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»

Выполнил: студент группы 19ВВ2

Трошкин В.К.

Принял: Д.Т.Н., профессор

Митрохин М.А.

Пенза 2020

**Цель работы:** составить программу поиска расстояний во взвешенном графе на ЭВМ.

**Ход работы:**

1. Сгенерировали (используя генератор случайных чисел) матрицу *M* смежности неориентированного взвешенного графа *G*. Вывели сгенерированную матрицу на экран.
2. Для матричной формы представления графов выполнили поиск расстояний путём обхода в ширину.
3. Результат выполнения операции вывели на экран.

**Листинг:**

Header.h

#pragma once

#include <stdio.h>

int pr\_adj\_matrix(char\*\* matrix, int size);

char\*\* gen\_adj\_matrix(int size);

void POSH(int v, char\*\* matrix, int size);

int POG(char\*\* matrix, int size);

main.cpp

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

#include <windows.h>

#include "Header.h"

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int size;

int start;

printf("Введите размер матрицы: ");

scanf("%d", &size);

char\*\* adj = gen\_adj\_matrix(size);

printf("Матрица смежности: \n");

pr\_adj\_matrix(adj, size);

printf("\n");

printf("Из какой вершины начинаем?\n");

scanf("%d", &start);

printf("\nПорядок обхода:\n");

POSH(start, adj, size);

\_getch();

}

adjmatrixgen.cpp

#include <time.h>

#include <stdlib.h>

#include "Header.h"

char\*\* gen\_adj\_matrix(int size) {

srand(time(0));

char\*\* matrix = (char\*\*)calloc(size, sizeof(char\*));

if (matrix == NULL)

return NULL;

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

matrix[i] = (char\*)calloc(size, sizeof(char));

if (matrix[i] == NULL)

return NULL;

}

do {

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

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

if (i == j) {

matrix[i][j] = 0;

}

else {

matrix[i][j] = rand() % 3;

if (matrix[i][j] > 0) {

matrix[i][j] = rand() % 10 + 1;

}

matrix[j][i] = matrix[i][j];

}

}

}

} while(POG(matrix, size) != 1); //пока не сгенерируется связный граф

return matrix;

}

BFSD.cpp

#include <stdlib.h>

#include <stdio.h>

#include "Header.h"

#include <queue>

using namespace std;

void BFSD(char\*\* matrix, int\* dist, int size, int v) {

queue <int> q;

q.push(v);

dist[v] = 0;

while (!q.empty()) {

v = q.front();

q.pop();

printf("%d ", v);

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

if ((matrix[v][i] > 0) && ((dist[i] == -1) || (dist[i] > dist[v] + matrix[v][i]))) {

q.push(i);

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

}

}

}

}

void POSH(int v, char\*\* matrix, int size) {

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

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

dist[i] = -1;

}

BFSD(matrix, dist, size, v);

printf("\n\nНомер узла: ");

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

printf("%-4d", i);

}

printf("\nРасстояние: ");

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

printf("%-4d", dist[i]);

}

}

pr\_output.cpp

#include <stdio.h>

int pr\_adj\_matrix(char\*\* matrix, int size) {

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

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

printf("%-4d", matrix[i][j]);

}

printf("\n");

}

return 0;

}

search.cpp

#include <stdlib.h>

#include <stdio.h>

void DFS(char\*\* matrix, int v, char\* visited, int size) {

visited[v] = 1;

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

if ((visited[i] == 0) && (matrix[v][i]))

{

DFS(matrix, i, visited, size);

}

}

}

int POG(char\*\* matrix, int size) {

int connum = 0;

char\* visited = (char\*)calloc(size, sizeof(char));

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

visited[i] = 0;

}

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

{

if (visited[v] == 0) {

DFS(matrix, v, visited, size);

connum++;

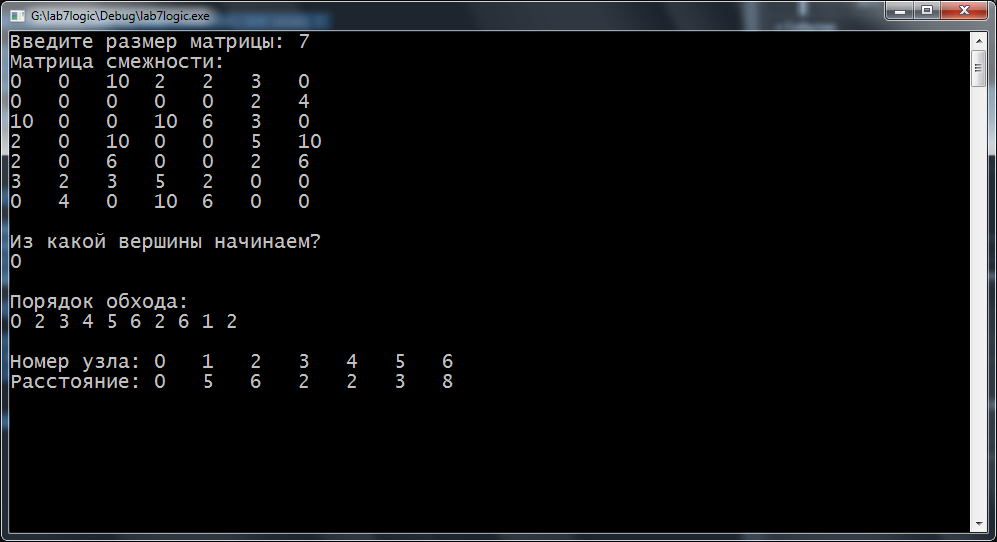
}

}

return connum;

}

Результаты работы программы:



**Вывод:** составили программу поиска путей во взвешенном графе в ширину на ЭВМ.