Министерство 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»

Выполнили:

студент группы 20ВВ2

Ермолаев А.Д.

Мартынов Н.В.

Принял:

д.т.н., профессор

Митрохин М.А.

к.т.н., доцент

Юрова О.В.

**Пенза 2021**

**Цель работы:**  составить программу, которая будет осуществлять поиск расстояния от выбранной вершины до всех остальных во взвешенном графе.

**Лабораторные задания:**

**Псевдокод:**

Вход: G – матрица смежности графа, v – исходная вершина.

Выход: DIST – вектор расстояний до всех вершин от исходной.

Алгоритм ПОШ

1.1. для всех i положим DIST [i] = 1000 пометим как “не посещенную”;

1.2. ВЫПОЛНЯТЬ BFSD (v).

1.3 для всех i вывести DIST [i] на экран;

Алгоритм BFSD(v):

2.1. Создать пустую очередь Q = {};

2.2. Поместить v в очередь Q.push(v);

2.3. Обновить вектор расстояний DIST [ x ] = 0;

2.4. ПОКА Q != ∅ очередь не пуста ВЫПОЛНЯТЬ

2.5. v = Q.front() установить текущую вершину;

2.6. Удалить первый элемент из очереди Q.pop();

2.7. вывести на экран v;

2.8. ДЛЯ i = 1 ДО size\_G ВЫПОЛНЯТЬ

2.9. ЕСЛИ G(v,i) > 0 И DIST [ i ] > DIST [ v ] + G(v,i)

2.10. ТО

2.11. Поместить i в очередь Q.push(i);

2.12. Обновить вектор расстояний DIST [ i ] = DIST [ v ] +G(v,i);

**Листинг:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <queue>

#include <iostream>

#include <stdio.h>

#include <time.h>

#include <stdlib.h>

#include <locale.h>

using namespace std;

#define MAXINT 32767

queue <int> q;

void BFS(int v, int\*\* M, int\* Dist, int Size)

{

int i;

q.push(v); Dist[v] = 0;

while (!q.empty())

{

v = q.front();

q.pop();

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

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

{

if ((Dist[i] > Dist[v] + M[v][i]) && (M[v][i] > 0))

{

q.push(i);

Dist[i] = Dist[v] + M[v][i];

}

}

}

}

void main()

{

int\*\* G = NULL, i = 0, j = 0, S = 0, \* Distance = NULL, N, Ves, O;

setlocale(LC\_ALL, "Rus");

srand(time(NULL));

printf("Введите размерность матрицы смежности: "); scanf("%d", &S);

printf("Введите:\n1-граф взвешенный, 2-граф невзвешенный: "); scanf("%d", &Ves);

printf("Введите:\n1-граф неориентированный, 2-граф ориентированный: "); scanf("%d", &O);

while (S <= 0 || (Ves < 1 || Ves>2) || (O < 1 || O>2))

{

printf("Неправильно введённые параметры! Введите положительные числа (или те, что Вам предложены):\n");

scanf("%d%d%d", &S, &Ves, &O);

}

G = (int\*\*)malloc(S \* sizeof(int));

Distance = (int\*)malloc(S \* sizeof(int));

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

Distance[i] = MAXINT;

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

{

G[i] = (int\*)malloc(S \* sizeof(int));

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

{

if (Ves == 1)

G[i][j] = rand() % 10;

else

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

G[i][i] = 0;

}

}

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

{

printf("\n%4d|", i + 1);

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

{

if (O == 1)

G[j][i] = G[i][j];

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

}

}

printf("\n\nС какой вершины начать обход в ширину? "); scanf("%d", &N);

while (N<1 || N>S)

{

printf("Некорректно введённое число! Введите целое положительное число в пределах размера матрицы смежности: ");

scanf("%d", &N);

}

printf("Порядок обхода графа в ширину (возможны повторяющиеся вершины в связи с обновлением расстояний):\n");

BFS(N - 1, G, Distance, S);

printf("\nВектор расстояний от выбранной вершины до всех остальных: ");

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

printf("%d ", Distance[i]);

