Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №8

# по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему «Определение характеристик графов»

**Выполнили:**

студенты группы 20ВВ3

Новиков Иван

**Приняли:**

Юрова О.В.

Митрохин М.А.

Пенза 2021

# **Название**

**Определение характеристик графов**

**Цель работы** – изучение характеристик графа. Реализация алгоритмов определения этих характеристик.

# **Методические указания**

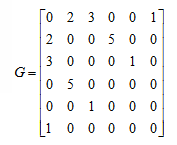
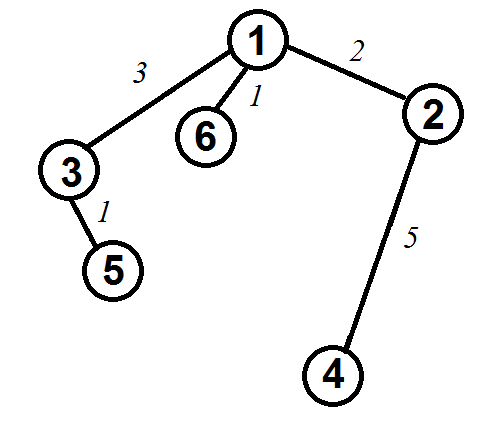
Если *G -* граф, содержащий непустое множество*n* вершин *V* и множество ребер *E* и *d(vi, vj)–* расстояние между двумя произвольными вершинами *vi* и*vj*, тогда для фиксированной вершины *v* величина

,

где *v*, *vj*∈*V* и *j* = 1…*n*называется эксцентриситетом вершины *vi*.

Другими словами эксцентриситетвершины – расстояние до наиболее удаленной вершины графа.

Например, для вершины 1 графа, показанного на рисунке 1 эксцентриситет равен 7.



*Рисунок 1 – Граф*

Максимальный эксцентриситетсреди эксцентриситетов всех вершин графа называется диаметромграфа *G* и обозначается через *D(G).*

Вершина *vi* называется периферийной*,* если её эксцентриситет равен диаметру графа *e(vi) = d(G).*

Минимальный из эксцентриситетов вершин графа называется его радиусом и обозначается через *r(G).*

Вершина *vi* называется центральной*,* еслиеё эксцентриситет равен радиусу графа *e(vi) = r(G).*

Множество всех центральных вершин графа называется его центром*.* Граф *G* может иметь единственную центральную вершину или несколько центральных вершин.

Степеньювершины графа *G* называется число инцидентных ей ребер. Степень вершины *vi* обозначается через *deg(vi).*

Вершина *vi* со степенью 0 называется изолированной*,* со степенью 1 – концевой*.*

Вершина графа, смежная с каждой другой его вершиной, называется доминирующей.

**Лабораторное задание**

Задание 1.

1. Сгенерируйте (используя генератор случайных чисел) матрицу смежности для неориентированного взвешенного графа *G*. Выведите матрицу на экран.
2. Определите радиус и диаметр графа *G*, используя матрицу смежности графа.
3. Определите подмножества периферийных и центральных вершин графа *G*, используя матрицу смежности.
4. Найдите изолированные, концевые и доминирующие вершины.

Задание 2.

1. Постройте для графа G матрицу инцидентности.
2. Определите радиус и диаметр графа *G*, используя матрицу инцидентности графа.
3. Определите подмножества периферийных и центральных вершин графа *G*, используя матрицу инцидентности.

**Ход работы**

**Листинг**

#define\_CRT\_NONSTDC\_NO\_WARNINGS

#define\_CRT\_SECURE\_NO\_WARNINGS

#include<iostream>

#include<stdio.h>

#include<stdlib.h>

#include<time.h>

#include<locale.h>

#include<string.h>

#include<windows.h>

#include<queue>

usingnamespace std;

void matrix(int\*\* g, intsize)

{

int n = 1;

srand(time(NULL));

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

{

g[i] = (int\*)malloc(size \* sizeof(int));

g[i][i] = 0;

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

{

if (rand() % 100 > 72)

{

g[i][j] = 0;

}

else

{

g[i][j] = rand() % size;

}

}

n++;

}

n = 1;

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

{

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

g[j][i] = g[i][j];

n++;

}

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

{

printf("%2d ", i + 1);

}

printf("\n");

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

{

printf("\_\_\_");

}

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

{

printf("\n");

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

{

printf("%2d ", g[i][j]);

