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. 6304 _____ Пискунов Я.А.

Преподаватель _____ Кирьянчиков В.А.

Санкт-Петербург

2020

Задание.

Выполнить оценивание структурной сложности двух программ с помощью критериев:

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

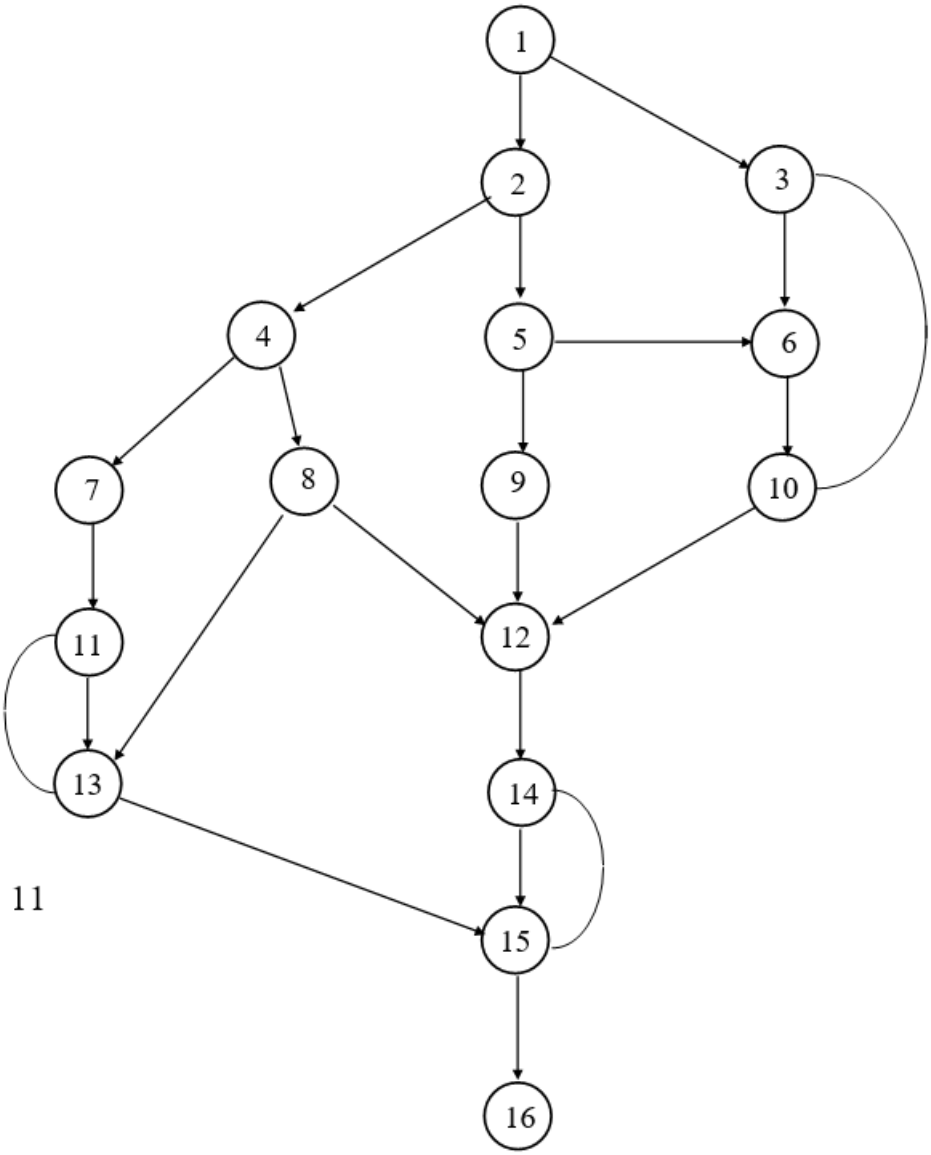
Варианты программ:

- Программа с заданной преподавателем структурой управляющего графа;
- 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.

Вариант 11.



11

Ход работы.

Оценим структурную сложность первой программы с помощью критерия минимального покрытия дуг графа. Сначала вручную.

Прежде всего подсчитаем число ветвлений – 1, 2, 4, 5, 8, 10, 13, 15. Всего 8.

Далее минимальный набор путей (жирным выделены узлы с ветвлениями):

- **1**-**2**-**4**-7-11-**13**-11-**13**-**15**-16 (6 ветвлений)
- **1**-**2**-**4**-**8**-**13**-**15**-16 (6 ветвлений)

- **1**-**2**-**4**-**8**-12-14-**15**-14-**15**-16 (6 ветвлений)
- **1**-3-6-**10**-3-6-**10**-12-14-**15**-16 (4 ветвления)
- **1**-**2**-**5**-6-**10**-12-14-**15**-16 (5 ветвлений)
- **1**-**2**-**5**-9-12-14-**15**-16 (4 ветвления)

Итого, сложность равна 6 + 6 + 6 + 4 + 5 + 4 = 31.

