Министерство науки Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №10

по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему: «Поиск расстояний во взвешенном графе»

Выполнили студенты

группы 21ВВ4:

Шутихин А.Э.

Приняли:

Юрова О. В.

Акифьев И.В.

Пенза, 2022

**Цель работы:** научиться находить расстояния до вершин в взвешенном неориентированном графе, используя алгоритм обхода в ширину.

**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++.

**3.**\* Сгенерируйте (используя генератор случайных чисел) матрицу смежности для ориентированного взвешенного графа *G*. Выведите матрицу на экран и осуществите процедуру поиска расстояний, реализованную в соответствии с приведенным выше описанием.

**Задание 2**

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

**Задание 3\***

1. Модернизируйте программу так, чтобы получить возможность запуска программы с параметрами командной строки (см. описание ниже).  В качестве параметра должны указываться тип графа (взвешенный или нет) и наличие ориентации его ребер (есть ориентация или нет).

**Ход работы**

**Листинг:**

#include <stdio.h>

#include <time.h>

#include <stdlib.h>

#include <locale.h>

#include <queue>

using namespace std;

queue<int>q;

void BFSD(int s, int n, int\* dis, int\*\* G,int\*ex) {

int l = s;

q.push(s);

dis[l] = 0;

while (!q.empty()) {

l = q.front();

q.pop();

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

{

if (G[l][i] != 0 && dis[i] == -1)

{

q.push(i);

dis[i] = dis[l] + G[l][i];

printf("%d[%d->%d] ", dis[i],l,i);

if (dis[i] > ex[l])

ex[l] = dis[i];

}

}

}

}

int main() {

int n, i = 0;

int R = 100, D = 0;

srand(time(NULL));

setlocale(LC\_ALL, "Russian");

printf("Введите количество вершин для матрицы: ");

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

int\*\* G = new int\* [n];

int\* dis = new int [n];

int\* ex = new int[n];

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

{

G[i] = new int[n];

ex[i] = 0;

}

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

{

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

{

if (i == j)

{

G[i][j] = 0;

}

else if (i != j)

{

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

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

}

}

}

printf("Матрица смежности:\n");

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

{

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

{

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

}

printf("\n");

}

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

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

dis[j] = -1;

BFSD(i, n, &dis[0], &(&G)[0][0],&ex[0]);

printf("\n");

}

printf("Эксцентриситет\n");

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

{

printf("%d[%d]\n", ex[i], i);

if (ex[i] > D)

D = ex[i];

if (ex[i] < R)

R = ex[i];

}

printf("Диаметр - %d\nРадиус - %d",D, R);

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

{

if (ex[i] == D)

printf("\nПерифирийная вершина - % d", i);

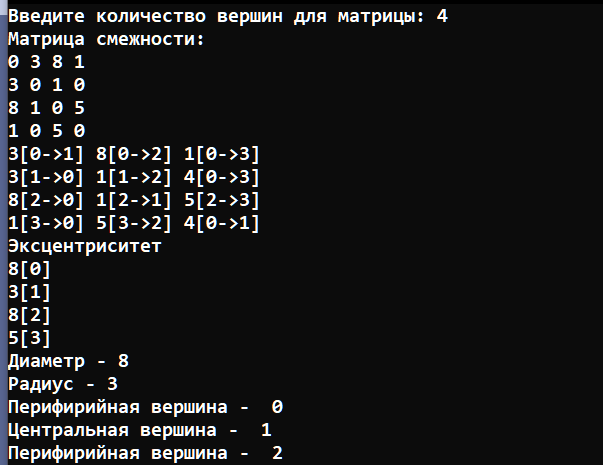
if (ex[i] == R)

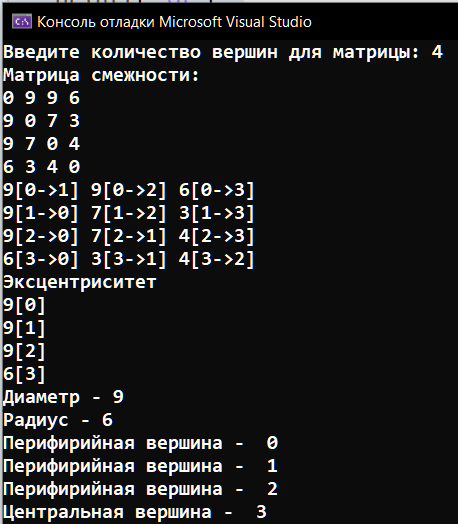
printf("\nЦентральная вершина - % d", i);

}

}

**Результат:**





**Вывод:** Я научился находить расстояния до вершин в взвешенном неориентированном графе, используя алгоритм обхода в ширину.