

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

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

Студент гр. 7304

\_\_\_\_\_

Абдульманов Э.М

Преподаватель

\_\_\_\_\_

Ефремов М.А.

Санкт-Петербург

2021

**Цель работы:**

Изучение структурной сложности графовых моделей программ и метрик ее оценки.

**Задание:**

Выполнить оценивание структурной сложности двух программ с помощью критериев:

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

Варианты программ:

- Программа с заданной преподавателем 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).

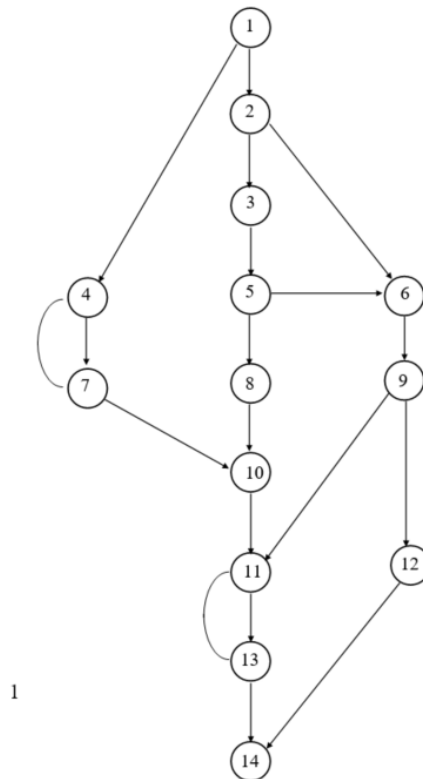
Оцениваемые характеристики структурной сложности:

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

## Ход работы

1. Был выбран вариант №1 для первой программы, у которой необходимо определить структурную сложность. Управляющий граф первой программы приведен на Рисунке 1:

ВАРИАНТ



1

Рисунок 1: Управляющий граф первой программы для определения структурной сложности

2. Была рассчитана структурная сложность первой программы вручную по первому критерию, а именно по минимальному покрытию вершин и дуг графа управления.

M1: **1-2-3-5-6-9-11-13-11-13-14** (6 ветвлений)

M2: **1-2-6-9-12-14** (3 ветвления)

M3: **1-4-7-4-7-10-11-13-14** (4 ветвления)

M4: **1-2-3-5-8-10-11-13-14** (4 ветвления)

Количество маршрутов:  $M = 4$

Сложность:  $S = 6 + 3 + 4 + 4 = 17$

3. Было рассчитано вручную цикломатическое число графа первой программы для дальнейшего применения второго критерия. Для данной программы число вершин в графе равно 14, число дуг – 19, число связных компонент графа 1, тогда:  $Z = Y - N + 2 * P = 19 - 14 + 2 * 1 = 7$ , то есть цикломатическое число равно 7.
4. Для управляющего графа первой программы вручную были построены 8 линейно-независимых циклов и линейно-независимых маршрутов, после чего была подсчитана структурная сложность по второму критерию.

M1: **1-2-3-5-6-9-11-13-14** (5 ветвлений)

M2: **1-2-3-5-6-9-12-14** (4 ветвления)

M3: **1-2-6-9-12-14** (3 ветвления)

M4: **1-4-7-10-11-13-14** (3 ветвления)

M5: **1-2-3-5-8-10-11-13-14** (4 ветвления)

M6: **11-13-11** (1 ветвление)

M7: **4-7-4** (1 ветвление)

Сложность:  $S = 5 + 4 + 3 + 3 + 4 + 1 + 1 = 21$

5. Для первой программы была подсчитана структурная сложность по двум критериям с помощью программы ways.exe. Результаты расчета структурной сложности по первому критерию приведены на Рисунке 2, по второму критерию на Рисунке 3:

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

Complexity = 17
Press a key...
```

Рисунок 2: Программный расчет структурной сложности первой программы по первому критерию

