# 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. 7304 |  | Кошманов Н.А. |
| Преподаватель |  | Ефремов М.А. |

Санкт-Петербург 2021

# Цель работы

Изучение применения метрик структурной сложности программ —

критерия минимального покрытия и анализа базовых маршрутов.

# Постановка задачи

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.

# Ход работы

1. Вариант – 8

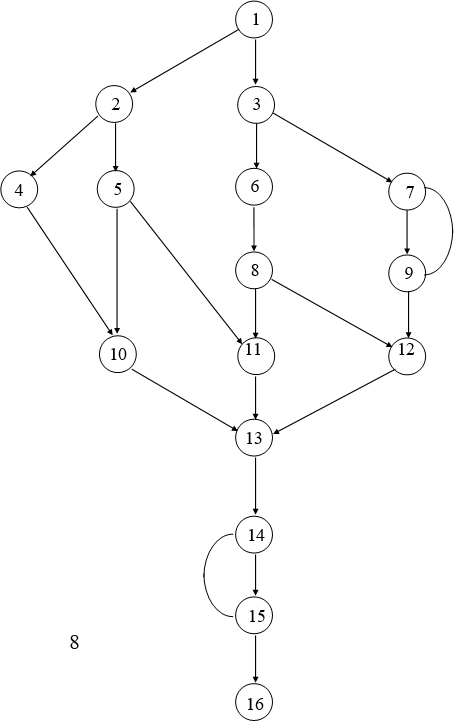


Рисунок 2 — Исходный граф

1. Ручной расчет
   1. Первый критерий
      * **М1: 1** – **2** – 4 – 10 – 13 – 14 – **15** – 14 – **15** –16 = 4
      * **М2: 1** – **2** – **5** – 10 – 13 – 14 – **15** – 16 = 4
      * **М3: 1** – **2** – **5** – 11 – 13 – 14 – **15** – 16 = 4
      * **М4: 1** – **3** – 6 – **8** – 11 –13 – 14 – **15** – 16 = 4
      * **М5: 1** – **3** – 6 – **8** – 12 –13 – 14 – **15** – 16 = 4
      * **М6: 1** – **3** – 7 – **9** – 7 – **9** – 12 – 13 – 14 – **15** – 16 = 5

По первому критерию требуется 6 маршрутов М = 6 Сложность программы по первому критерию:

* 1. Второй критерий

**M**

**S 2**  **i**

**i****1**

= 4 + 4 + 4 + 4 + 4 + 5= 25

Y = 22, N = 16, P = 1, nв = 7

Z = Y - N + 2 \* P = 22 - 16 + 2\*1 = 8

Z = nв + 1 = 7 + 1 = 8

Линейно-независимые маршруты:

* **М1:** 14 – **15** – 14 = 1
* **М2:** 7 – **9** – 7 = 1
* **М3: 1** – **2** – 4 – 10 – 13 – 14 – **15** – 16 = 3
* **М4: 1** – **2** – **5** – 10 – 13 – 14 – **15** – 16 = 4
* **М5: 1** – **2** – **5** – 11 – 13 – 14 – **15** – 16 = 4
* **М6: 1** – **3** – 6 – **8** – 11 – 13 – 14 – **15** – 16 = 4
* **М7: 1** – **3** – 6 – **8** – 12 – 13 – 14 – **15** – 16 = 4
* **М8: 1** – **3** – 7 – **9** – 12 – 13 – 14 – **15** – 16 = 4

**M**

**S 2**  **i**

**i****1**

1. Автоматический расчет

= 1 + 1 + 3 + 4 + 4 + 4 + 4 + 4= 25

Данный граф рис. 1 представлен в необходимом для программы ways.exe

формате (см. Приложение А).

Результаты работы программы по первому и второму критериям представлены на рис. 2, 3 соответственно.

