

**МИНОБРНАУКИ РОССИИ**  
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**  
**Кафедра МОЭВМ**

**ОТЧЕТ**  
**по лабораторной работе №2**  
**по дисциплине «Качество и метрология программного обеспечения»**  
**Тема: Анализ структурной сложности графовых моделей программ**

Студент гр. 8304

\_\_\_\_\_

Рыжиков А.В.

Преподаватель

\_\_\_\_\_

Ефремов М.А.

Санкт-Петербург

2022

### **Цель работы.**

Выполнить оценивание структурной сложности двух программ с помощью критериев:

- Минимального покрытия вершин и дуг графа управления;
- Выбора маршрутов на основе цикломатического числа графа.

Варианты программ:

- Программа с заданной структурой управляющего графа, выбираемой из файла `zadan_struct.doc` в соответствии с номером в списке группы;
- Программа из 1-ой лабораторной работы (управляющий граф составить самостоятельно).

Оцениваемые характеристики структурной сложности:

- Число учитываемых маршрутов проверки программы для заданного критерия;
- Цикломатическое число;
- Суммарное число ветвлений по всем маршрутам – оценка структурной сложности;

### **Ход выполнения. Вариант 15**

1. Было проведено оценивание структурной сложности программы с заданной структурой управляющего графа — см. рис. 1.

Вычисление сложности программы по критерию минимального покрытия вершин и дуг графа управления:

m1: 1 – 3 – 14 – 15	S1=2
m2: 1 – 3 – 6 – 9 – 12 – 14 – 15	S2=3
m3: 1 – 2 – 5 – 7 – 9 – 12 – 9 – 12 – 14 – 15	S3=5
m4: 1 – 2 – 5 – 8 – 10 – 12 – 14 – 15	S4=6
m5: 1 – 2 – 5 – 8 – 10 – 13 – 14 – 15	S5=5
m6: 1 – 2 – 5 – 8 – 11 – 13 – 14 – 15	S6=4
m7: 1 – 2 – 4 – 6 – 9 – 12 – 14 – 15	S7=3
	S=28

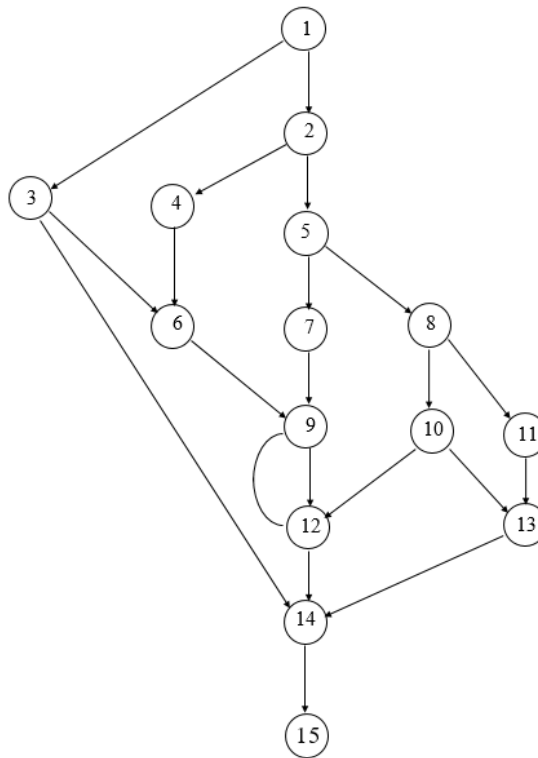


Рисунок 1 — Управляющий граф программы в соответствии с вариантом

Вычисление сложности программы с выбором маршрутов на основе цикломатического числа графа. Цикломатическое число  $Z$  исходного графа программы определяется формулой  $Z = Y - N + 2 \times \Omega$ , где  $Y$  – общее число дуг в графе;  $N$  – общее число вершин в графе;  $\Omega$  – число связных компонент графа, число связных компонент графа  $\Omega$  равно количеству дуг, необходимых для превращения исходного графа в максимально связный граф.:

$$\text{Цикломатическое число } Z = 21 - 15 + 2 \times 1 = 7$$

m1: 1 – 3 – 14 – 15	S1=2
m2: 1 – 3 – 6 – 9 – 12 – 14 – 15	S2=3
m3: 1 – 2 – 5 – 7 – 9 – 12 – 9 – 12 – 14 – 15	S3=5
m4: 1 – 2 – 5 – 8 – 10 – 12 – 14 – 15	S4=6
m5: 1 – 2 – 5 – 8 – 10 – 13 – 14 – 15	S5=5
m6: 1 – 2 – 5 – 8 – 11 – 13 – 14 – 15	S6=4
m7: 1 – 2 – 4 – 6 – 9 – 12 – 14 – 15	S7=3
	S=28

