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

ОТЧЁТ

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

Студент гр. 6304 _____ Корытов П.В.
Преподаватель _____ Кирьянчиков В.А.

Санкт-Петербург
2020

1. Цель работы

Изучить применение метрик структурной сложности программ — критерия минимального покрытия и анализа базовых маршрутов.

2. Постановка задачи

Выполнить оценивание структурной сложности двух программ с помощью критериев:

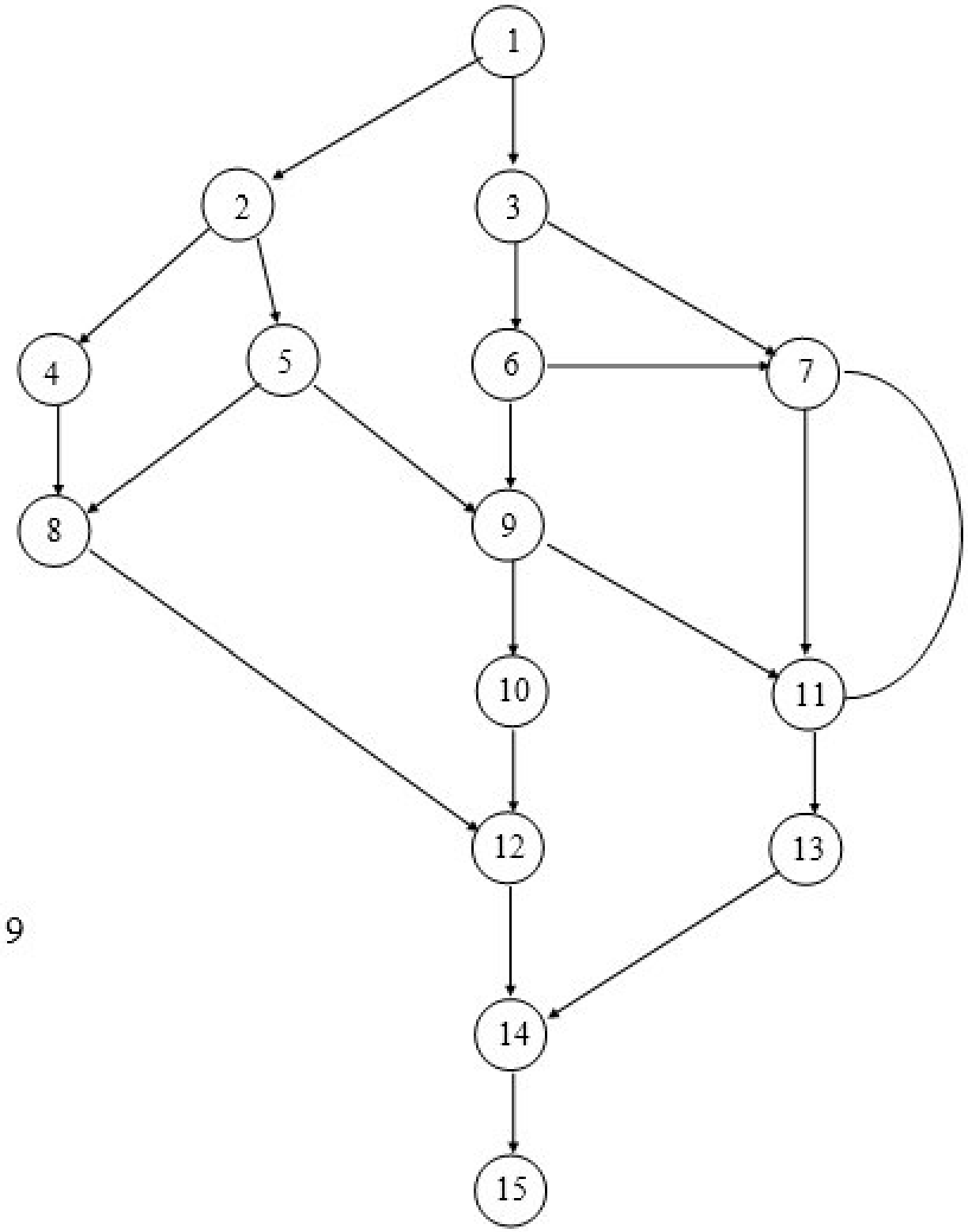
- Минимального покрытия дуг графа;
- Выбора маршрутов на основе цикломатического числа графа.

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.