Далее нужно подсчитать то же с помощью программы ways.exe. Для этого необходимо представить граф в читаемом программой формате. Данный граф представлен в приложении А.

Результат работы программы следующий. Набор путей:

- 1-2-4-7-11-13-15-14-15-16
- 1-3-6-10-12-14-15-16
- 1-2-5-6-10-12-14-15-16
- 1-2-4-8-13-15-16
- 1-2-4-8-12-14-15-16
- 1-2-5-9-12-14-15-16

Сложность – 31. Как можно отметить, пути частично отличаются, однако полученная в результате сложность такая же.

Далее оценим структурную сложность с помощью критерия на основе цикломатического числа. Сначала вручную.

Число вершин в графе – 16, число ребер – 23. Для того, чтобы граф стал связным (из каждой вершины существовал путь в любую другую) достаточно добавить одно ребро из 16 в 1. Таким образом, цикломатическое число графа равно 23 – 16 + 2*1 = 9. Значит необходимо рассмотреть 9 линейно независимых циклов и путей.

- 3-6-**10** (1 ветвление)
- 11-**13** (1 ветвление)
- 14-**15** (1 ветвление)
- **1**-**2**-**4**-7-11-**13**-**15**-16 (5 ветвлений)
- **1**-**2**-**4**-**8**-**13**-**15**-16 (6 ветвлений)

- **1**-**2**-**4**-**8**-12-14-**15**-16 (5 ветвлений)
- **1**-3-6-**10**-12-14-**15**-16 (3 ветвления)
- **1**-**2**-**5**-6-**10**-12-14-**15**-16 (5 ветвлений)
- **1**-**2**-**5**-9-12-14-**15**-16 (4 ветвления)

Итого, сложность равна 1 + 1 + 1 + 5 + 6 + 5 + 3 + 5 + 4 = 31.

Произведем аналогичный расчет с помощью программы.

Пути:

- 3-6-10
- 11-13
- 14-15
- 1-2-4-7-11-13-15-16
- 1-2-4-8-13-15-16
- 1-2-4-8-12-14-15-16
- 1-2-5-6-10-12-14-15-16
- 1-2-5-9-12-14-15-16
- 1-3-6-10-12-12-14-15-16

Сложность – 31. В данном случае также вычисленная автоматически сложность соответствует ручной.

Далее проведем аналогичные расчеты для программы на языке Pascal из первой лабораторной работы. Для этого сначала необходимо составить граф. Он представлен на рис. 1. Ключевые узлы графа:

