

**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**

Студентка гр. 7304

\_\_\_\_\_

Каляева А.В.

Преподаватель

\_\_\_\_\_

Ефремов М.А.

Санкт-Петербург

2021

**Задание:**

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 работы:

#### 1. Оценивание структурной сложности для программы с заданной структурой управляющего графа

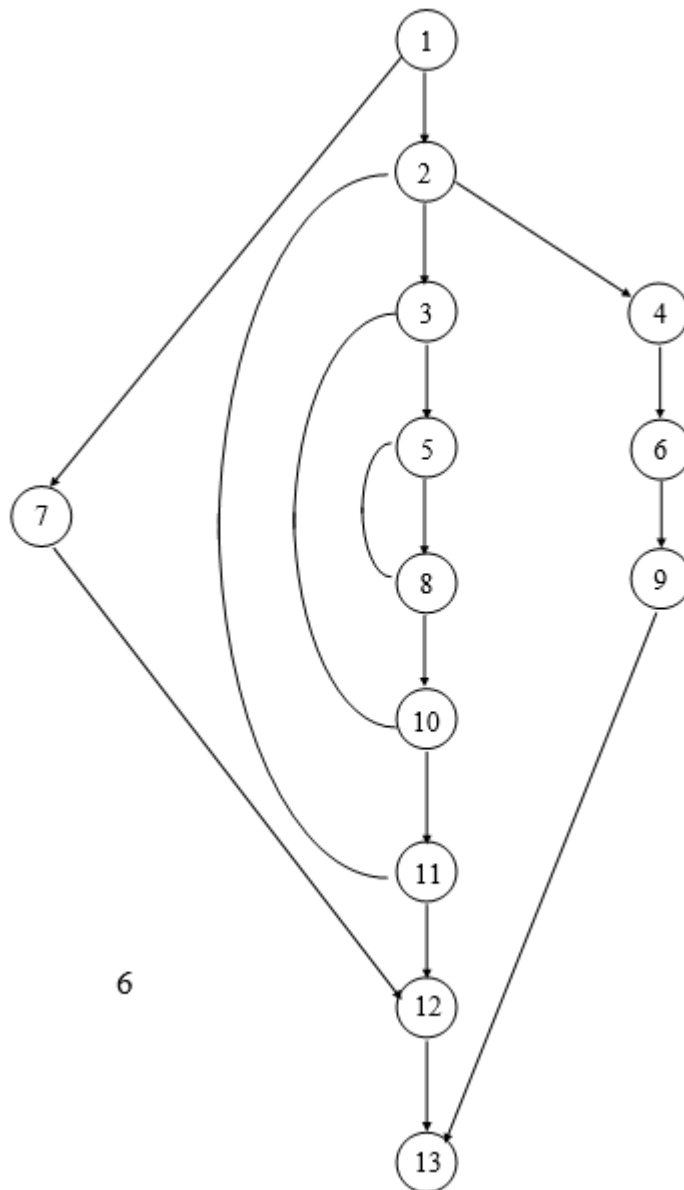


Рисунок 1 – Граф выполнения программы

- Ненаправленная дуга 2-11 была удалена из графа, так как при ее наличии программа ways.exe сообщала о некорректной структуре графа.
- Ненаправленная дуга 5-8 ориентирована из 8 в 5, так как в противном случае она повторяет уже имеющуюся в графе дугу.
- Ненаправленная дуга 3-10 ориентирована из 10 в 3.