```

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

Complexity = 21
Press a key...

```

Рисунок 3: Программный расчет структурной сложности первой программы по второму критерию

6. Для программы из первой лабораторной работы был составлен управляющий граф. Данный управляющий граф представлен на Рисунке 4:

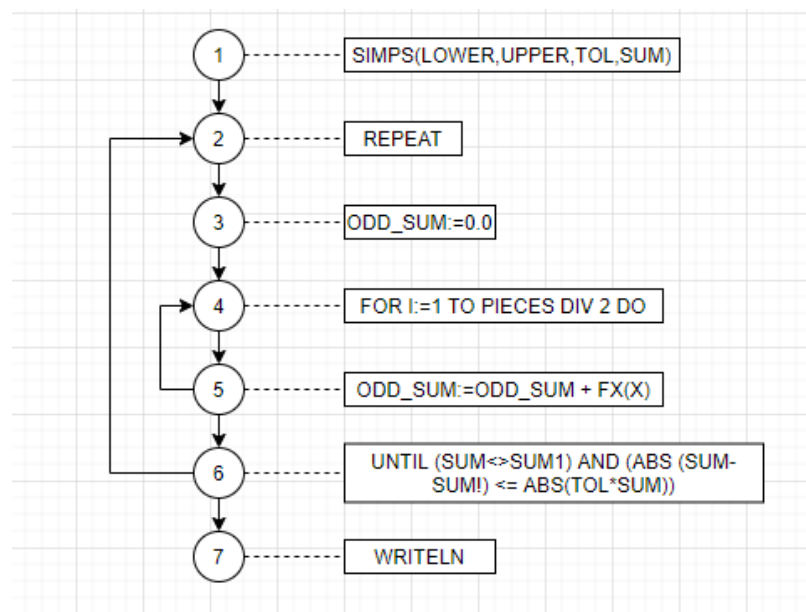


Рисунок 4: Управляющий граф программы из первой лабораторной работы для определения структурной сложности

7. Была рассчитана структурная сложность программы из первой лабораторной работы вручную по первому критерию, а именно по минимальному покрытию вершин и дуг графа управления.

M1: 1-2-3-4-5-4-5-6-2-3-4-5-6-7 (5 ветвлений)

Количество маршрутов:  $M = 1$

Сложность:  $S = 5$

8. Было рассчитано вручную цикломатическое число графа программы из первой лабораторной работы для дальнейшего применения второго критерия. Для данной программы число вершин в графе равно 7, число дуг – 8, число связных компонент графа 1, тогда:  $Z = Y - N + 2 * P = 8 - 7 + 2 * 1 = 3$ , то есть цикломатическое число равно 3.

9. Для управляющего графа программы из первой лабораторной работы вручную были построены 3 линейно-независимых цикла и линейно – независимых маршрута, после чего была подсчитана структурная сложность по второму критерию.

M1: 1-2-3-4-5-6-7 (2 ветвления)

M2: 4-5-4 (1 ветвление)

M3: 2-3-4-5-6-2 (2 ветвления)

Сложность:  $S = 2 + 1 + 2 = 5$

10. Для программы из первой лабораторной работы была подсчитана структурная сложность по двум критериям с помощью программы ways.exe. Результаты расчета структурной сложности по первому критерию приведены на Рисунке 5, по второму критерию на Рисунке 6.

```
Min ways.... Path #1 -----
-> 1 -> 2 -> 3 -> 4 -> 5 -> 4 -> 5 -> 6 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7
-----Press a key to continue -----
Complexity = 5
Press a key...
```

Рисунок 5: Программный расчет структурной сложности программы из первой лабораторной работы по первому критерию

```

Z ways....
----- Path #1 -----
-> 4 -> 5 -> 4
-----Press a key to continue -----
----- Path #2 -----
-> 2 -> 3 -> 4 -> 5 -> 6 -> 2
-----Press a key to continue -----
----- Path #1 -----
-> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7
-----Press a key to continue -----

Complexity = 5
Press a key...

```

Рисунок 6: Программный расчет структурной сложности программы из первой лабораторной работы по второму критерию

### **Выводы.**

В ходе выполнения лабораторной работы были изучены методы оценки структурной сложности программы на основе его управляющего графа, была рассчитана структурная сложность двух программ по двум критериям: минимальное покрытие дуг графа и выбор маршрутов на основе цикломатического числа графа. Расчеты были проведены как ручным, так и программным способом.



## ПРИЛОЖЕНИЕ А.

### КОД ПРОГРАММЫ НА ЯЗЫКЕ ПАСКАЛЬ.

```
PROGRAM SIMP1;
{ INTEGRATION BY SIMPSON'S METHOD }
CONST TOL = 1.0E-6;
VAR SUM,UPPER,LOWER      : REAL;
FUNCTION FX(X: REAL): REAL;
BEGIN
    FX:=EXP(-X/2)
END;
{ FUNCTION FX }
FUNCTION DFX(X: REAL): REAL;
BEGIN
    DFX:=- (EXP(-X/2)) / 2
END;
{ FUNCTION DFX }
PROCEDURE SIMPS(LOWER,UPPER,TOL      : REAL; VAR SUM      : REAL);
{ NUMERICAL INTEGRATION BY SIMPSON'S RULE }
{ FUNCTION IS FX, LIMITS ARE LOWER AND UPPER }
{ WITH NUMBER OF REGIONS EQUAL TO PIECES }
{ PARTITION IS DELTA_X, ANSWER IS SUM }
VAR I : INTEGER;
X,DELTA_X,EVEN_SUM, ODD_SUM,END_SUM, END_COR,SUM1 : REAL;
PIECES      : INTEGER;
BEGIN
    PIECES:=2;
    DELTA_X:=(UPPER-LOWER)/PIECES;
    ODD_SUM:=FX(LOWER+DELTA_X);
    EVEN_SUM:=0.0;
    END_SUM:=FX(LOWER)+FX(UPPER);
    END_COR:=DFX(LOWER)-DFX(UPPER);
    SUM:=(END_SUM+4.0*ODD_SUM)*DELTA_X/3.0;
    REPEAT
        PIECES:=PIECES*2;
        SUM1:=SUM;
        DELTA_X:=(UPPER-LOWER)/PIECES;
        EVEN_SUM:=EVEN_SUM+ODD_SUM;
        ODD_SUM:=0.0;
        FOR I:=1 TO PIECES DIV 2 DO
            BEGIN
                X:=LOWER+DELTA_X*(2.0*I-1.0);
                ODD_SUM:=ODD_SUM+FX(X)
            END;
        SUM:=(7.0*END_SUM+14.0*EVEN_SUM+16.00*ODD_SUM+END_COR*DELTA_X)*DELTA_X/15.0;
        UNTIL (SUM<>SUM1) AND (ABS(SUM-SUM1)<=ABS(TOL*SUM))
    END; { SIMPS }
BEGIN { MAIN PROGRAM }
    LOWER:=1.0;
    UPPER:=9.0;
    SIMPS(LOWER,UPPER,TOL,SUM);
    WRITELN;
END.
```