Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЁТ**

по лабораторной работе №10

по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему «Поиск расстояний во взвешенном графе»

Выполнил:

студент группы 21ВВ1

Федулов Е.

Жбанников Д.

Приняли:

Юрова О.В.

Акифьев И.В.

Пенза 2022

**Цель работы:** научиться писать программы для осуществления поиска расстояний во взвешенном графе на языке Си.

**Задание 1**

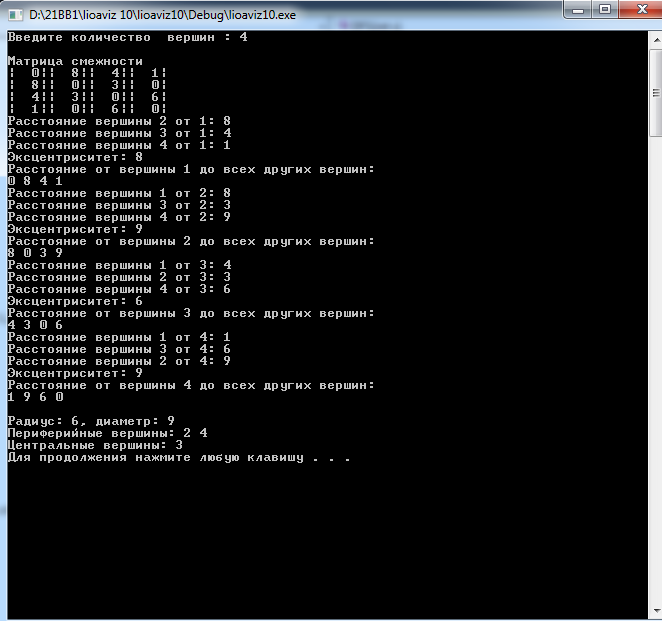
1. Сгенерируйте (используя генератор случайных чисел) матрицу смежности для неориентированного взвешенного графа *G*. Выведите матрицу на экран.
2. Для сгенерированного графа осуществите процедуру поиска расстояний, реализованную в соответствии с приведённым выше описанием. При  реализации алгоритма в качестве очереди используйте класс **queue** из стандартной библиотеки С++.

**Задание 2**

1. Для каждого из вариантов сгенерированных графов (ориентированного и не ориентированного) определите радиус и диаметр.
2. Определите подмножества периферийных и центральных вершин.

**Ход работы**

**Результат работы программы**

****

**Вывод**

В ходе выполнения лабораторной работы были разработаны программы, выполняющие поиск расстояний от каждой вершины графа до всех остальных вершин, а также поиск радиуса и диаметра графа и соответствующих им центральное и периферийное множества вершин графа.

Получили опыт в создании проектов в среде MicrosoftVisualStudio, научились писать и отлаживать программы с применением бинарных деревьев на языке Си.

**Код**

//#include "stdafx.h"

#defineCRT\_SECURE\_NO\_WARNINGS

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

#include<locale.h>

#include<Windows.h>

#include<malloc.h>

#include<time.h>

#include<iostream>

#include<queue>

usingnamespace std;

void DFS(intv);

int n;

int\* vis;

int v = 0;

int\*\* M;

int\* maxim;

void main() {

setlocale(LC\_ALL, "rus");

printf("Введите количество элементов в квадратичной матрице: ");

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

srand((unsigned)time(NULL));

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

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

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

}

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

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

vis[i] = -1;

}

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

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

maxim[i] = 0;

}

//Задание 1

//Генерация матрицы смежности:

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

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

if (rand() % 2) {

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

}

else {

M[i][j] = 0;

}

if (i == j) {

M[i][j] = 0;

}

M[j][i] = M[i][j];

}

}

//Вывод матрицы смежности:

printf\_s("\nМатрица смежности");

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

printf\_s("\n");

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

printf("|%3d|", M[i][j]);

}

}

printf("\n");

for (int s = 1; s < (n + 1); s++) {

DFS(s);

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

vis[i] = -1;

}

}

printf("\n");

int maxm = 0;

int minm = 100;

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

if (maxm < maxim[i]) {

maxm = maxim[i];

}

elseif (minm > maxim[i] && maxim[i] > 0) {

minm = maxim[i];

}

}

printf("Радиус: %d, диаметр: %d\n", minm, maxm);

printf("Периферийные вершины: ");

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

if (maxim[i] == maxm) {

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

}

}

printf("\n");

printf("Центральные вершины: ");

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

if (maxim[i] == minm) {

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

}

}

printf("\n");

system("pause");

}

void DFS(ints) {

queue<int> q;

q.push(s - 1);

vis[s - 1] = 1;

int st = s;

while (!q.empty()) {

s = q.front();

//printf("%d\n", s+1);

q.pop();

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

if ((M[s][i] != 0) && (vis[i] < 0)) {

q.push(i);

vis[i] = vis[s] + M[s][i];

printf("Расстояние вершины %d от %d: %d\n", i + 1, st, vis[i] - 1);

}

}

}

maxim[st - 1] = 0;

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

if (maxim[st - 1] < vis[i]) {

maxim[st - 1] = vis[i] - 1;

}

}

printf("Эксцентриситет: %d\n", maxim[st - 1]);

printf("Расстояние от вершины %d до всех других вершин:\n", st);

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

printf("%d ", vis[i] - 1);

}

printf("\n");

}