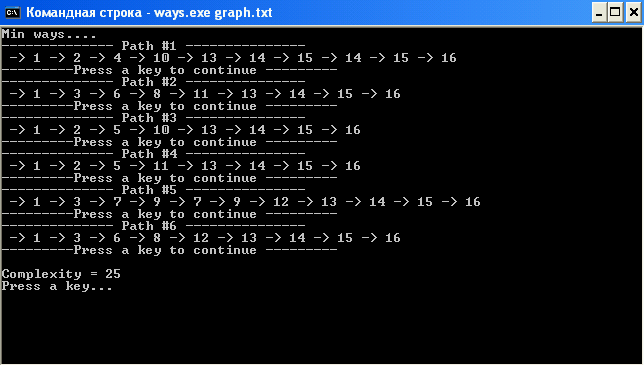


Рисунок 2 — Автоматический расчет по первому критерию

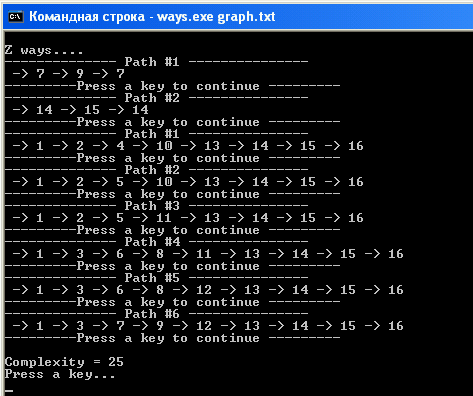


Рисунок 3 — Автоматический расчет по второму критерию

1. Исходный код программы представлен в Приложении Б. Графовое представление программы — в Приложении В.
2. Ручной подсчет. 5.1.Первый критерий

* **М1**: 1 – 2 – 3 – 4 – 5 – **6** – 5 – **6** – 7 – **8** – 4 – 5 – **6** – 7 – **8** – 9 – 10 – 11 – 12

– **13** – 15 – **16** – 12 – **13** – 14 – 15 – **16** – 17 – **18** – 11 – 12 – **13** – **16** – 17 –

**18** – 19 – 20 – **21** – 20 – **21** – 22 – **23** – 10 – 11 – 12 – **13** – 15 – **16** – 17 – **18**

– 19 – 20 – **21** – 22 – **23** – 24 = 21

5.2.Второй критерий

**M**

**S 2**  **i**

**i****1**

= 21

Y = 30, N = 24, P = 1, nв = 7

Z = Y - N + 2 \* P = 30 - 24 + 2\*1 = 8

Z = nв + 1 = 7 + 1 = 8

Линейно-независимые маршруты:

* **М1:** 5 – **6** – 5 = 1
* **М2:** 12 – **13** – 15 – **16** – 12 = 2
* **М3:** 4 – 5 – **6** – 7 – **8** – 4 = 2
* **М4:** 11 – 12 – **13** – 15 – **16** – 17 – **18** – 11 = 3
* **М5:** 20 – **21** – 20 = 1
* **М6:** 10 – 11 – 12 – **13** – 15 – **16** – 17 – **18** – 19 – 20 – **21** – 22 – **23** – 10 = 4
* **М7:** 1 – 2 – 3 – 4 – 5 – **6** – 7 – **8** – 9 – 10 – 11 – 12 – **13** – 14 – 15 – **16** – 17

– **18** – 19 – 20 – **21** – 22 – **23** – 24 = 7

* **М8:** 1 – 2 – 3 – 4 – 5 – **6** – 7 – **8** – 9 – 10 – 11 – 12 – **13**– 15 – 16 – 17 – **18** –

19 – 20 – **21** – 22 – **23** – 24 = 7

**M**

**S 2**  **i**

**i****1**

= 1 + 2 + 2 + 3 + 1 + 4 + 7 + 7 = 28

1. Автоматический расчет

Данный граф из Приложения В представлен в необходимом для программы

ways.exe формате (см. Приложение Г).

Результаты работы программы по первому и второму критериям представлены на рис. 4, 5 соответственно.

