Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №5

по курсу “Логика и основа алгоритмизации в ИЗ”

на тему “Определение характеристик графов”

Выполнили студенты группы 21ВВ3:

Чабуев Р.П.

Костюков И.Д.

Приняли:

д.т.н., профессор Митрохин М.А.,

к.т.н., доцент Юрова О.В.

Пенза 2022

**Цель работы:**

Выполнить лабораторные указания 1-3, где необходимо сгенерировать матрицу смежности для неориентированного взвешенного графа, определить размер графа и найти изолированные, концевые, доминирующие вершины. Выполнить лабораторные указания 1-3 в задании 2\*.

**Лабораторное задание:**

*Задание 1*

1. Сгенерируйте (используя генератор случайных чисел) матрицу смежности для неориентированного взвешенного графа *G*. Выведите матрицу на экран.
2. Определите размер графа *G*, используя матрицу смежности графа.
3. Найдите изолированные, концевые и доминирующие вершины.

*Задание 2\**

1. Постройте для графа G матрицу инцидентности.
2. Определите размер графа *G*, используя матрицу смежности графа.
3. Найдите изолированные, концевые и доминирующие вершины.

**Задание 1**

**Листинг**

#include <Windows.h>

#include <time.h>

#include <stdio.h>

void podschet(int n, int\*\* G)

{

int razm = 0;

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

{

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

{

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

{

razm++;

}

}

}

printf("Размер графа = %d\n", razm);

}

void poisk(int n, int\*\* G)

{

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

{

int s = 0; // кол-во ребер с данной вершиной

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

{

if (G[i][j] == 1)

{

s++;

}

}

if (s == 0)

{

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

}

else if (s == 1)

{

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

}

else if (s == (n - 1))

{

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

}

else

{

printf("Вершина %d просто вершина\n", i + 1);

}

}

}

int main(void)

{

SetConsoleOutputCP(1251);

srand(time(NULL));

int\*\* G;

int n;

printf("Введите n = ");

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

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

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

{

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

}

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

{

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

{

G[i][j] = rand() % 2;

if (i == j)

{

G[i][j] = 0;

}

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

}

}

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

{

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

{

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

}

printf("\n");

}

podschet(n, G);

poisk(n, G);

}

**Результат работы программы:**



**Рис 1 – результат работы программы №1.**

**Задание 2**

**Листинг**

#include <Windows.h>

#include <time.h>

#include <stdio.h>

void poisk(int n, int\*\* P, int M)

{

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

{

int s = 0; // кол-во ребер с данной вершиной

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

{

if (P[j][i] == 1)

{

s++;

}

}

if (s == 0)

{

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

}

else if (s == 1)

{

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

}

else if (s == (n - 1))

{

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

}

else

{

printf("Вершина %d просто вершина\n", i + 1);

}

}

}

void in(int n, int\*\* G)

{

int M = 0;

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

{

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

{

if (G[i][j] == 1)

{

M++;

}

}

}

M /= 2;

printf("Количество рёбер (Размер графа) = %d\n", M);

int\*\* P = (int\*\*)malloc(M \* sizeof(int\*));

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

{

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

}

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

{

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

{

P[i][j] = 0;

}

}

int C = 0;

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

{

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

{

if (G[i][j])

{

P[C][i] = 1;

P[C][j] = 1;

C++;

}

}

}

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

{

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

{

printf("%d\t", P[i][j]);

}

printf("\n");

}

poisk(n, P, M);

free(P);

}

int main(void)

{

SetConsoleOutputCP(1251);

srand(time(NULL));

int n;

printf("Введите n = ");

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

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

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

{

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

}

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

{

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

{

G[i][j] = rand() % 2;

if (i == j)

{

G[i][j] = 0;

}

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

}

}

printf("Матрица смежности\n");

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

{

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

{

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

}

printf("\n");

}

printf("\n");

printf("Матрица инцидентности\n");

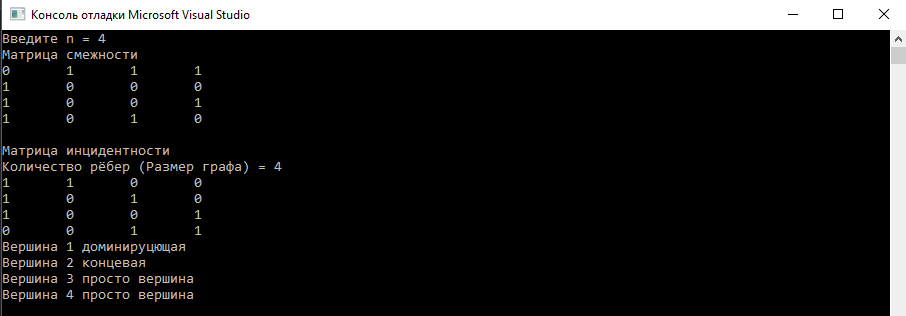
in(n, G);

free(G);

return 0;

}

**Результат работы программы:**



**Рис 2 – результат работы программы №2.**

**Вывод:**

В данной лабораторной работе в первой части задания, мы сгенерировали матицу для неориентированного взвешенного графа, а также смогли определить его, используя матрицу смежности. Выполнили задание для нахождения изолированной, концевой и доминирующей вершины.

Во второй части задания мы построили матрицу инцидентности для графа, определили его и нашли изолированные, кольцевые и доминирующие вершины.