9

Рисунок 1 – Заданный граф

3. Ход работы

3.1. Программа из задания

3.1.1. Критерий минимального покрытия

Маршруты для минимального покрытия:

1. **1**, **2**, 4, 8, 12, 14, 15
2. **1**, **2**, **5**, 8, 12, 14, 15
3. **1**, **2**, **5**, **9**, **11**, 7, **11**, 13, 14, 15
4. **1**, **3**, **6**, **9**, 10, 12, 14, 15
5. **1**, **3**, 7, **11**, 13, 14, 15
6. **1**, **3**, **6**, 7, **11**, 13, 14, 15

Сложность программы:

$$S_2 = \sum_{i=1}^{M} \xi_i = 2 + 3 + 6 + 4 + 3 + 5 = 23. \tag{3.1}$$

3.1.2. Анализ базовых маршрутов

Число вершин графа $N = 15$, число дуг графа $Y = 21$, число связных компонент $\Omega = 1$. Цикломатическое число Z:

$$Z = Y - N + 2 * \Omega = 8. \tag{3.2}$$

Маршруты:

1. 7, **11**.
2. **1**, **2**, 4, 8, 12, 14, 15;
3. **1**, **2**, **5**, 8, 12, 14, 15;
4. **1**, **2**, **5**, **9**, 10, 12, 14, 15;
5. **1**, **2**, **5**, **9**, **11**, 13, 14, 15;
6. **1**, **3**, **6**, **9**, 10, 12, 14, 15;
7. **1**, **3**, 7, **11**, 13, 14, 15;
8. **1**, **3**, **6**, 7, **11**, 13, 14, 15;

Сложность программы:

$$S_2 = \sum_{i=1}^{M} \xi_i = 2 + 2 + 3 + 4 + 5 + 4 + 3 + 4 = 27. \tag{3.3}$$

3.1.3. Программный анализ

Граф задан в нотации приложенной программы. Файл с описанием — в приложении А.

С графом в приведенном описании возникла ошибка работы из-за ребра 9–11. Для решения добавлено дополнительная вершина. Лог работы программы в приложении Б.

Маршруты для минимального покрытия:

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

Сложность: 25

Базовые маршруты:

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

Сложность: 25

3.2. Программа из ЛР1

Произведено построение графа работы для программы из ЛР1 (приложение В). Полученный граф представлен на рис. 2.

3.2.1. Критерий минимального покрытия

Маршруты для минимального покрытия:

1. 1, 2, **3**, 2, **3**, 4, 5, 6, **7**, **8**, 5, 6, **7**, 6, **7**, **8**, 9, 10, **11**, 10, **11**, **12**, 9, 10, **11**, **12**, 13, **14**, 13, **14**, 15, 16, **17**, 16, 17, **18**, 15, 16,

