МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
Кафедра «Вычислительной техники»

**ОТЧЁТ**

по лабораторной работе №5  
по курсу «Логика и основы алгоритмизации в инженерных задачах»  
на тему «Определение характеристик графов»

Выполнили:

Студенты группы 23ВВВ2

Лисов Е.А.

Кочегин В.В.

Приняли:

Митрохин М. А.  
Юрова О.В.

Пенза 2024

**Цель работы**

Приобрести навыки программирования и анализа графовс использованием матриц смежности и инцидентности. Генерировать случайные графыи представлять их в виде матриц. Анализировать свойства графов, такие как изолированные, концевые и доминирующие вершины. Понимать и применять различные представления графов, что является важным аспектом в теории графов и их практическом использовании.

**Задание**

**Задание 1:**

1. Сгенерируйте (используя генератор случайных чисел) матрицу смежности для неориентированного 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 инцидентности графа.
3. Найдите изолированные, концевые и доминирующие вершины.

**Ход работы**

**Задание 1.**

Количество вершин графа вводит пользователь с клавиатуры. Выделяется память под матрицу смежности, она заполняется случайными числа (0 или 1)

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

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

svyaz = rand() % 2;

matA[i][j] = svyaz;

matA[j][i] = svyaz;

}

}

Определяется размер графа

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

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

if (matA[i][j] == 1) {

cou++;

}

}

}

Находятся изолированные, концевые и доминирующие вершины графа и записываются в выделенные под них массивы:

int versh = 0;

int k = 0;

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

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

k += matA[i][j];

}

if (k == 0) {

Izol[cI++] = i;

}

else if (k == 1) {

Konc[cK++] = i;

}

else if (k == n) {

Domin[cD++] = i;

}

versh++;

k = 0;

}

Вывод матрицы происходит с помощью функции print().

**Задание 2.**

Выделяется память под матрицу инцидентности для графа. Построение матрицы инцидентности для графа происходит на основе найденных ребер в матрице смежности и ее самой:

int versh = 0;

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

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

if (matA[i][j] == 1) {

matAI[i][versh] = 1;

matAI[j][versh] = 1;

versh++;

}

}

}

Находятся изолированные, концевые и доминирующие вершины графа и записываются в выделенные под них массивы:

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

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

k += matAI[i][j];

}

if (k == 0) {

Izol[cI++] = i;

}

else if (k == 1) {

Konc[cK++] = i;

}

else if (k == n) {

Domin[cD++] = i;

}

versh++;

k = 0;

}

Вывод матрицы происходит с помощью функции print().

### Результаты работы программы

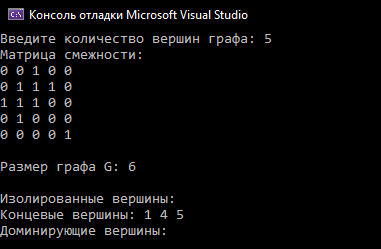


Рисунок 1 — Результаты работы программы пункта №1

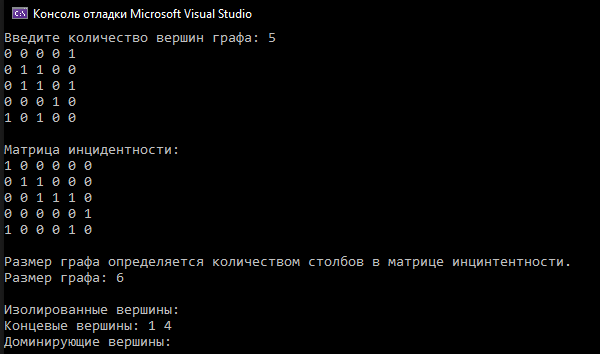


Рисунок 2 — Результаты работы программы пункта №2

### Вывод

В результате выполнения работы были успешно разработаны и реализованы функции работы c графами: создание матриц смежности и матриц инцидентности, определение размера графа по матрице, поиск изолированных, концевых и доминирующих вершин.

**Листинг**

**Задание 1:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include "stdlib.h"

#include "stdio.h"

#include "time.h"

#include "locale.h"

void print(int\*\* matA, int n, int cou) {

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

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

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

}

printf("\n");

}

}

int main() {

setlocale(LC\_ALL, "");

srand(time(NULL));

int n;

int svyaz, cou = 0;

printf("Введите количество вершин графа: ");

scanf("%d", &n);

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

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

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

}

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

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

svyaz = rand() % 2;

matA[i][j] = svyaz;

matA[j][i] = svyaz;

}

}

printf("Матрица смежности:\n");

print(matA, n, n);

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

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

if (matA[i][j] == 1) {

cou++;

}

}

}

printf("\nРазмер графа G: %d\n", cou);

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

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

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

int cI = 0, cK = 0, cD = 0;

int versh = 0;

int k = 0;

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

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

k += matA[i][j];

}

if (k == 0) {

Izol[cI++] = i;

}

else if (k == 1) {

Konc[cK++] = i;

}

else if (k == n) {

Domin[cD++] = i;

}

versh++;

k = 0;

}

printf("\nИзолированные вершины: ");

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

printf("%d ", Izol[i] + 1);

}

printf("\nКонцевые вершины: ");

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

printf("%d ", Konc[i] + 1);

}

printf("\nДоминирующие вершины: ");

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

printf("%d ", Domin[i] + 1);

}

printf("\n");

return 0;

}

**Задание 2:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <Windows.h>

#include "stdlib.h"

#include "stdio.h"

#include "time.h"

#include "locale.h"

void print(int\*\* matA, int n, int cou) {

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

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

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

}

printf("\n");

}

}

int main() {

setlocale(LC\_ALL, "");

srand(time(NULL));

int n;

int svyaz;

printf("Введите количество вершин графа: ");

scanf("%d", &n);

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

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

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

}

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

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

svyaz = rand() % 2;

matA[i][j] = svyaz;

matA[j][i] = svyaz;

}

}

print(matA, n, n);

int cou = 0;

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

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

if (matA[i][j] == 1) {

cou++;

}

}

}

if (cou == 0) {

printf("\nМатрица инцендетности не получится, ребер нет.\nВсе вершины изолированные.\n\n");

return 0;

}

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

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

matAI[i] = (int\*)malloc(cou \* sizeof(int));

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

matAI[i][j] = 0;

}

}

int versh = 0;

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

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

if (matA[i][j] == 1) {

matAI[i][versh] = 1;

matAI[j][versh] = 1;

versh++;

}

}

}

printf("\nМатрица инцидентности: \n");

print(matAI, n, cou);

printf("\nРазмер графа определяется количеством столбов в матрице инцинтентности. \nРазмер графа: %d\n", cou);

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

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

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

versh = 0;

int k = 0;

int cI = 0, cK = 0, cD = 0;

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

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

k += matAI[i][j];

}

if (k == 0) {

Izol[cI++] = i;

}

else if (k == 1) {

Konc[cK++] = i;

}

else if (k == n) {

Domin[cD++] = i;

}

versh++;

k = 0;

}

printf("\nИзолированные вершины: ");

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

printf("%d ", Izol[i] + 1);

}

printf("\nКонцевые вершины: ");

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

printf("%d ", Konc[i] + 1);

}

printf("\nДоминирующие вершины: ");

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

printf("%d ", Domin[i] + 1);

}

printf("\n");

return 0;

}