Также было проведено оценивание структурной сложности заданной программы с помощью программы ways (вход программы см. в приложении А) — см. рис 2 и 3.

```
----- Path #1 -----
-> 1 -> 3 -> 6 -> 9 -> 12 -> 9 -> 12 -> 14 -> 15
-----Press a key to continue -----
----- Path #2 -----
-> 1 -> 2 -> 4 -> 6 -> 9 -> 12 -> 14 -> 15
-----Press a key to continue -----
----- Path #3 -----
-> 1 -> 3 -> 14 -> 15
-----Press a key to continue -----
----- Path #4 -----
-> 1 -> 2 -> 5 -> 7 -> 9 -> 12 -> 14 -> 15
-----Press a key to continue -----
----- Path #5 -----
-> 1 -> 2 -> 5 -> 8 -> 10 -> 13 -> 14 -> 15
-----Press a key to continue -----
----- Path #6 -----
-> 1 -> 2 -> 5 -> 8 -> 11 -> 13 -> 14 -> 15
-----Press a key to continue -----
----- Path #7 -----
-> 1 -> 2 -> 5 -> 8 -> 10 -> 12 -> 14 -> 15
-----Press a key to continue -----

Complexity = 28
Press a key...
```

Рисунок 2 — Результат программы ways по критерию минимального покрытия

```
DOSBox 0.73, Cpu Cycles: 3000, Frameskip 0, Program: WAYS-1
----- Path #1 -----
-> 1 -> 3 -> 6 -> 9 -> 12 -> 14 -> 15
-----Press a key to continue -----
----- Path #2 -----
-> 1 -> 3 -> 14 -> 15
-----Press a key to continue -----
----- Path #3 -----
-> 1 -> 2 -> 4 -> 6 -> 9 -> 12 -> 14 -> 15
-----Press a key to continue -----
----- Path #4 -----
-> 1 -> 2 -> 5 -> 7 -> 9 -> 12 -> 14 -> 15
-----Press a key to continue -----
----- Path #5 -----
-> 1 -> 2 -> 5 -> 8 -> 10 -> 13 -> 14 -> 15
-----Press a key to continue -----
----- Path #6 -----
-> 1 -> 2 -> 5 -> 8 -> 10 -> 12 -> 14 -> 15
-----Press a key to continue -----
----- Path #7 -----
-> 1 -> 2 -> 5 -> 8 -> 11 -> 13 -> 14 -> 15
-----Press a key to continue -----

Complexity = 28
Press a key...
```

Рисунок 3 — Результат программы ways по путям по цикломатическому числу

Результат программы ways по критерию минимального покрытия (рис. 2) совпал с ручным расчетом. Результат программы по путям по цикломатическому числу (рис. 3) совпал с ручным расчетом.

2. Для программы из 1-ой лабораторной работы был составлен управляющий граф — см. рис. 4 (см. код программы в приложении В с комментариями, отражающими соответствие операторов и вершин).

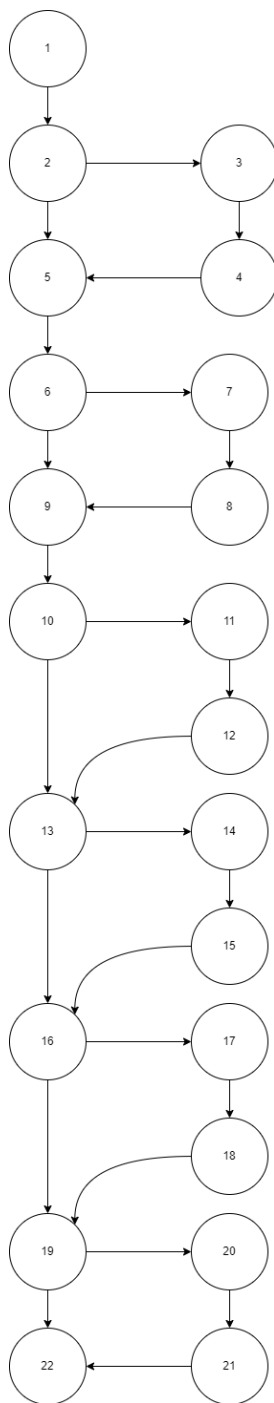


Рисунок 4 — Управляющий граф программы из лаб. работы 1

Вычисление сложности программы по критерию минимального покрытия вершин и дуг графа управления:

