МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра «Вычислительной техники»

**ОТЧЁТ**

по лабораторной работе №10

по курсу «Л и ОА в ИЗ»

на тему «Поиск расстояний в графе»

Выполнили:

Студенты группы 22ВВС1

Костин Максим

Макеева Дарья

Приняли:

Акифьев И.В.

Юрова О.В.

Пенза 2023

**Лабораторноезадание*:***

**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++.

### Задание 2

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

**Листинг:**

#include <stdio.h>

#include <locale.h>

#include <time.h>

#include <stdlib.h>

#include <conio.h>

#include <queue>

using namespace std;

void bfs(int v, int\* dist, int n, int\*\* matr, int g) {

queue <int> q;

q = {};

q.push(v);

dist[v] = 0;

while (q.size() != 0) {

v = q.front();

q.pop();

if (g == 1)

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

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

if (matr[v][i] > 0 && dist[i] == -1) {

q.push(i);

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

}

}

}

}

int main() {

setlocale(LC\_ALL, "Rus");

srand(time(NULL));

int n = 0;

printf("Введите количество вершин: ");

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

int\*\* matr1 = (int\*\*)malloc(sizeof(int\*) \* n);

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

matr1[i] = (int\*)malloc(sizeof(int) \* n);

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

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

matr1[i][j] = 0;

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

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

matr1[i][j] = rand() % 15;

matr1[j][i] = matr1[i][j];

if (i == j)

matr1[i][j] = 0;

}

}

printf("\n Матрица M1: \n№ ");

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

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

}

printf("\n\_\_|");

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

printf("\_\_");

}

printf("\n");

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

{

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

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

printf("%d ", matr1[i][j]);

printf("\n");

}

}

int ver = 0, op = 0, diam = 0, rad = 0, ch = 0, ch1 = 0;

while (1) {

printf("\nВведите номер операции:\n1)Выбрать вершину для обхода 2)Найти радиус, диаметр, подмножества периферийных и центральных вершин 3)Выйти\n");

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

switch (op) {

case 1:

{

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

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

dist[i] = -1;

printf("Выберите вершину, с которой начинать обход:");

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

if (ver <= n && ver > 0) {

ver -= 1;

printf("Обход в ширину: \n");

bfs(ver, dist, n, matr1, 1);

printf("\n\nРасстояния от указанной вершины\n\n");

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

printf("[%d] = %d\t", i + 1, dist[i]);

free(dist);

}

else

printf("Нет вершины!");

break;

}

case 2:

{

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

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

eks[i] = -1;

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

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

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

dist[i] = -1;

bfs(ver, dist, n, matr1, 0);

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

if (dist[i] > eks[ver])

eks[ver] = dist[i];

}

free(dist);

}

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

printf("%d\t", eks[i]);

if (diam < eks[i])

diam = eks[i];

}

rad = diam;

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

if (rad > eks[i])

rad = eks[i];

}

printf("\nДиаметр = %d, радиус = %d\n", diam, rad);

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

if (eks[i] == rad)

printf("\n%d центральная вершина\n", i + 1);

if (eks[i] == diam)

printf("\n%d периферийная вершина\n", i + 1);

}

free(eks);

break;

}

case 3:

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

free(matr1[i]);

}

free(matr1);

return 0;

}

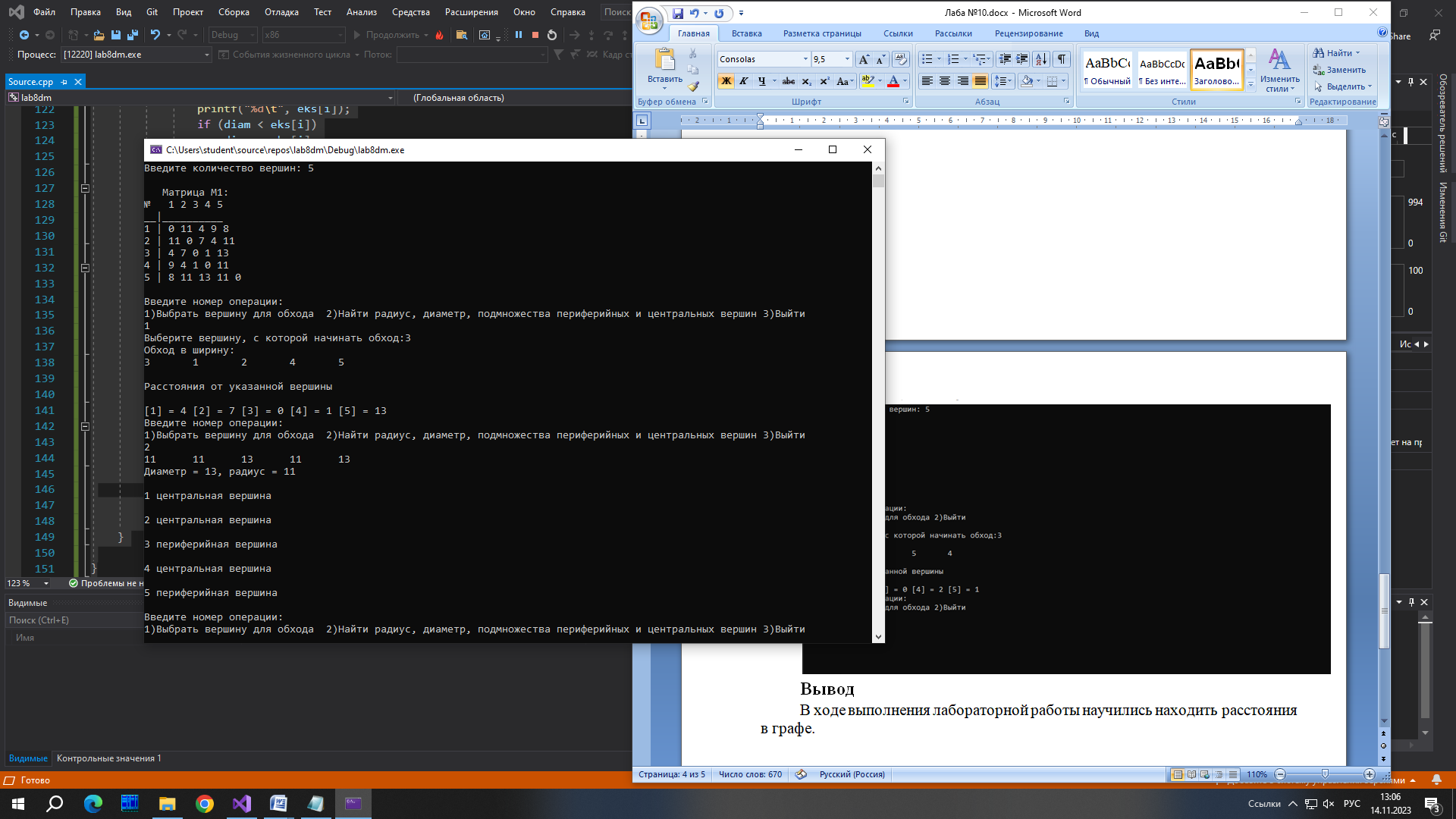
}

# }

# Пояснительный текст к программе:

При реализации алгоритма в качестве очереди использовали класс **queue** из стандартной библиотеки С++, нужный для создания очереди.

**Результат работы программы**

****

**Вывод**

В ходе выполнения лабораторной работы научились находить расстояния во взвешенном графе, также находить диаметр и радиус графа ,периферийные и центральные вершины.