Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №6

по курсу «Л и ОА в ИЗ»

на тему «Определение характеристик графа»

**Выполнили:**

студенты группы 19ВВ2

Сидоров Н. Р.

Карамышев А. А.

Горбунов Д.А.

**Приняли:**

Юрова О. В.

Митрохин М. А.

Пенза 2020

**Название:**

Определение характеристик графа

**Цель работы:**

Определять характеристики графа: эксцентриситеты вершин, диаметр и радиус графа, является ли вершина периферийной, изолированной, центральной, концевой и доминирующей.

**Лабораторные задания:**

### ***Задание 1***

1. Сгенерировали (используя генератор случайных чисел) матрицу смежности для неориентированного графа G. Вывели матрицу на экран.

2. Определили радиус и диаметр графа G, используя матрицу смежности графа.

3. Определили подмножества периферийных и центральных вершин графа G, используя матрицу смежности.

4. Нашли изолированные, концевые и доминирующие вершины.

**Описание понятий для решения задач:**

*Эксцентриситет вершины* – расстояние до наиболее удаленной вершины графа.

*Диаметр –* максимальный эксцентриситет среди всех вершин в графе.

*Центральная вершина –* если эксцентриситет вершины равен радиусу

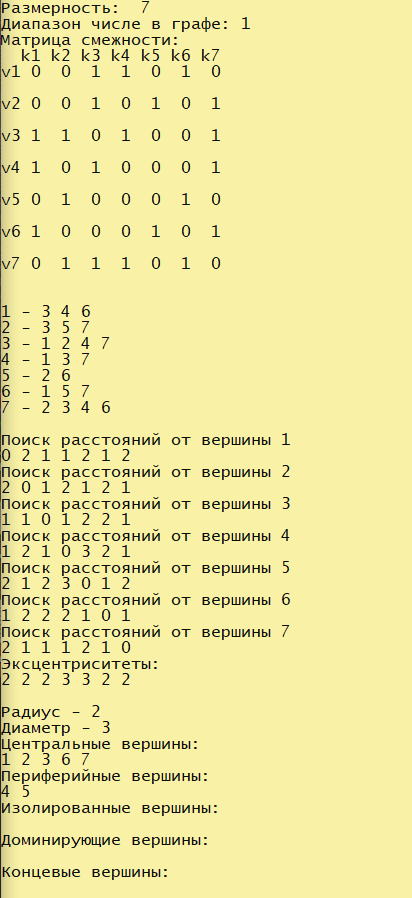
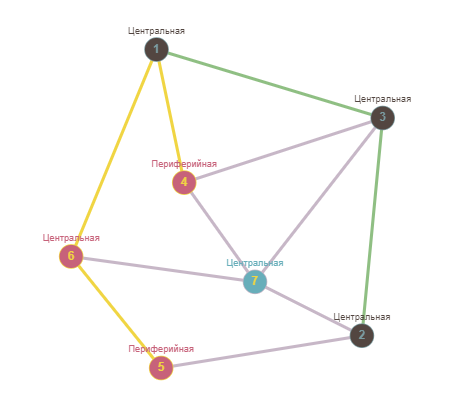
*Периферийная вершина –* если эксцентриситет вершины равен диаметру

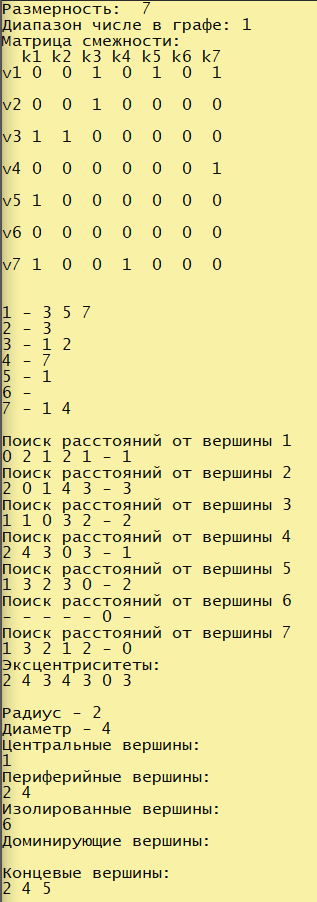
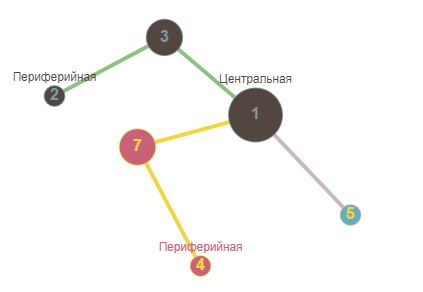
*Концевая вершина –* вершина со степенью 1 (т. е. только одна смежная вершина)

*Доминирующая вершина –* вершина, которая смежная с каждой вершиной в графе

*Изолированная вершина –* вершина со степенью 0 (т. е. не имеет смежных вершин)

**Результат работы программы:**





концевая

концевая

изолированная концевая

Исходя из результатов работы программы и ручного выявления разных характеристик графа и вершин на визуализации графа можно сделать вывод, что результаты в том и другом случае совпадают и программа работает правильно.

