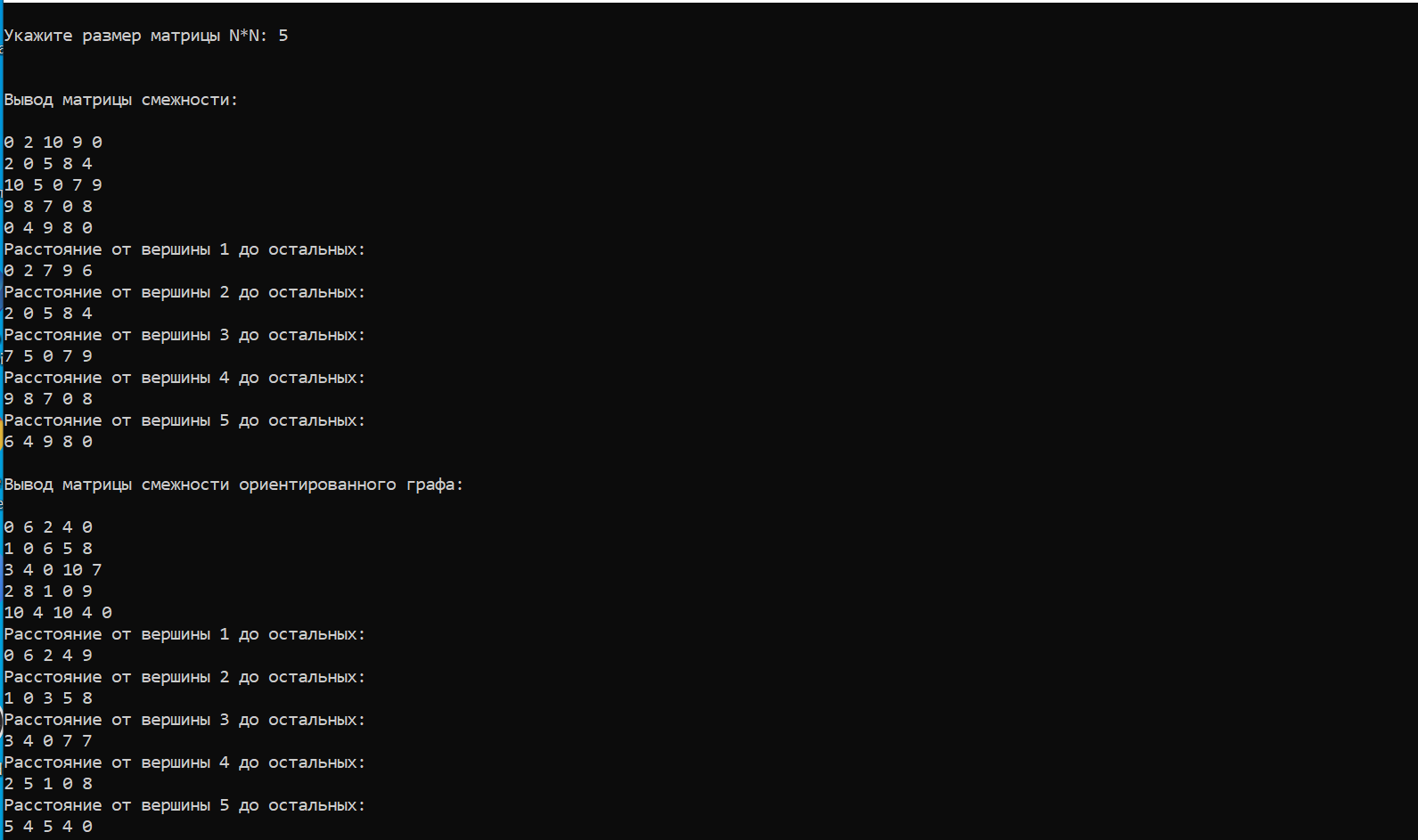
dist[k] = 1000;

}

}

}

**Результаты работы программы**



## Вывод

В ходе выполнения данной лабораторной работы мы научились работать с алгоритмом поиска расстояний во взвешенном графе. Научились реализовывать его с помощью стандартной библиотеки <queue>.