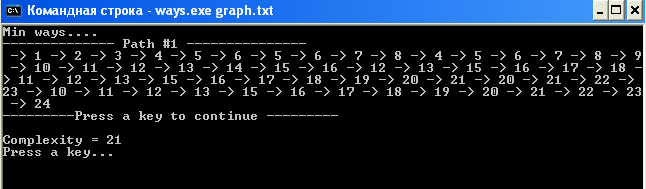


Рисунок 2 — Автоматический расчет по первому критерию

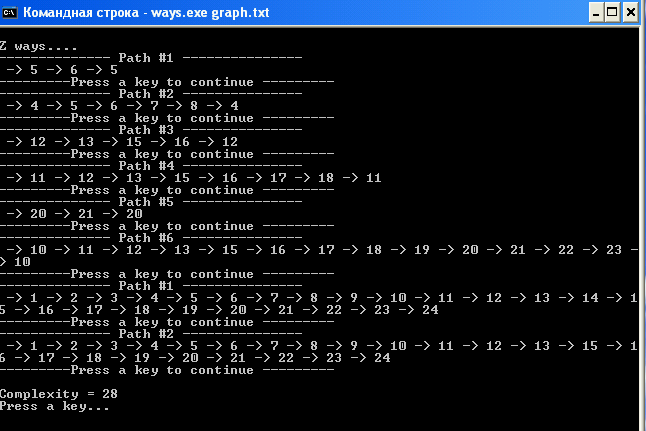


Рисунок 3 — Автоматический расчет по второму критерию

# Вывод

В данной лабораторной работе была выполнена оценка структурной сложности двух программ с помощью критериев: минимального покрытия дуг графа и выбора маршрутов на основе цикломатического числа графа. Расчеты были проведены как ручным, так и программным способом.

# ПРИЛОЖЕНИЕ А

**Описание графа**

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(3, 7);

arc(4, 10);

arc(5, 10);

arc(5, 11);

arc(6, 8);

arc(7, 9);

arc(8, 11);

arc(8, 12);

arc(9, 7);

arc(9, 12);

arc(10, 13);

arc(11, 13);

arc(12, 13);

arc(13, 14);

arc(14, 15);

arc(15, 14);

arc(15, 16);

}

#include <stdio.h> #define rmax 9

#define cmax 3

# ПРИЛОЖЕНИЕ Б

**Исходный код на C**

typedef double ary[rmax]; typedef double arys[cmax]; typedef double ary2[rmax][cmax];

typedef double ary2s[cmax][cmax]; typedef double\* pointer;

typedef double\*\* dpointer;

void get\_data(ary2 x, ary y, int nrow, int ncol) { for (int i = 0; i < nrow; i++) {

x[i][0] = 1;

for (int j = 1; j < ncol; j++) { x[i][j] = (i + 1) \* x[i][j - 1];

}

y[i] = 2 \* (i + 1);

}

}

void square(ary2 x, double\* y, ary2s a, double\* g, int nrow, int ncol) { for (int k = 0; k < ncol; k++) {

for (int l = 0; l <= k; l++) { a[k][l] = 0;

for (int i = 0; i < nrow; i++) {

a[k][l] = a[k][l] + x[i][l] \* x[i][k]; if (k != l)

a[l][k] = a[k][l];

}

}

g[k] = 0;

for (int i = 0; i < nrow; i++) { g[k] = g[k] + y[i] \* x[i][k];

}

}

}

int main() {

int nrow = 5; int ncol = 3;

ary2 x; ary y;

arys g; ary2s a;