**Вывод:**

Научились определять различные характеристики графа: эксцентриситеты вершин, диаметр и радиус графа, является ли вершина периферийной, изолированной, центральной, концевой и доминирующей, на примере матрицы смежности.

Листинг 1:

#include <time.h>

#include <iostream>

#include <stack>

#include <queue>

#include <conio.h>

//работа сделана, есть баг с изолированной вершиной

using namespace std;

queue <int> Q;

int i, j, m, n, h;

int\* vis;

int\* v;

int\*\* graph;

int\* ex;

int\* e;

void BFS() {

for (int j = 0; j < m; j++) {

printf("\nПоиск расстояний от вершины %d\n", j + 1);

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

vis[i] = 1000;

v[j] = 0;

e[j] = 0;

}

int s = j;

Q.push(s);

int st = s;

vis[s] = 0;

while (!Q.empty())

{

s = Q.front();

Q.pop();

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

if ((graph[s][r] > 0) && (vis[r] > vis[s] + graph[s][r]))

{

Q.push(r);

vis[r] = vis[s] + graph[s][r];

}

}

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

{

//проверка на концевую вершину

if (vis[i] == h) {

e[j]++;

}

//проверка на изолированную вершину

if (vis[i] == 1000) {

printf("- ");

vis[i] = 0;

}

//вывод вершин

else

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

v[j] += vis[i];

if (vis[i] > ex[st])

{

ex[st] = vis[i];

}

}

}

int k = 0;

int rad = 100000;

int dia = 0;

printf("\nЭксцентриситеты: \n");

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

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

if (ex[i] == 0)

continue;

if (ex[i] > dia)

dia = ex[i];

if ((ex[i] < rad) && (ex[i] != -1))

rad = ex[i];

}

printf("\n\nРадиус - %d\nДиаметр - %d", rad, dia);

printf("\nЦентральные вершины: \n");

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

if (ex[i] == rad)

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

}

printf("\nПериферийные вершины: \n");

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

if (ex[i] == dia)

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

}

printf("\nИзолированные вершины: \n");

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

if (ex[i] == 0) {

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

k++;

}

}

printf("\nДоминирующие вершины: \n");

if (k == 0) {

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

if (v[i] == m - 1)

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

}

}

printf("\nКонцевые вершины: \n");

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

if (e[i] == h)

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

}

}

void DFS\_main()

{

setlocale(LC\_ALL, "Rus");

printf("Размерность: ");

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

printf("Диапазон числе в графе: ");

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

h++;

n = m;

graph = new int\* [m];

v = (int\*)malloc(m \* sizeof(int));

ex = (int\*)malloc(m \* sizeof(int));

e = (int\*)malloc(m \* sizeof(int));

vis = (int\*)malloc(m \* sizeof(int));

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

graph[i] = new int[m];

}

//генерация матрицы

int start;

printf("Матрица смежности: \n");

srand(time(NULL));

for (i = 0; i < m; ++i) {

for (j = i; j < m; ++j) {

graph[i][j] = graph[j][i] = rand() % h;

}

graph[i][i] = 0; //обнуление вершины

}

h--;

//нумерование столбцов

printf(" ");

for (j = 0; j < m; j++)

{

printf("k%d ", j + 1);

}

printf("\n");

//нумерование строк

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

{

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

for (j = 0; j < m; ++j)

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

printf("\n\n");

}

//вывод списков смежности (работает только с невзвешенным графом)

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

{

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

for (j = 0; j < m; j++) {

if (graph[i][j] != 0) printf("%d ", j + 1);

}

}

printf("\n");

BFS();

\_getch();

}

int Menu(void)

{

system("CLS");

int c = 0;

setlocale(LC\_ALL, "Rus");

while (c < '0' || c > '1')

{

printf(

"::=================================::\n"

"|| 1 : Начать ||\n"

"|| 0 : Выход ||\n"

"::=================================::\n"

"> ");

c = \_getch();

system("cls");

}

return c;

}

void main(void)

{

int Select;

while ((Select = Menu()) != '0' && Select != 27)

switch (Select)

{

case '1':

DFS\_main();

break;

}

}