m1: 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 13 – 14 – 15 – 16 – 17 – 18 – 19 – 20 – 21 – 22      S1=6

m2: 1 – 2 – 5 – 6 – 9 – 10 – 13 – 16 – 19 – 22      S2=6

S=12

Вычисление сложности программы с выбором маршрутов на основе цикломатического числа графа:

Цикломатическое число  $Z = 27 - 22 + 2 \times 1 = 7$

m1: 1 – 2 – 3 – 4 – 5 – 6 – 9 – 10 – 13 – 16 – 19 – 22      S1=6

m2: 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 13 – 16 – 19 – 22      S2=6

m3: 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 13 – 16 – 19 – 22      S3=6

m4: 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 13 – 14 – 15 – 16 – 19 – 22      S4=6

m5: 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – 10 – 11 – 12 – 13 – 14 – 15 – 16 – 17 – 18 – 19 – 22      S5=6

m6: 1 – 2 – 3 – 4 – 5 – 6 – 9 – 10 – 11 – 12 – 13 – 14 – 15 – 16 – 17 – 18 – 19 – 20 – 21 – 22      S6=6

m7: 1 – 2 – 5 – 6 – 9 – 10 – 13 – 16 – 19 – 22      S7=6

S=42

Также было проведено оценивание структурной сложности заданной программы с помощью программы ways (вход программы см. в приложении С) — см. рис 5, 6 и 7.

```
DOSBox 0.73, Cpu Cycles: 3000, Frameskip 0, Program: WAYS~1
Min ways....
----- Path #1 -----
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> 11 -> 12 -> 13 -> 14 -> 1
5 -> 16 -> 17 -> 18 -> 19 -> 20 -> 21 -> 22
-----Press a key to continue -----
----- Path #2 -----
-> 1 -> 2 -> 5 -> 6 -> 9 -> 10 -> 13 -> 16 -> 19 -> 22
-----Press a key to continue -----

Complexity = 12
Press a key...
```

Рисунок 5 — Результат программы ways по критерию минимального покрытия

```
Z ways....
----- Path #1 -----
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> 11 -> 12 -> 13 -> 14 -> 1
5 -> 16 -> 17 -> 18 -> 19 -> 20 -> 21 -> 22
-----Press a key to continue -----
----- Path #2 -----
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> 11 -> 12 -> 13 -> 14 -> 1
5 -> 16 -> 17 -> 18 -> 19 -> 22
-----Press a key to continue -----
----- Path #3 -----
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> 11 -> 12 -> 13 -> 14 -> 1
5 -> 16 -> 19 -> 22
-----Press a key to continue -----
----- Path #4 -----
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> 11 -> 12 -> 13 -> 16 -> 1
9 -> 22
-----Press a key to continue -----
----- Path #5 -----
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> 13 -> 16 -> 19 -> 22
-----Press a key to continue -----
----- Path #6 -----
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 9 -> 10 -> 13 -> 16 -> 19 -> 22
-----Press a key to continue -----
```

Рисунок 6 — Результат программы ways по путям по цикломатическому числу

```
DOSBox 0.73, Cpu Cycles: 3000, Frameskip 0, Program: DOSBOX
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> 11 -> 12 -> 13 -> 14 -> 1
5 -> 16 -> 17 -> 18 -> 19 -> 22
-----Press a key to continue -----
----- Path #3 -----
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> 11 -> 12 -> 13 -> 14 -> 1
5 -> 16 -> 19 -> 22
-----Press a key to continue -----
----- Path #4 -----
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> 11 -> 12 -> 13 -> 16 -> 1
9 -> 22
-----Press a key to continue -----
----- Path #5 -----
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> 13 -> 16 -> 19 -> 22
-----Press a key to continue -----
----- Path #6 -----
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 9 -> 10 -> 13 -> 16 -> 19 -> 22
-----Press a key to continue -----
----- Path #7 -----
-> 1 -> 2 -> 5 -> 6 -> 9 -> 10 -> 13 -> 16 -> 19 -> 22
-----Press a key to continue -----

Complexity = 42
Press a key...
```