## 1.1.Оценивание структурной сложности первой программы с помощью критерия минимального покрытия дуг графа

### 1.1.1. Ручной расчет

Ветвления в вершинах: 1, 2, 8, 10

Минимальный набор путей:

M1: 1-7-12-13

M2: 1-2-4-6-9-13

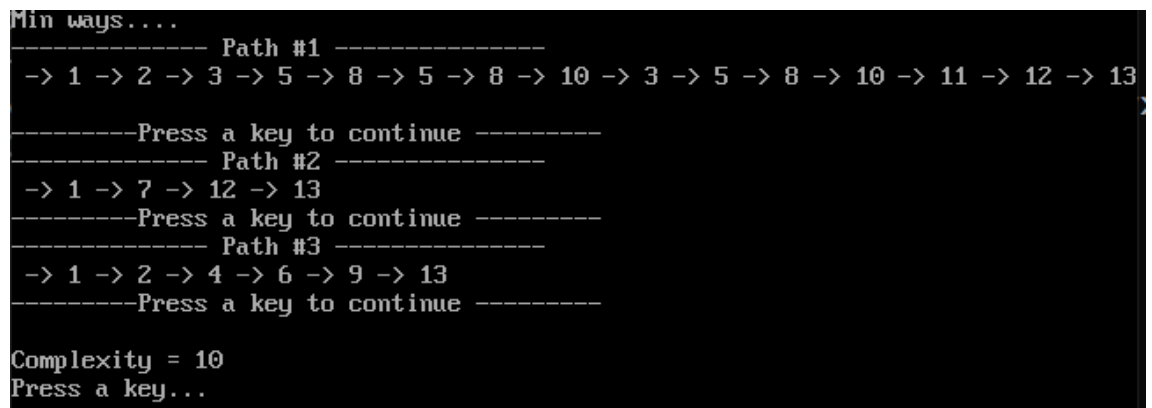
M3: 1-2-3-5-8-5-8-10-3-5-8-10-11-12-13

Итого: 10

### 1.1.2. Программный расчет

Запись графа для запуска программы ways.exe представлена в приложении А.

Результат запуска программы представлен на рисунке 2.



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

Complexity = 10
Press a key...
```

Рисунок 2 – Результат работы программы

### 1.1.3. Сравнение результатов

Анализируя результаты построения маршрутов и расчета сложности вручную и программно, можно заметить, что построенные маршруты, а также рассчитанная сложность совпадают.

## 1.2.Оценивание структурной сложности первой программы с помощью критерия на основе цикломатического числа

### 1.2.1. Ручной расчет

Количество ребер: 16

Количество вершин: 13

Цикломатическое число:  $16-13+2*1=5$

Ветвления в вершинах: 1, 2, 8, 10

Набор путей:

M1: 1-7-12-13

M2: 5-8-5

M3: 3-5-8-10-3

M4: 1-2-4-6-9-13

M5: 1-2-3-5-8-10-11-12-13

Итого: 10

### 1.2.2. Программный расчет

Результат запуска программы представлен на рисунке 3.

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

Complexity = 10
Press a key...
```

Рисунок 3 – Результат работы программы

### 1.2.3. Сравнение результатов

Анализируя результаты построения маршрутов и расчета сложности вручную и программно, можно заметить, что построенные маршруты, а также рассчитанная сложность совпадают.

## 2. Оценивание структурной сложности для программы из 1-ой лабораторной работы

Исходный код программы, разработанной в 1-ой лабораторной работе представлен в приложении Б.

Граф, построенный для программы из 1-ой лабораторной работы представлен на рисунке 4.



Рисунок 4 – Граф выполнения программы

## 2.1.Оценивание структурной сложности второй программы с помощью критерия минимального покрытия дуг графа

### 2.1.1. Ручной расчет

Ветвления в вершинах: 5, 10, 13

Минимальный набор путей:

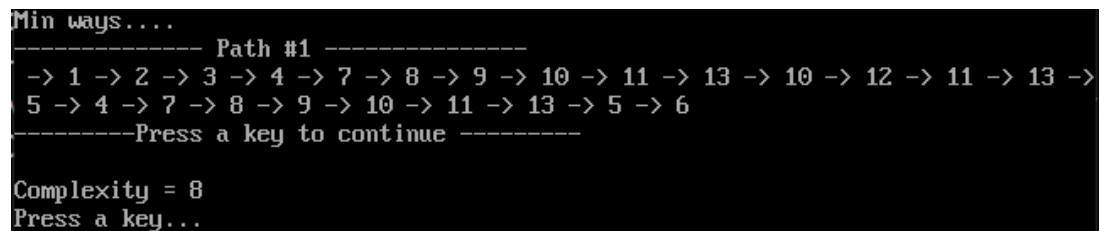
M1: 1-2-3-4-7-8-9-10-11-13-5-4-7-8-9-10-12-11-13-10-11-13-5-6

Итого: 8

### 2.1.2. Программный расчет

Запись графа для запуска программы ways.exe представлена в приложении В.