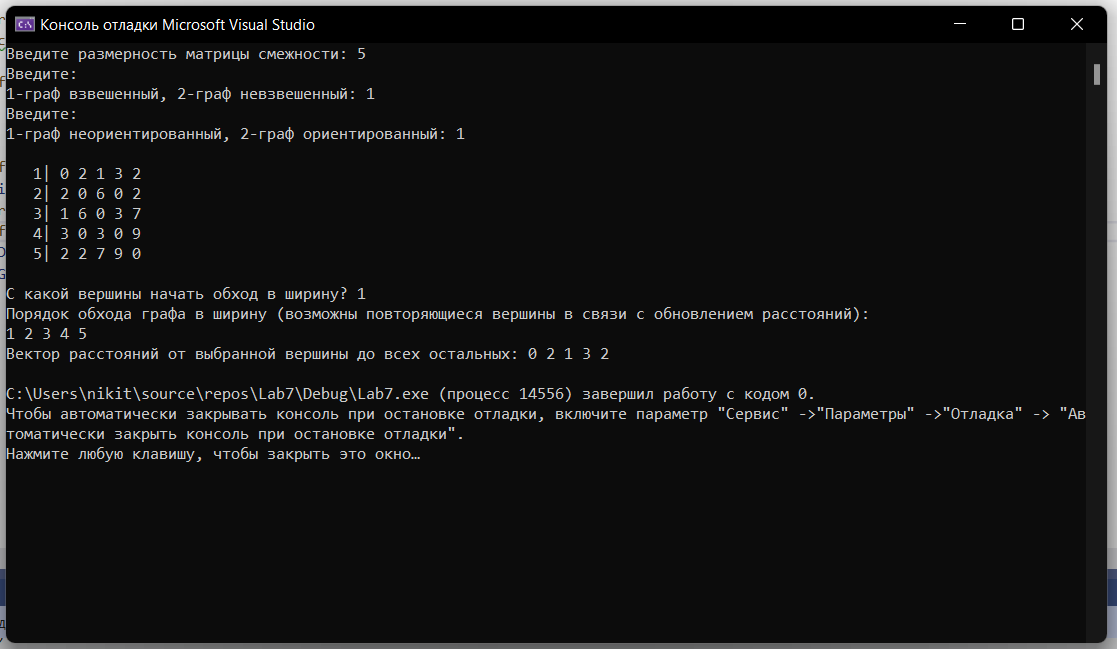
printf("\n");

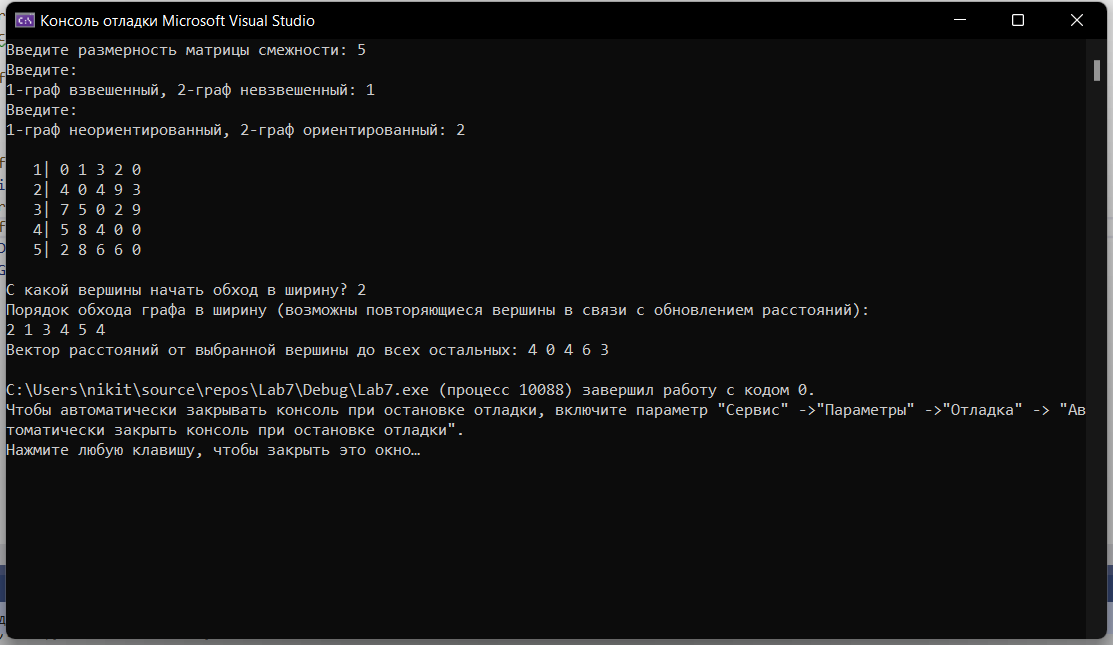
free(Distance);

