Министерство высшего образования и науки Российской Федерации

Пензенский Государственный Университет

Кафедра «Вычислительная техника»

ОТЧЕТ

Лабораторная работа № 8

по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему «Определение характеристик графов»

Выполнил студент группы 20ВВ2:

Пантелеев И. А.

Приняли:

Митрохин М. А.

Юрова О. В.

Пенза 2021

**Цель работы:** Научиться искать эксцентриситеты графа, определять диаметры, радиусы и находить множества периферийных и центральных вершин.

**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.

**Листинг:**

#include <iostream>

#include <queue>

#include <vector>

#include <string>

#include <cstdlib>

using namespace std;

int i, j;

void RANDMATR(int\*\* G, const int n)

{

srand(time(NULL));

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

{

G[i][i] = 0;

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

{

if (rand() % 100 <= 50) //заполнение графа случайными числами

{

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

}

else

{

G[i][j] = G[j][i] = 0;

}

}

}

}

void OUTMATR(int\*\* G, const int n)

{

cout << "Матрица смежности графа: \n\n";

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

{

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

{

cout << " " << G[i][j];

}

cout << endl;

}

}

void DELETEMASS(int\*\* G, const int n)

{

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

delete[] G[i];

delete[] G;

}

void NKS(int\*\* G, const int n, int v, vector<vector<int>>&DIST)

{

int y = v;

queue <int> Q;

Q.push(v);

DIST[y].at(v) = 0;

while (!Q.empty())

{

v = Q.front();

Q.pop();

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

{

if ((G[v][i] != 0)&&(DIST[y].at(i) > DIST[y].at(v) + G[v][i])&&(DIST[y].at(i) == 1000))

{

Q.push(i);

DIST[y].at(i) = DIST[y].at(v) + G[v][i];

}

}

}

}

void main() {

setlocale(LC\_ALL, "Rus");

int n;

cout << "Введите размер графа: ";

cin >> n;//размер

cout << "------------------------------------------------------------\n";

vector<vector<int>>DIST;

DIST.resize(n);

vector<vector<int>>DRE;

DRE.resize(n);

vector<int>PER;

vector<int>CEN;

vector<int>IZO;

vector<int>END;

vector<int>DOM;

vector<int>STEP;

STEP.resize(n);

int R = 100;

int D = 0;

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

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

G[i] = new int[n];

RANDMATR(G, n);

OUTMATR(G, n);

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

{

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

DIST[i].push\_back(1000);

}

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

NKS(G, n, z, DIST);

cout << "------------------------------------------------------------";

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

{

cout << endl << "Расстояния от " << i + 1 << " вершины до остальных: \n";

for (j = 0; j < DIST[i].size(); j++)

{

if (DIST[i].at(j) == 1000)

DIST[i].at(j) = -1;

cout << DIST[i].at(j) << " ";

}

cout << "\n";

}

cout << "------------------------------------------------------------";

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

{

DRE[i].push\_back(0);

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

{

if (DIST[i].at(j) > DRE[i].at(0))

DRE[i].at(0) = DIST[i].at(j);

}

if (DRE[i].at(0) != 0)

{

cout << "\nЭксцентриситет " << i + 1 << "-ой вершины: " << DRE[i].at(0) << "\n";

if (D < DRE[i].at(0))

D = DRE[i].at(0);

if (R > DRE[i].at(0))

R = DRE[i].at(0);

}

else

{

cout << "\nВершина " << i + 1 << " изолированна.\n";

IZO.push\_back(i);

}

}

cout << "------------------------------------------------------------\n";

cout << "Радиус графа равен: " << R << "\n\n";

cout << "Диаметр графа равен: " << D << "\n";

cout << "------------------------------------------------------------\n";

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

{

if (DRE[i].at(0) == R)

PER.push\_back(i);

if (DRE[i].at(0) == D)

CEN.push\_back(i);

}

cout << "Множество периферийных вершин: ";

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

{

cout << PER.at(i)+1 <<" ";

}

cout << endl<<endl;

cout << "Множество центральных вершин: ";

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

{

cout << CEN.at(i)+1 << " ";

}

cout << endl;

cout << "------------------------------------------------------------\n";

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

{

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

{

if (G[i][j])

STEP.at(i)++;

}

if (STEP.at(i) != 0)

{

if (STEP.at(i) == 1)

END.push\_back(i);

else

DOM.push\_back(i);

}

}

cout << "Степени вершин: ";

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

{

cout << STEP.at(i)<<" ";

}

cout << "\n------------------------------------------------------------\n";

cout << "Множество концевых вершин: ";

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

{

cout << END.at(i) + 1 << " ";

}

cout << endl << endl;

cout << "Множество доминирующих вершин: ";

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

{

cout << DOM.at(i) + 1 << " ";

}

cout << endl << endl;

cout << "Множество изолированных вершин: ";

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

{

cout << IZO.at(i) + 1 << " ";

}

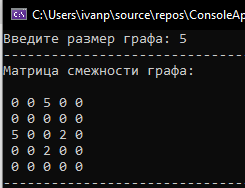
cout << "\n------------------------------------------------------------\n";

DELETEMASS(G, n);

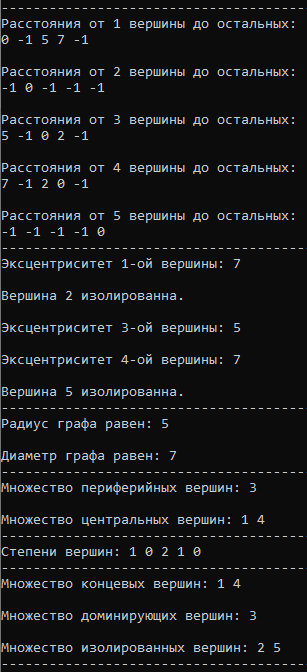
system("pause>>void");

}

**Результаты работы программы:**

****

Определяет эксцентриситеты. Ищет диаметр, радиус и находит множества периферийных и центральных вершин. Определяет количество связей у вершин.



**Вывод:** Научился искать эксцентриситеты графа, определять диаметры, радиусы и находить множества периферийных и центральных вершин. Улучшил свои навыки работы с матрицами смежности и инцидентности.