Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЁТ**

по лабораторной работе №7

по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему «Поиск расстояний во взвешенном графе»

Выполнили:

студенты группы 20ВВ2

Киреев Б.П.

Верховский М.В.

Лукин В.Д.

Приняли:

д.т.н. Митрохин М.А.

к.т.н. доцент Юрова О.В.

Пенза 2021

Название: Поиск расстояний во взвешенном графе.

Цель работы: Научиться осуществлять поиск расстояний во взвешенном графе.

Лабораторное задание:

**Задание 1:**

1. Сгенерируйте (используя генератор случайных чисел) матрицу

смежности для неориентированного 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.\* Реализуйте процедуру поиска расстояний для графа, представленного

списками смежности.

**Задание 2\*:**

1. Реализуйте процедуру поиска расстояний на основе обхода в глубину.

2. Реализуйте процедуру поиска расстояний на основе обхода в глубину

для графа, представленного списками смежности.

3. Оцените время работы реализаций алгоритмов поиска расстояний на

основе обхода в глубину и обхода в ширину для графов разных порядков.

**Задание 1,2:**

Листинг:

Файл Lab7Dop.cpp

#include <iostream>

#include <time.h>

#include <malloc.h>

#include <locale.h>

#include <queue>

#include <ctime>

#include <conio.h>

#include <climits> //Для предельного значения int

#include <string>

using namespace std;

int i, num = 0, pogr = 0;

int\* vis = NULL, n = 7, \*\* p = NULL;

int Wei = 1, Way = 1;

//Поиск расстояний от вершины графа

void BFSD(int v, int\*\* p) {

queue <int> q;

q.push(v);

vis[v] = 0;

while (!q.empty()) {

v = q.front();

q.pop();

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

if (vis[i] > vis[v] + p[i][v] && p[i][v] != 0) {

q.push(i);

vis[i] = vis[v] + p[i][v];

}

}

}

cout << endl;

}

//Функция для создания нужного графа в виде матрицы смежности // Weight = макимально возможный вес// Way = 1 / 0(Направлен граф или нет)// Size - размер

void CreateGraphFunc(int Weight, int Way, int Size) {

p = (int\*\*)calloc(Size, sizeof(int)); //Выделяем место под динамическую матрицу

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

p[i] = (int\*)calloc(Size, sizeof(int)); //Выделение места для строк динамической матрицы

}

for (int i = 0; i < n; i++) { //Генерация путей и их веса

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

p[i][m] = 0 + rand() % Weight;

switch (Way) {

case 1:

p[m][i] = 0 + rand() % Weight;

break;

case 0:

p[m][i] = p[i][m];

break;

}

p[i][i] = 0;

}

}

}

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

setlocale(LC\_ALL, "Rus");

srand(time(0));

cout.precision(3 \* n);

vis = (int\*)calloc(n, sizeof(int)); //Выделяем место под массив прохождения

for (i = 0; i < n; i++) { vis[i] = 0; } //Обнуляем массив прохождения

if (argc <= 3) {

cout << "Отсутствуют параметры для работы программы.\nАвтоматическое создание." << endl;

CreateGraphFunc(2, 0, 7);

}

else {

Wei = atoi(argv[1]);

Way = atoi(argv[2]);

n = atoi(argv[3]);

CreateGraphFunc(Wei, Way, n);

}

cout << "\nМатрица:" << endl; //Вывод динамической матрицы

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

for (int m = 0; m < n; m++) { cout << p[i][m] << " "; }

cout << endl;

}

for (int i = 0; i < n; i++) { vis[i] = INT\_MAX; } //Для нахождения минимальной длинны пути, нужно максимальное значение int

BFSD(0, p);

for (int i = 1; i < n; i++) { cout << "Расстояние от вершины 1 до вершины " << i + 1 << " равно " << vis[i] << endl; } //Вывод расстояний

cout << endl;

\_getch();

return 0;

}

Пояснительный текст к программе:

Данная программа осуществляет поиск расстояний во взвешенном/невзвешенном ориентированном/неориентированном графе. Программа модернизирована так, чтобы была возможность запуска с параметрами командной строки. В качестве параметра указываются максимально возможный вес, наличие ориентации ребер (есть ориентация(1) или нет(0)) и размерность графа.

Результаты работы программы:



Рисунок №1 – Запуск программы с параметрами командной строки.

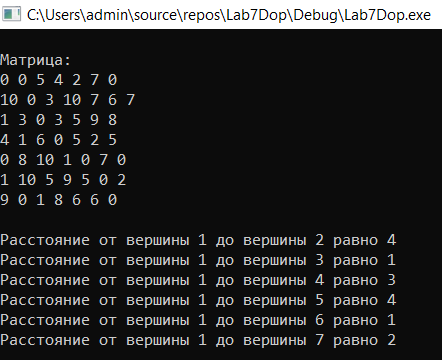


Рисунок №2 – Запуск программы через командную строку.

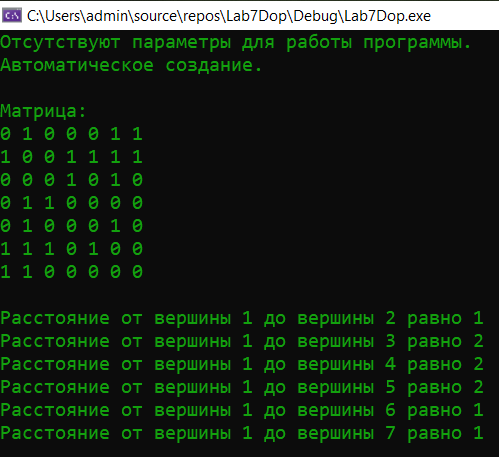


Рисунок №3 – Результат работы программы без параметров командной строки.

Вывод: При выполнении данной лабораторной работы мы смогли научиться осуществлять поиск расстояний во взвешенном/невзвешенном ориентированном/неориентированном графе.