Результат запуска программы представлен на рисунке 5.



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

Рисунок 5 – Результат работы программы

### 2.1.3. Сравнение результатов

Маршрут, построенный программой немного отличается от маршрута, построенного вручную. Результаты расчета сложности ручным и программным способом совпадают. Таким образом, сложность программы по критерию минимального покрытия дуг не зависит от выбора маршрута для расчетов.

## 2.2.Оценивание структурной сложности второй программы с помощью критерия на основе цикломатического числа

### 2.2.1. Ручной расчет

Количество ребер: 15

Количество вершин: 13

Цикломатическое число:  $15-13+2*1=4$

Ветвления в вершинах: 5, 10, 13

Набор путей:

M1: 10-11-13-10

M2: 4-7-8-9-10-11-13-5-4

M3: 1-2-3-4-7-8-9-10-12-11-13-5-6

M4: 1-2-3-4-7-8-9-10-11-13-5-6

Итого: 12

### 2.2.2. Программный расчет

Результат запуска программы представлен на рисунке 6.

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

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

Рисунок 6 – Результат работы программы

### 2.2.3. Сравнение результатов

Анализируя результаты построения маршрутов и расчета сложности вручную и программно, можно заметить, что построенные маршруты, а также рассчитанная сложность совпадают.



**Выводы:**

В ходе выполнения данной работы были изучены критерии оценивания структурной сложности программ. Была проведена оценка структурной сложности двух программ: программы, соответствующей заданному варианту и программы, разработанной в первой лабораторной работе.

**ПРИЛОЖЕНИЕ А.**  
**КОД ПРОГРАММЫ НА ЯЗЫКЕ ПАСКАЛЬ.**

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

```
Top{1}  
Last{13}
```

```
Arcs{  
arc(1,2);  
arc(1,7);  
arc(2,3);  
arc(2,4);  
arc(3,5);  
arc(4,6);  
arc(5,8);  
arc(6,9);  
arc(7,12);  
arc(8,5);  
arc(8,10);  
arc(9,13);  
arc(10,3);  
arc(10,11);  
arc(11,12);  
arc(12,13);  
}
```

**ПРИЛОЖЕНИЕ Б.**  
**КОД ПРОГРАММЫ НА ЯЗЫКЕ СИ.**

```
void swap(float* p, float* q) {
    float hold = *p;
    *p = *q;
    *q = hold;
}

float* sort(float* a, int n) {
    int i;
    int done;
    int jump = n;
    while (jump > 0) {
        jump = jump / 2;
        do {
            done = 1;
            for (int j = 0; j < n; j++) {
                i = j + jump;
                if ((n>i) && (a[j] > a[i])) {
                    swap(&a[i], &a[j]);
                    done = 0;
                }
            }
        } while(!done);
    }

    return a;
}
```

**ПРИЛОЖЕНИЕ В.**  
**КОД ПРОГРАММЫ НА ЯЗЫКЕ АССЕМБЛЕР.**

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

```
Top{1}  
Last{6}
```

```
Arcs{  
arc(1,2);  
arc(2,3);  
arc(3,4);  
arc(4,7);  
arc(5,4);  
arc(5,6);  
arc(7,8);  
arc(8,9);  
arc(9,10);  
arc(10,11);  
arc(10,12);  
arc(11,13);  
arc(12,11);  
arc(13,5);  
arc(13,10);  
}
```