Пензенский государственный университет

Кафедра «Вычислительная техника»

**Отчёт**

По лабораторной работе №8

по дисциплине: «Логика и основы оптимизации в инженерных задачах»

## на тему «Определение характеристик графов»

Выполнил студент группы 19ВВ1:

Гусев В.

Приняли:

Митрохин М. А.

Юрова О.В.

Пенза 2020

**Цель работы:** освоить алгоритм определения различных характеристик графа.

**Общие сведения:**

Эксцентриситет вершины – расстояние до наиболее

удаленной вершины графа.

Максимальный эксцентриситет среди эксцентриситетов всех вершин

графа называется диаметром графа G и обозначается через D(G).

Вершина v i называется периферийной, если её эксцентриситет равен

диаметру графа e(v i ) = d(G).

Минимальный из эксцентриситетов вершин графа называется его

радиусом и обозначается через r(G).

Вершина v i называется центральной, если её эксцентриситет равен

радиусу графа e(v i ) = r(G).

Множество всех центральных вершин графа называется его центром.

Граф G может иметь единственную центральную вершину или несколько

центральных вершин.

Степенью вершины графа G называется число инцидентных ей ребер.

Степень вершины v i обозначается через deg(v i ).

Вершина v i со степенью 0 называется изолированной, со степенью 1 –

концевой.

Вершина графа, смежная с каждой другой его вершиной, называется

доминирующей.

**Практическая часть**

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++.

**Листинг.**

// Lab\_8.cpp: определяет точку входа для консольного приложения.

//

#include "stdafx.h"

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <stdio.h>

#include <conio.h>

#include <malloc.h>

#include <stdlib.h>

#include <time.h>

#include <queue>

using namespace std;

int\*\* G;

int\* dist;

int ver;

int\* ext;

int D=0;

int R=1000;

int\* stp;

void BFSD(int s, int rows) {

int v;

queue <int> q;

q.push(s);

dist[s] = 0;

while (!q.empty())

{

v = q.front();

q.pop();

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

{

if (G[v][i] > 0 && dist[i] > dist[v] + G[v][i])

{

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

q.push(i);

}

}

}

}

int main()

{

int rows;

int n = 0;

int k = 0;

system("cls");

printf("rows:");

scanf("%d", &rows);

srand(time(NULL));

stp = (int\*)malloc(rows \* sizeof(int));

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

{

stp[i] = 0;

}

G = (int\*\*)malloc(rows \* sizeof(int\*));

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

{

G[i] = (int\*)malloc(rows \* sizeof(int));

}

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

{

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

{

G[i][j] = rand() % 5;

if (i == j)

{

G[i][j] = 0;

}

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

}

}

printf("Matrix smeg\n");

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

{

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

{

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

if (G[i][j] > 0)

{

stp[i]++;

}

}

printf("\n");

}

printf("\n");

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

ext = (int\*)malloc(rows \* sizeof(int));

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

{

ext[i] = -1;

}

printf("\n");

for (int a = 0; a < rows; a++)

{

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

{

dist[i] = 1000;

}

BFSD(a, rows);

printf("Dist do %d versh: \n", a);

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

{

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

if (ext[a] < dist[i])

{

ext[a] = dist[i];

}

}

printf("\n");

}

printf("\n");

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

{

printf("Excentrisitet %d versh %d\n",i , ext[i]);

if (ext[i] > D && ext[i] !=0 && ext[i] != 1000)

{

D = ext[i];

}

if (ext[i] < R && ext[i]!=0 && ext[i] != 1000)

{

R = ext[i];

}

}

printf("\n");

printf("Diametr: %d\nRadius: %d", D, R);

printf("\n");

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

{

printf("Stepen %d versh: %d\n",i , stp[i]);

}

printf("\nIzolir versh:");

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

{

if (stp[i] == 0)

{

printf("%d ", i);

}

}

printf("\nKoncevie versh:");

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

{

if (stp[i] == 1)

{

printf("%d ", i);

}

}

printf("\nDominir versh:");

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

{

if (stp[i] == rows-1)