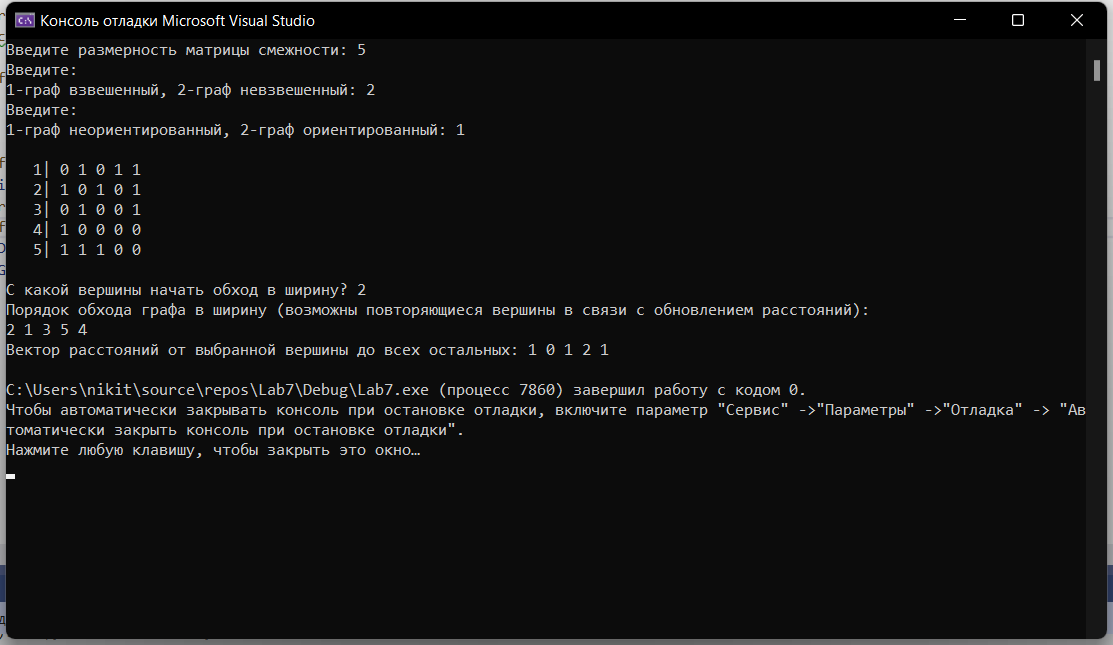
free(G);

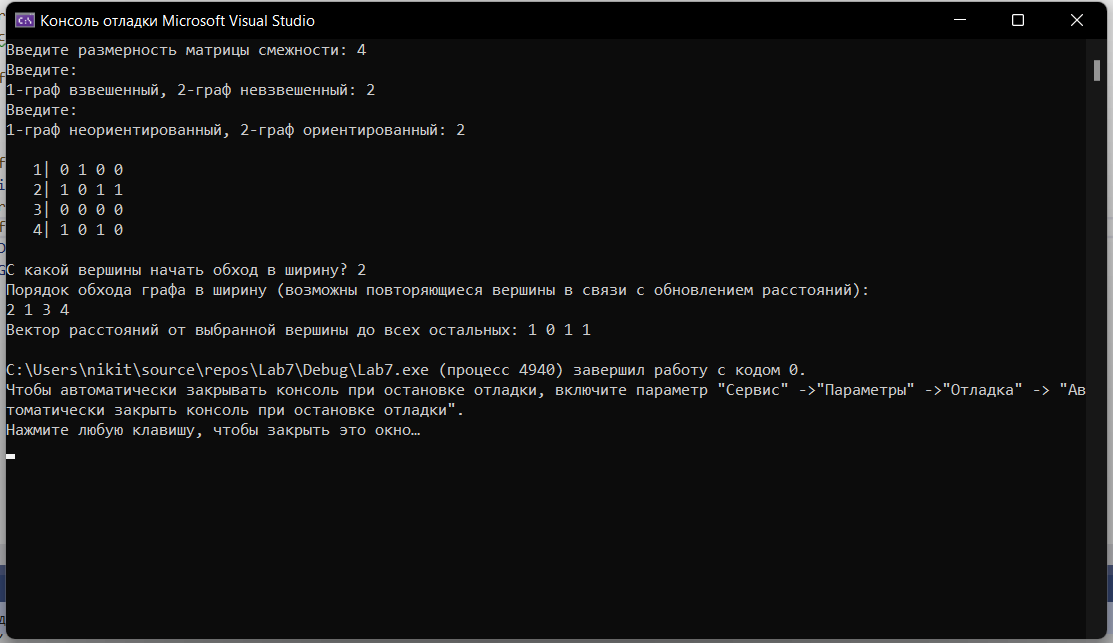
}

**Результаты программы:**









**Вывод:** составили программу, осуществляющую поиск расстояний во взвешенном графе.