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:**

студент группы 20ВВ3

Пантюшов Егор

**Приняли:**

Юрова О.В.

Митрохин М.А.

Пенза 2021

# Название

Определение характеристик графов

**Цель работы** – изучение характеристик графа и алгоритмов их определения

# Методические указания

**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.

# Лабораторное задание

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.

# Листинг

#define \_CRT\_NONSTDC\_NO\_WARNINGS

#define \_CRT\_SECURE\_NO\_WARNINGS

#define HEADER ("Лабораторная работа №6\nВыполнил:Пантюшов Е.\n\n")

#include <queue>

#include <stdio.h>

#include <locale.h>

#include <stdlib.h>

#include <time.h>

#include <limits>

using namespace std;

void bfs(int\*\* G, int\* DIST, int v, int n)

{

queue <int> q;

q.push(v);

DIST[v] = 0;

while (!q.empty())

{

v = q.front();

q.pop();

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

{

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

{

q.push(i);

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

}

}

}

}

void task\_1\_1(int\*\* G, int n)

{

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

{

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

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

{

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

if (G[i][j] < 5)

{

G[i][j] = 0;

}

else

{

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

}

}

}

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

{

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

{

if (i == j)

{

G[i][j] = 0;

}

else

{

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

}

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

}

printf("\n");

}

}

void task\_1\_2(int\*\* G, int n,int \*r,int \*d,int \*EX)

{

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

int v;

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

{

EX[i] = -1;

}

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

{

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

{

DIST[j] = INT\_MAX;

}

bfs(G, DIST, i, n);

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

{

if ((EX[i] < DIST[k]) && (DIST[k] != INT\_MAX))

EX[i] = DIST[k];

}

}

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

printf("Эксцентриситет %d-ой вершины= %d \n", i, EX[i]);

printf("\n");

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

{

if (\*r > EX[i])

\*r = EX[i];

}

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

{

if (\*d < EX[i])

\*d = EX[i];

}

printf("Диаметр графа=%d\n", \*d);

printf("Радиус графа=%d\n", \*r);

free(DIST);

}

void task\_1\_3(int\*\* G, int n,int \*d,int \*r,int \*EX)

{

printf("Множество периферийных вершин:");

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

{

if (EX[i] == \*d)

{

printf("%d ", i);

}

}

printf("\n");

printf("Множество центральных вершин:");

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

{

if (EX[i] == \*r)

{

printf("%d ", i);

}

}

printf("\n");

}

void task\_1\_4(int\*\* G, int n)

{

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

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

{

VER[i] = 0;

}

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

{

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

if (G[i][j] != 0)

VER[i] = VER[i] + 1;

}

printf(" Изолированные вершины:");

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

{

if (VER[i] == 0)

printf("%d ", i);

}

printf("\n");

printf(" Кольцевые вершины:");

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

{

if (VER[i] == 1)

printf("%d ", i);

}

printf("\n");

printf(" Доминирующие вершины:");

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

{

if (VER[i] == n - 1)

printf("%d ", i);

}

printf("\n");

}

int main()

{

setlocale(LC\_ALL, "RUS");

srand(time(NULL));

printf(HEADER);

printf("Введите размерность матрицы: ");

int n;

scanf("%d", &n);

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

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

int r, d;

r = INT\_MAX;

d = -1;

task\_1\_1(G, n);

task\_1\_2(G, n,&r,&d,EX);

task\_1\_3(G, n, &d, &r, EX);

task\_1\_4(G, n);

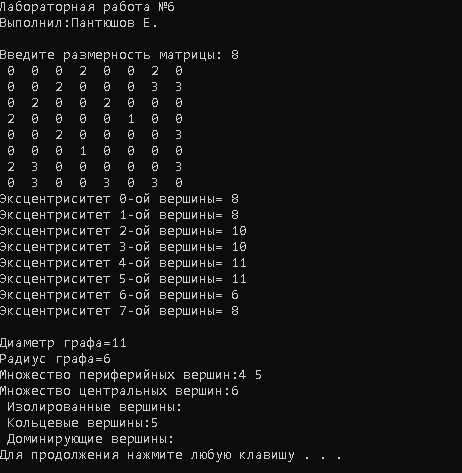
system("PAUSE");

return 0;

}

# Результат работы программы

Результат работы программы показан на рисунке



**Рисунок – Результат работы программы**

# Вывод

# В ходе выполнения лабораторной работы была разработана программа, в который были реализованы алгоритмы нахождения характеристик графа.