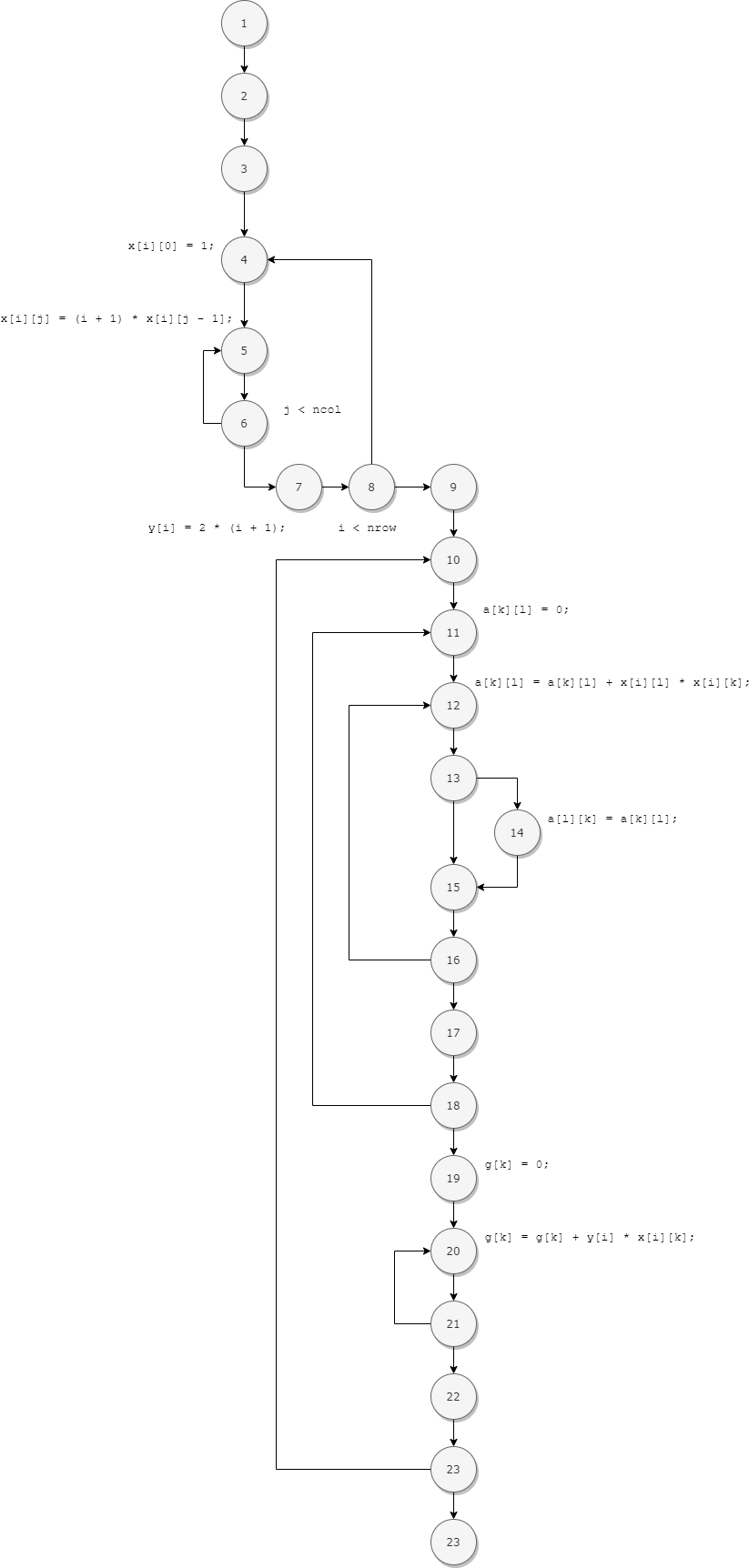
get\_data(x, y, nrow, ncol); square(x, y, a, g, nrow, ncol);

return 0;

}

# ПРИЛОЖЕНИЕ В

**Граф для программы на С**



# ПРИЛОЖЕНИЕ Г

**Граф для программы на С**

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}

Top {1}

Last {24} Arcs {

arc(1, 2);

arc(2, 3);

arc(3, 4);

arc(4, 5);

arc(5, 6);

arc(6, 5);

arc(6, 7);

arc(7, 8);

arc(8, 4);

arc(8, 9);

arc(9, 10);

arc(10, 11);

arc(11, 12);

arc(12, 13);

arc(13, 14);

arc(13, 15);

arc(14, 15);

arc(15, 16);

arc(16, 12);

arc(16, 17);

arc(17, 18);

arc(18, 11);

arc(18, 19);

arc(19, 20);

arc(20, 21);

arc(21, 20);

arc(21, 22);

arc(22, 23);

arc(23, 10);

arc(23, 24);

}