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. 8304 _____ Щука А. А.

Преподаватель _____ Кирьянчиков В. А.

Санкт-Петербург
2022

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 с заданной структурой управляющего графа, выбираемой из файла zadan_struct.doc в соответствии с номером в списке группы (рисунок 1). Граф был изменен, чтобы его принимала программа, ребро 5-6 заменено на ребро 3-6;
- 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.

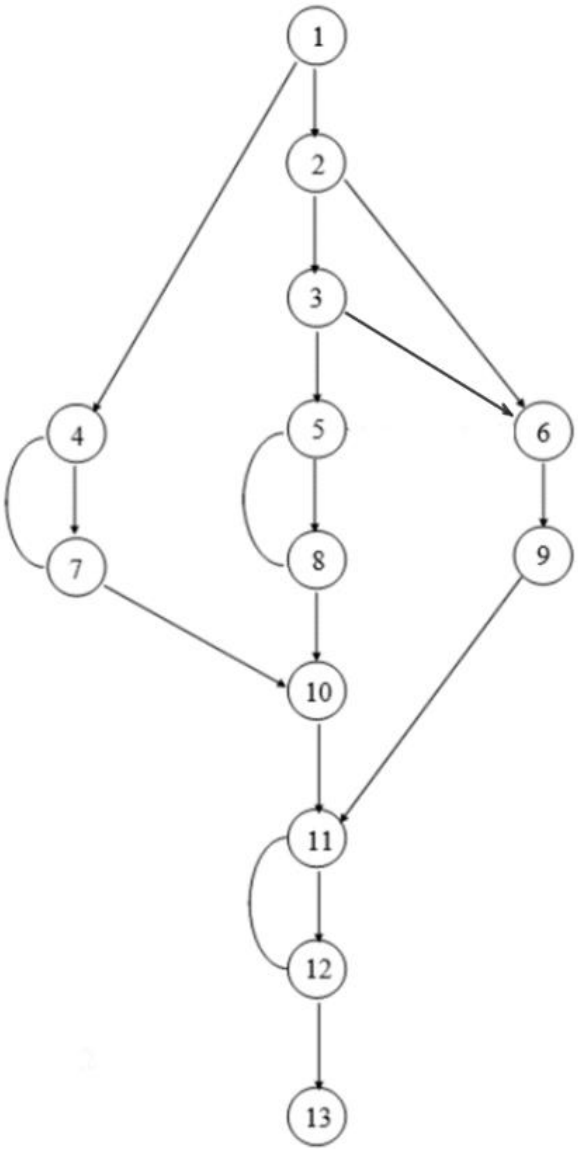

Рисунок 1 – Граф из файла zadan_struct.doc

Оценка структурной сложности программы из файла zadan_struct.doc.

Оценивание структурной сложности с помощью критерия минимального покрытия дуг графа **вручную**.

Ветвления: 1, 2, 3, 7, 8, 12. Всего ветвлений – 6.

Минимальный набор путей (жирным выделены ветвления):

- **1**-4-**7**-4-**7**-10-11-**12**-13 (4 ветвлений);
- **1**-**2**-**3**-5-**8**-5-**8**-10-11-**12**-11-**12**-13 (7 ветвлений);
- **1**-**2**-6-9-11-**12**-13 (3 ветвления);
- **1**-**2**-**3**-6-9-11-**12**-13 (4 ветвления).

Итого сложность равна 4 + 7 + 3 + 4 = 18.

Оценивание структурной сложности с помощью критерия на основе цикломатического числа **вручную**.

Число вершин в графе – 13, число ребер – 18. Для того, чтобы граф стал связным (из каждой вершины существовал путь в любую другую) достаточно добавить одно ребро (13-1). Таким образом, цикломатическое число графа равно $18 - 13 + 2 * 1 = 7$. Значит необходимо рассмотреть 7 линейно-независимых циклов и путей.

- 4-**7**-4 (1 ветвление);
- 5-**8**-5 (1 ветвление);
- 11-**12**-11 (1 ветвление);
- **1**-**2**-**3**-6-9-11-**12**-13 (4 ветвления);
- **1**-**2**-**3**-5-**8**-10-11-**12**-13 (5 ветвлений);
- **1**-4-**7**-10-11-**12**-13 (3 ветвления)
- **1**-**2**-6-9-11-**12**-13 (3 ветвления).