{

printf("%d ", i);

}

}

printf("\nCentraln versh:");

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

{

if (ext[i] == R)

{

printf("%d ", i);

}

}

printf("\nPerifer versh:");

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

{

if (ext[i] == D)

{

printf("%d ", i);

}

}

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

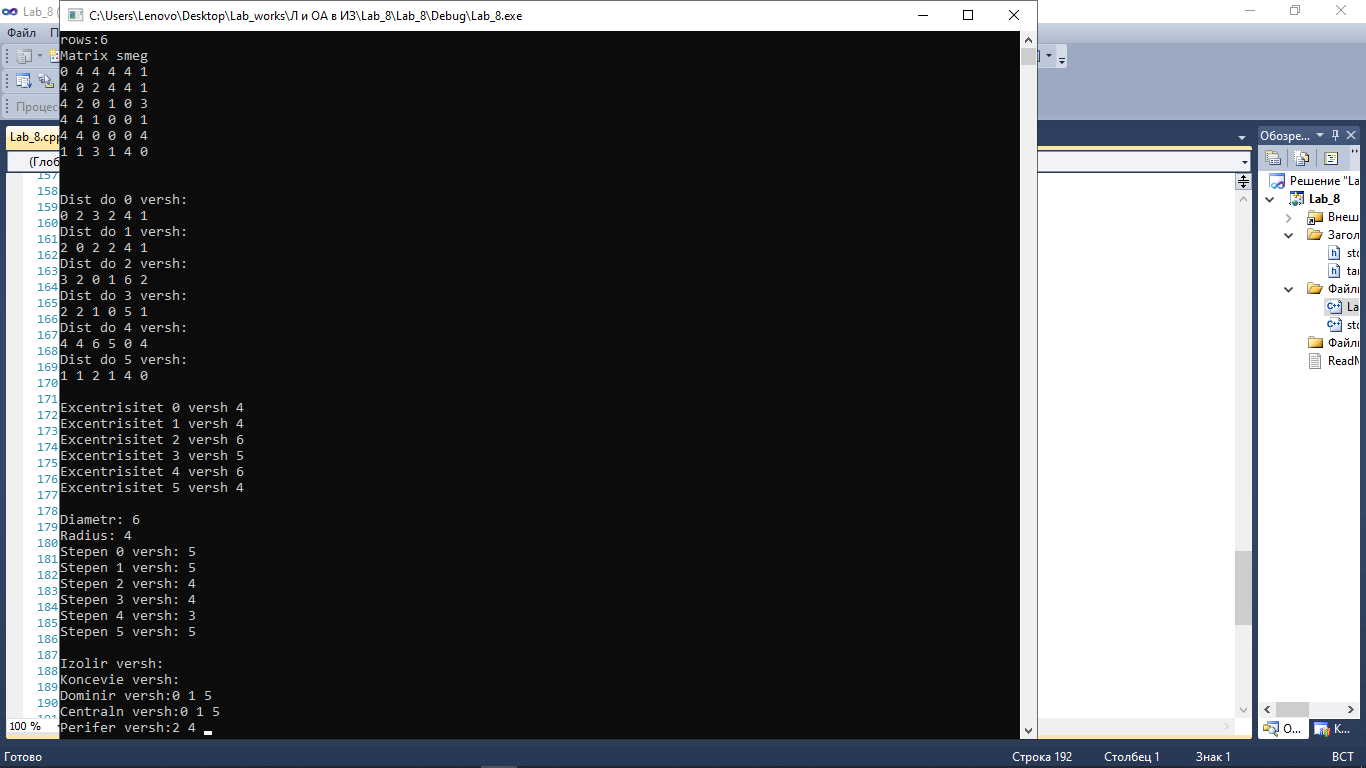
free(G[i]);

free(G);

getch();

}

## Результат работы.



**Вывод:** освоили алгоритмы определения характеристик графа таких как: эксцентриситет вершины, диаметр графа, радиус графа , периферийных вершин, центральной вершины, степень вершины, концевых вершин , доминирующих вершин.