2 – цикл repeat… until …

3 – передача управления get_data

7 – передача управления solve

11 – проверка, что det=0.0

14, 15, 16 – передача управления setup

23 – проверка на ошибку в main

25 – передача управления write_data

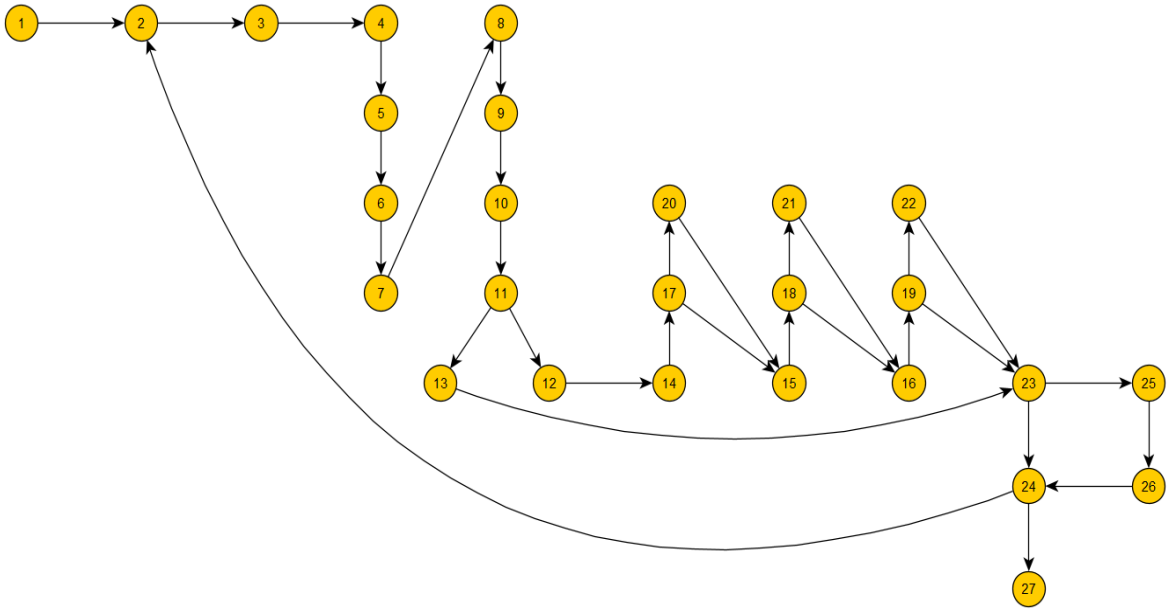


Рисунок 1 – структурный граф второй программы

Приведенный в формат, принимаемый программой, граф представлен в приложении Б.

Произведем ручной расчет с помощью минимального покрытия дуг графа. Ветвления – 11, 17, 18, 19, 23, 24. Пути:

1-2-3-4-5-6-7-8-9-10-**11**-12-14-**17**-20-15-**18**-21-16-**19**-22-**23**-25-26-**24**-2-3-4-5-6-7-8-9-10-**11**-12-14-**17**-15-**18**-16-**19**-**23**-**24**-2-3-4-5-6-7-8-9-10-**11**-13-**23**-**24**-27 (15 ветвлений). Таким образом, сложность – 15.

Произведем аналогичный расчет с помощью программы. Пути:

1-2-3-4-5-6-7-8-9-10-11-12-14-17-20-15-18-21-16-19-22-23-24-2-3-4-5-6-7-8-9-10-11-13-23-25-26-24-2-3-4-5-6-7-8-9-10-11-12-14-17-15-18-16-19-23-25-26-24-27, сложность – 15.

Произведем ручной расчет сложности на основе цикломатического числа. Число вершин – 27, число ребер – 32, для того чтобы сделать граф связным, необходимо добавить одно ребро из 27 в 1. Таким образом, цикломатическое число графа равно 32 – 27 + 2*1 = 7. Пути:

- 2-3-4-5-6-7-8-9-10-**11**-13-**23**-**24**-2 (3 ветвления)

- 1-2-3-4-5-6-7-8-9-10-**11**-12-14-**17**-20-15-**18**-21-16-**19**-22-**23**-25-26-**24**-27 (6 ветвлений)
- 1-2-3-4-5-6-7-8-9-10-**11**-12-14-**17**-20-15-**18**-21-16-**19**-22-**23**-**24**-27 (6 ветвлений)
- 1-2-3-4-5-6-7-8-9-10-**11**-12-14-**17**-15-**18**-16-**19**-**23**-**24**-27 (6 ветвлений)
- 1-2-3-4-5-6-7-8-9-10-**11**-12-14-**17**-15-**18**-16-**19**-**23**-25-26-**24**-27 (6 ветвлений)
- 1-2-3-4-5-6-7-8-9-10-**11**-13-**23**-**24**-27 (3 ветвления)
- 1-2-3-4-5-6-7-8-9-10-**11**-13-**23**-25-26-**24**-27 (3 ветвления)

Итого, сложность выходит 33.

Программный расчет по данному способу невозможен из-за завершения программы с неизвестной ошибкой, которая кроется не во входных данных (так как на тех же входных данных расчет по первому способу прошел успешно).

Выводы.

В ходе выполнения данной лабораторной работы были изучены критерии оценивания структурной сложности программ. Была проведена ручная и автоматическая оценка структурной сложности программ. Кроме того, на примере программы из предыдущей лабораторной были изучены и применены на практике принципы построения структурного графа программы.

ПРИЛОЖЕНИЕ А

ПЕРВЫЙ ГРАФ

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

ПРИЛОЖЕНИЕ Б

ВТОРОЙ ГРАФ

```
Nodes{1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,
26,27}
Top{1}
Last{27}
Arcs{
arc(1,2);
arc(2,3);
arc(3,4);
arc(4,5);
arc(5,6);
arc(6,7);
arc(7,8);
arc(8,9);
arc(9,10);
arc(10,11);
arc(11,12);
arc(11,13);
arc(12,14);
arc(13,23);
arc(14,17);
arc(15,18);
arc(16,19);
arc(17,20);
arc(17,15);
arc(18,21);
arc(18,16);
arc(19,22);
arc(19,23);
arc(20,15);
arc(21,16);
arc(22,23);
arc(23,24);
arc(23,25);
arc(24,2);
arc(24,27);
arc(25,26);
arc(26,24);
}
```