Итого сложность равна 4 + 7 + 3 + 4 = 18.

Оценивание структурной сложности с помощью критерия минимального покрытия дуг графа **программно** представлено на рисунке 2. Структура графа для программы представлена в приложении Б.

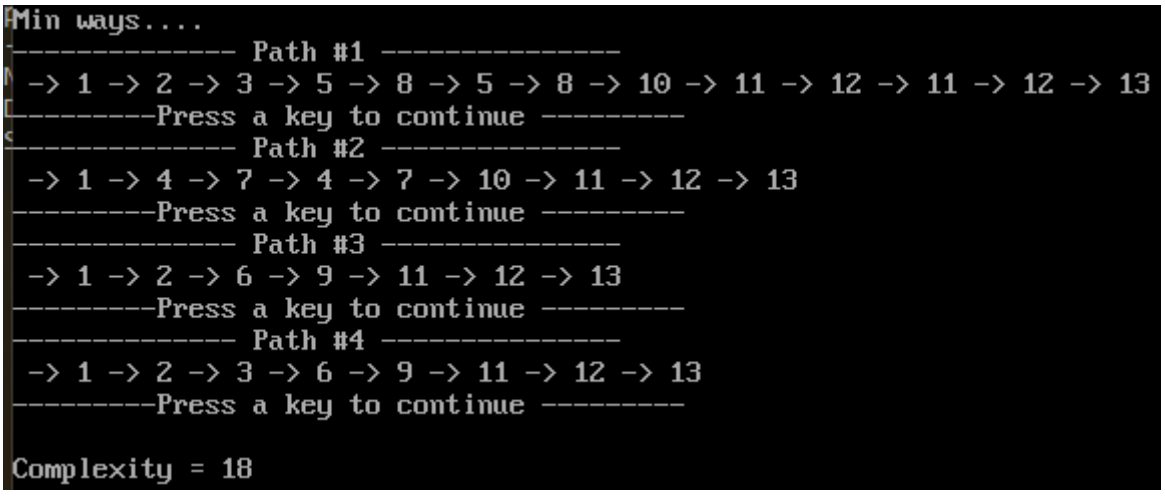

Рисунок 2 - Минимальное покрытие дуг

Оценивание структурной сложности с помощью критерия на основе цикломатического числа **программно** представлено на рисунке 3.

