Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**Отчёт**

по лабораторной работе №8

по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему «Определение характеристик графа»

Выполнил:

 студент группы 20ВВ2

Лазутин Д.Д.

 Принял:

 к.т.н., доцент Юрова О.В.

Пенза 2020

### **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\*** (В разработке)

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.

**Описание метода решения:**

Объявлены переменные: матрица G, размер матрицы S, матрица расстояний Distance, счётчики i и j, глобальная переменная – очередь q, массив эксцентриситетов E, радиус графа R, диаметр графа D, массив степеней вершин Deg.  
Создаём и заполняем матрицу смежности, выводим её на экран. Далее применяем к каждой вершине обход в ширину посредством вызова функции BFS, таким образом формируя матрицу расстояний от каждой вершины до всех остальных.

На основе матрицы смежности составляем подмножества изолированных, доминирующих и концевых вершин. Делается это следующим образом: в каждой строке матрицы смежности проверяется наличие ребра между вершинами, и если оно есть, то степень i-й вершины Deg[i] инкрементируется. Изолированными будут те вершины, чья степень = 0. Концевыми – те, чья степень = 1. Доминирующими – те, чья степень равна кол-ву вершин графа.

На основе матрицы расстояний формируется массив эксцентриситетов через выбор максимального расстояния в каждой строке матрицы. В свою очередь, минимальный эксцентриситет будет считаться радиусом R, а вершины, чей эксцентриситет = R – центральными. Диаметром D же будет считаться максимальный эксцентриситет, а вершины с таким же эксцентриситетом – периферийными.

**Листинг** LZ\_8.cpp

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <queue>

#include <iostream>

#include <stdio.h>

#include <time.h>

#include <stdlib.h>

#include <locale.h>

#define MAXINT 32767

using namespace std;

queue <int> q;

void BFS(int v, int \*\*M, int \*Dist, int Size)

{

int i;

q.push(v); Dist[v]=0;

while (!q.empty())

{

v=q.front();

q.pop();

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

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

{

if((Dist[i]>Dist[v]+M[v][i])&&(M[v][i]>0))

{

q.push(i);

Dist[i]=Dist[v]+M[v][i];

}

}

}

}

int main()

{

int \*\*G=NULL, i=0, j=0, S=0, \*\*Distance=NULL;

int \*E=NULL, R, D, \*Deg=NULL;

setlocale(LC\_ALL, "Rus");

srand(time(NULL));

printf("Введите размерность матрицы смежности: ");

scanf("%d", &S);

while(S<=0)

{

printf("Некорректно введённое число! Введите целое положительное число: ");

scanf("%d", &S);

}

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

Distance=(int\*\*)malloc(S\*sizeof(int));

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

{

Distance[i]=(int\*)malloc(S\*sizeof(int));

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

Distance[i][j]=MAXINT;

}

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

{

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

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

{

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

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

G[i][i]=0;

}

}

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

{

printf("\n%3d|", i+1);

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

{

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

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

}

}

printf("\nПорядок обхода графа в ширину ото всех вершин (возможны повторяющиеся вершины в связи с обновлением расстояний): ");

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

{

printf("%\n%3d| ", i+1);

BFS(i, G, Distance[i], S);

}

printf("\nМатрица расстояний от i-й вершины до всех остальных: ");

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

{

printf("\n%3d|", i+1);

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

printf("%4d", Distance[i][j]);

}

Deg=(int\*)malloc(S\*sizeof(int));

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

{

Deg[i]=0;

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

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

Deg[i]++;

}

printf("\n");

printf("\nИзолированные вершины: ");

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

if(Deg[i]==0) printf("%3d", i+1);

printf("\nКонцевые вершины: ");

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

if(Deg[i]==1) printf("%3d", i+1);

printf("\nДоминирующие вершины: ");

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

if(Deg[i]==S-1) printf("%3d", i+1);

printf("\nМассив эксцетриситетов: ");

E=(int\*)malloc(S\*sizeof(int));

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

{

E[i]=Distance[i][0];

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

if(Distance[i][j]>E[i])

E[i]=Distance[i][j];

printf("%4d", E[i]);

}

for(i=0, R=E[0];i<S-1;i++)

if(E[i]<=0)