Рисунок 7 — Результат программы ways по путям по цикломатическому числу

### **Выводы.**

В результате выполнения данной лабораторной работы была оценена структурная сложности двух программ с помощью критериев - минимального покрытия вершин и дуг графа управления; и выбора маршрутов на основе цикломатического числа графа.



## ПРИЛОЖЕНИЕ А. УПРАВЛЯЮЩИЙ ГРАФ ПРОГРАММЫ В СООТВЕТСТВИИ С ВАРИАНТОМ

Nodes{ 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15}

Top{ 1}

Last{ 15}

Arcs{

arc(1,3);

arc(1,2);

arc(2,4);

arc(4,6);

arc(3,6);

arc(3,14);

arc(6,9);

arc(2,5);

arc(5,7);

arc(5,8);

arc(8,10);

arc(8,11);

arc(10,13);

arc(11,13);

arc(10,12);

arc(7,9);

arc(9,12);

arc(12,9);

arc(12,14);

arc(13,14);

arc(14,15);

}

## ПРИЛОЖЕНИЕ В. ПРОГРАММА НА ЯЗЫКЕ СИ ИЗ ЛАБ. РАБОТЫ 1

```
#include <stdio.h>
#include <stdlib.h>

int n = 5;

void linfit2(float x[n], float y[n], float *y_calc) {

    float sum_x = 0, sum_y = 0, sum_xy = 0,
          sum_x2 = 0, sum_y2 = 0;
    float xi, yi, sxy, sxx, syy;
    for (int i = 0; i < n; ++i) {

        xi = x[i];
        yi = y[i];
        sum_x += xi;
        sum_y += yi;
        sum_xy += xi * yi;
        sum_x2 += xi * xi;
        sum_y2 += yi * yi;
    }

    sxx = sum_x2 - sum_x * sum_x / n;
    sxy = sum_xy - sum_x * sum_y / n;
    syy = sum_y2 - sum_y * sum_y / n;

    float b = sxy / sxx;
    float a = (sum_x2 * sum_y - sum_x * sum_xy) / n / sxx;
    for (int i = 0; i < n; ++i) {
        *(y_calc + i) = a + b * x[i];
    }
}

int main() {
    float x[5];
    float y[5];
    float y_calc[5];

    for (int i = 0; i < n; i++) {
        x[i] = (float) (rand() % 100) / 100;
        y[i] = (float) (rand() % 100) / 100;
    }

    linfit2(x, y, &y_calc);

    printf("x: ");

    for (int i = 0; i < n; ++i) {
        printf("%f ;", x[i]);
    }

    printf("\ny: ");

    for (int i = 0; i < n; ++i) {
        printf("%f ;", y[i]);
    }

    printf("\ny_calc: ");

    for (int i = 0; i < n; ++i) {
        printf("%f ;", y_calc[i]);
    }
}
```

## **ПРИЛОЖЕНИЕ С. УПРАВЛЯЮЩИЙ ГРАФ ПРОГРАММЫ ИЗ ЛАБ. РАБОТЫ 1**

```
Nodes{ 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22}  
Top{ 1}  
Last{ 22}  
Arcs{  
  arc(1,2);  
  arc(2,3);  
  arc(3,4);  
  arc(4,5);  
  arc(2,5);  
  arc(5,6);  
  arc(6,7);  
  arc(7,8);  
  arc(8,9);  
  arc(6,9);  
  arc(9,10);  
  arc(10,11);  
  arc(11,12);  
  arc(12,13);  
  arc(10,13);  
  arc(13,14);  
  arc(14,15);  
  arc(15,16);  
  arc(13,16);  
  arc(16,17);  
  arc(17,18);  
  arc(18,19);  
  arc(16,19);  
  arc(19,20);  
  arc(20,21);  
  arc(21,22);  
  arc(19,22);  
}
```