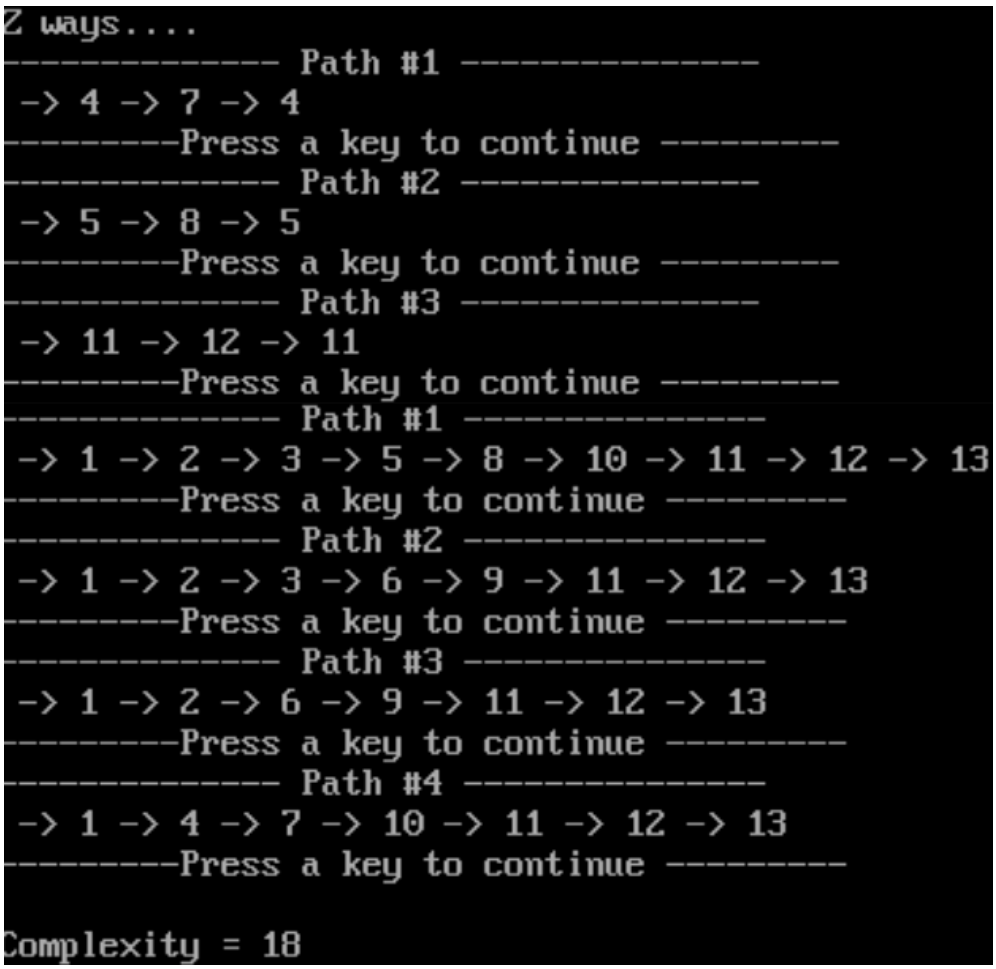

Рисунок 3 – Цикломатическое число

Оценка структурной сложности программы из 1-ой лабораторной.

Код программы из первой лабораторной представлен в приложении А. Граф программы представлен на рисунке 4.

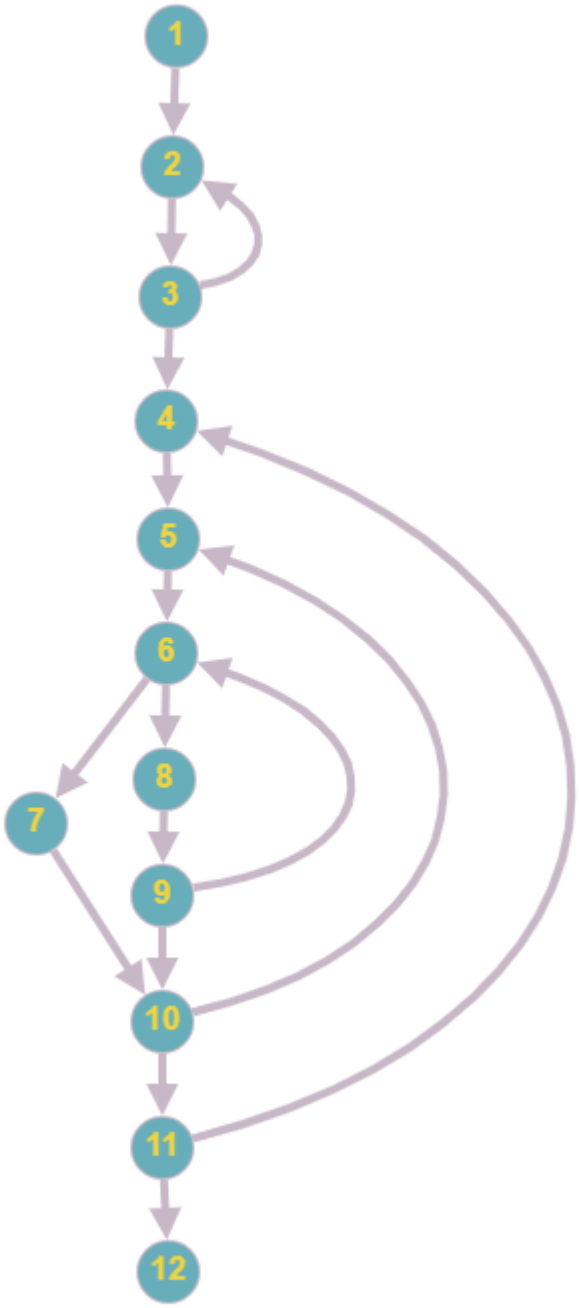

Рисунок 4 - Граф программы на языке СИ

Ключевые узлы графа:

- 2-3-2 – Генерация массива данных;
- 7 – Проверка на $arr[i] \geq arr[j]$;
- 8 – Проверка на $arr[i] < arr[j]$;
- 6-9-6 – Второй вложенный цикл;
- 5-10-5 – Первый вложенный цикл;

- 4-11-4 – Внешний цикл.