R=E[i+1];

if(R<=0) R=0;

for(i=0, D=E[0];i<S;i++)

{

if(E[i]<R && E[i]>0) R=E[i];

if(E[i]>D) D=E[i];

}

printf("\nРадиус R=%d", R);

printf("\nДиаметр D=%d", D);

printf("\nПодмножество центральных вершин:");

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

if(E[i]==R)

printf("%3d", i+1);

printf("\nПодмножество периферийных вершин:");

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

if(E[i]==D)

printf("%3d", i+1);

printf("\n");

free(E);

free(Distance);

free(G);

return 0;

}

**Результаты работы программ**

Результаты работы программы для 1-го задания представлены на рис.1-2.

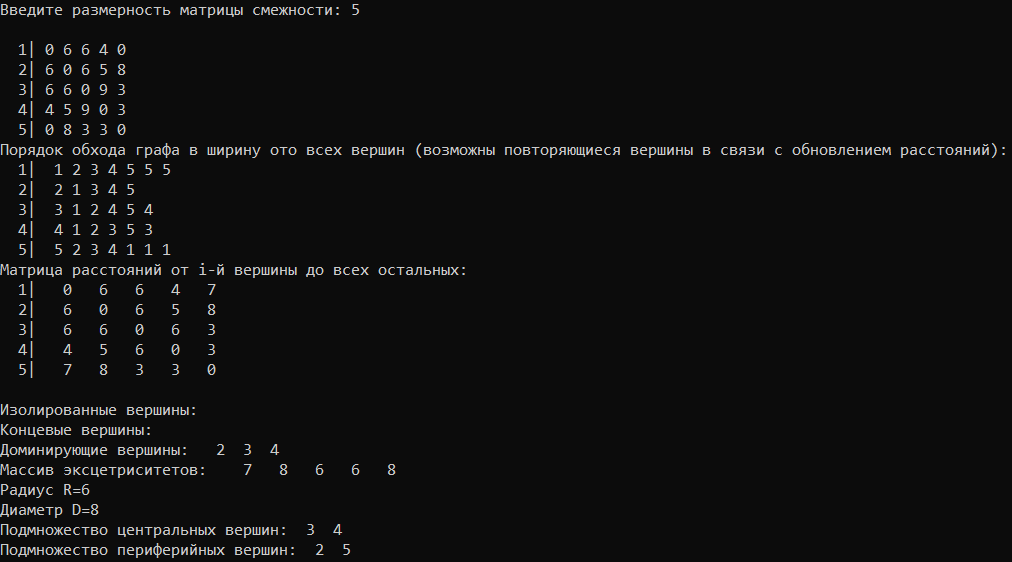


Рис.1

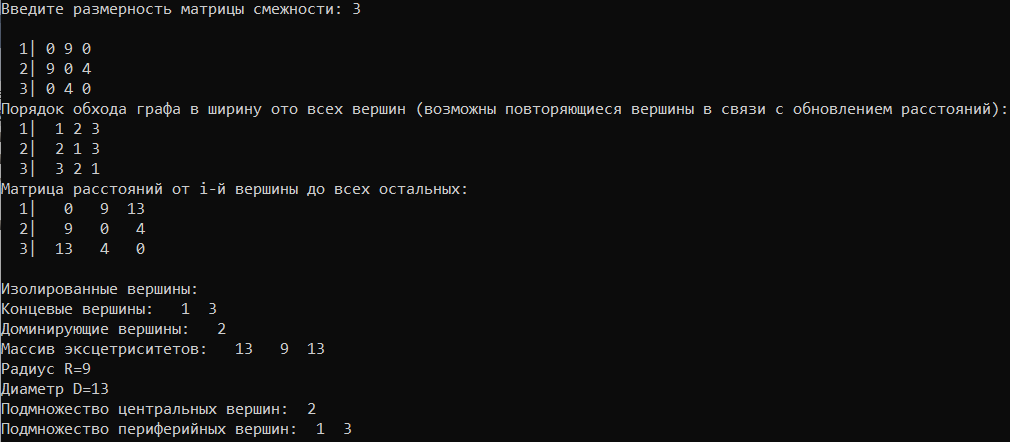


Рис.2