}

}

printf("\n");

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

{

printf("\_\_\_");

}

}

void BFSD(intv, int\* dist, int\*\* g, intsize)

{

queue<int> q;

q.push(v);

dist[v] = 0;

while (!q.empty())

{

v = q.front();

q.pop();

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

{

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

{

q.push(i);

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

}

}

}

//printf("\n\n");

//for (int i = 0; i < size; i++)

//{

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

//}

}

void F(int\*\* g, intsize)

{

int d = -1, r = 10000, n = 0;

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

int\* k = (int\*)malloc(size \* sizeof(int)), \* is = (int\*)malloc(size \* sizeof(int)), \* kon = (int\*)malloc(size \* sizeof(int)), \* dom = (int\*)malloc(size \* sizeof(int)), \* extra = (int\*)malloc(size \* sizeof(int));

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

{

for (int k = 0; k <size; k++)

{

dist[k] = 9999;

}

BFSD(i, dist, g, size);

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

{

if (extra[i] < dist[j] && dist[j] != 9999)

{

extra[i] = dist[j];

}

}

}

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

{

if (d < extra[i])

{

d = extra[i];

}

if (r > extra[i] && extra[i] != 0)

{

r = extra[i];

}

}

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

{

k[i] = 0;

}

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

{

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

{

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

{

k[i]++;

}

}

}

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

{

is[i] = 0;

}

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

{

if (k[i] == 0)

{

is[i] = 1;

}

}

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

{

kon[i] = 0;

}

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

{

if (k[i] == 1)

{

kon[i] = 1;

}

}

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

{

dom[i] = 0;

}

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

{

if (k[i] == size - 1)

{

dom[i] = 1;

}

}

printf("\nR: %d\n", r);

printf("D: %d\n", d);

printf("Эксцентриситет:");

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

{

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

}

printf("\nПереферийные вершины:");

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

{

if (extra[i] == d) printf("%d ", i + 1);

}

printf("\nЦентральные вершины:");

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

{

if (extra[i] == r)

{

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

}

}

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

{

if (is[i] == 1)

{

printf("\nИзолированная: %d", i + 1);

}

//else

//{

// printf("\nНе изолированная: %d", i + 1);

//}

}

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

{

if (kon[i] == 1) {

printf("\nКонцевая: %d", i + 1);

}

//else

//{

// printf("\nНе концевая: %d", i + 1);

//}

}

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

{

if (dom[i] == 1) {

printf("\nДоминантная: %d", i + 1);

}

//else

//{

// printf("\nНе доминантная: %d", i + 1);

//}

}

}

void main()

{

setlocale(LC\_ALL, "RUS");

int i = 0;

int j = 0;

int size = 0;

int\*\* g = 0;

int v = 0;

setlocale(LC\_ALL, "RUS");

srand(time(NULL));

printf("Размер матрицы: ");

scanf("%d", &size);

g = (int\*\*)malloc(size \* sizeof(int\*));

for (i = 0; i < size; i++)

{

g[i] = (int\*)malloc(size \* sizeof(int));

}

matrix(g, size);

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

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

{

dist[i] = INT16\_MAX;

}

F(g, size);

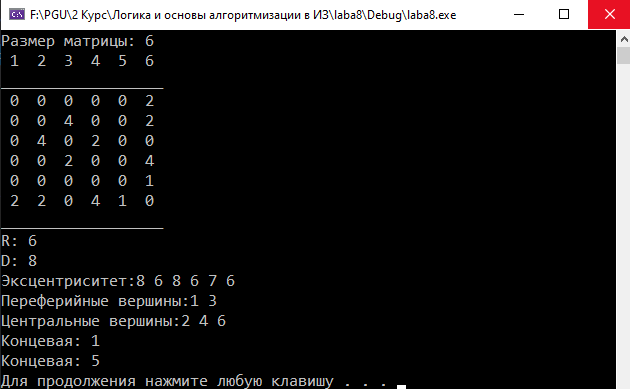
printf("\n");

system("pause");

}

**Задание 1:**

1. Сгенерировали матрицу смежности



1. Определили радиус и диаметр графа, используя матрицу смежности.
2. Определили подмножества периферийных и центральных вершин.
3. Нашли концевые, изолированные и доминирующие вершины.

**Вывод:** узнали, какие характеристики графа существуют, написали и протестировали программу, которая определяет эти характеристики.