Оценивание структурной сложности с помощью критерия минимального покрытия дуг графа **вручную**.

Ветвления: 3, 6, 9, 10, 11. Всего ветвлений – 5.

Минимальный набор путей (жирным выделены ветвления):

- 1-2-**3**-2-**3**-4-5-**6**-8-**9**-**6**-7-**10**-5-**6**-7-**10**-**11**-4-5-**6**-7-**10**-**11**-12 (12 ветвл.)

Итого сложность равна 12.

Оценивание структурной сложности с помощью критерия на основе цикломатического числа **вручную**.

Число вершин в графе – 12, число ребер – 16. Для того, чтобы граф стал связным (из каждой вершины существовал путь в любую другую) достаточно добавить одно ребро (12-1). Таким образом, цикломатическое число графа равно $16-12+2*1=6$. Значит необходимо рассмотреть 6 линейно-независимых циклов и путей.

- 2-**3**-2 (1 ветвление);
- **6**-8-**9**-**6** (3 ветвления);
- 5-**6**-7-**10**-5 (2 ветвления);
- 4-5-**6**-7-**10**-**11**-4 (3 ветвления);
- 1-2-**3**-4-5-**6**-8-**9**-**10**-**11**-12 (5 ветвлений);
- 1-2-**3**-4-5-**6**-7-**10**-**11**-12 (4 ветвления).

Итого сложность равна 1 + 3 + 2 + 3 + 5 + 4 = 18.

Оценивание структурной сложности с помощью критерия минимального покрытия дуг графа и с помощью критерия на основе цикломатического числа **программно** не удалось, т.к. граф имеет «запрещенную» вершину 6, и избавиться от нее без потери соответствия графа коду программы нельзя. Результаты попытки представлены на рисунке 5. Структура графа для программы представлена в приложении В.

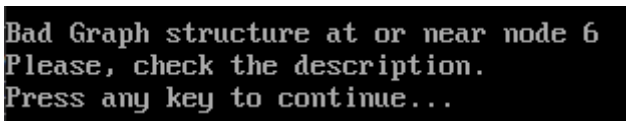

Рисунок 5 - Неудачный запуск программы

Выводы.

В результате выполнения данной лабораторной работы были изучены критерии оценивания структурной сложности программ. Была проведена оценка структурной сложности двух программ: соответствующая варианту и из первой лабораторной работы.

По результатам оценки можно сделать заключение, что программный и ручной способы совпадают с точностью до порядка путей.

ПРИЛОЖЕНИЕ А. ПРОГРАММА НА ЯЗЫКЕ СИ

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

void swap(float* x, float* y)
{
    float temp;
    temp = *x;
    *x = *y;
    *y = temp;
}

void shellsort(float arr[], int num)
{
    int i, j, k;
    for (i = num / 2; i > 0; i = i / 2)
    {
        for (j = i; j < num; j++)
        {
            for (k = j - i; k >= 0; k = k - i)
            {
                if (arr[k + i] >= arr[k])
                    break;
                else
                {
                    swap(&arr[k], &arr[k + i]);
                }
            }
        }
    }
}


int main()
{
    const int num = 200;
    float my_max = 100.0;
    float arr[num];
    int k;

    for (k = 0; k < num; k++)
    {
        arr[k] = (float)rand() / (float)(RAND_MAX / my_max);
    }
    shellsort(arr, num);
    return 0;
}
```

ПРИЛОЖЕНИЕ Б. СТРУКТУРА ГРАФА ИЗ ФАЙЛА

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

Top{1}
Last{13}

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

ПРИЛОЖЕНИЕ В. СТРУКТУРА ГРАФА НА ОСНОВЕ ПРОГРАММЫ

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

Top{1}
Last{12}

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