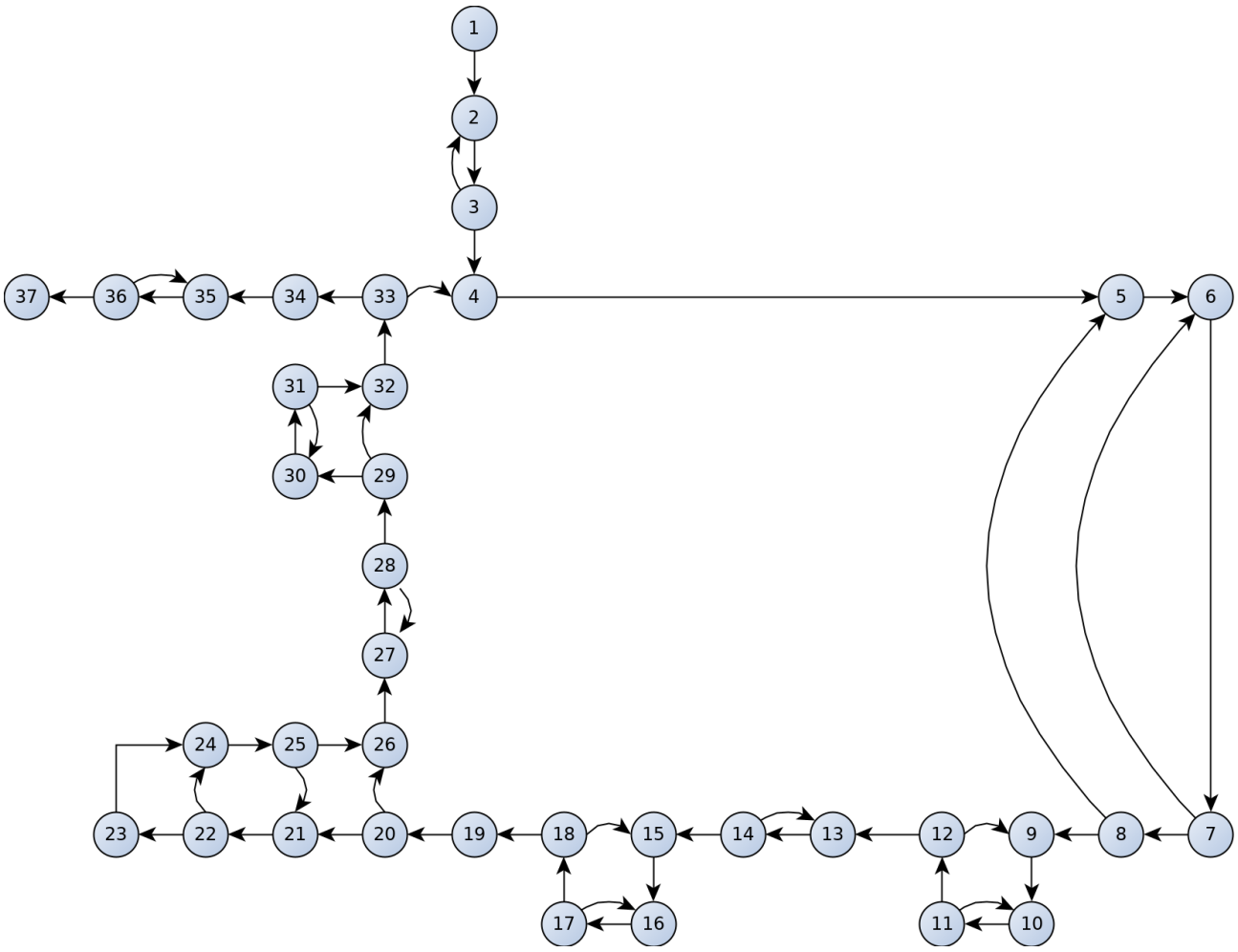


Рисунок 2 – Граф приложения

17, 18, 19, **20**, 21, **22**, 23, 24, **25**, 21, **22**, 24, **25**, 26, 27, **28**, 27, **28**, **29**, 30, **31**, 30, **31**, 32, **33**, 4, 5, 6, **7**, **8**, 9, 10, **11**, **12**, 13, **14**, 15, 16, **17**, **18**, 19, **20**, 26, 27, **28**, **29**, 32, **33**, 34, 35, **36**, 35, **36**, 37.

Сложность: 41.

3.2.2. Анализ базовых маршрутов

Число вершин графа $N = 37$, число дуг графа $Y = 52$, число связных компонент $\Omega = 1$. Цикломатическое число Z:

$$Z = Y - N + 2 * \Omega = 17. \tag{3.4}$$

Маршруты:

1. 2, **3**, 2
2. 6, **7**, 6

3. 5, 6, **7**, **8**, 5
4. 10, **11**, 10
5. 9, 10, **11**, **12**, 9
6. 16, **17**, 16
7. 15, 16, **17**, **18**, 15
8. 21, **22**, 23, 24, **25**, 21
9. 27, **28**, 27
10. 30, **31**, 30,
11. 35, 36
12. 4, 5, 6, **7**, **8**, 9, 10, **11**, **12**, 13, **14**, 15, 16, **17**, **18**, 19, **20**, 26, 27, **28**, **29**, 32, **33**, 4
13. 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, **28**, **29**, 30, **31**, 32, **33**, 34, 35, **36**, 37
14. 1, 2, **3**, 4, 5, 6, **7**, **8**, 9, 10, **11**, **12**, 13, **14**, 15, 16, **17**, **18**, 19, **20**, 21, **22**, 24, **25**, 26, 27, **28**, **29**, 30, **31**, 32, **33**, 34, 35, **36**, 37
15. 1, 2, **3**, 4, 5, 6, **7**, **8**, 9, 10, **11**, **12**, 13, **14**, 15, 16, **17**, **18**, 19, **20**, 26, 27, **28**, **29**, 30, **31**, 32, **33**, 34, 35, **36**, 37
16. 1, 2, **3**, 4, 5, 6, **7**, **8**, 9, 10, **11**, **12**, 13, **14**, 15, 16, **17**, **18**, 19, **20**, 21, **22**, 24, **25**, 26, 27, **28**, **29**, 32, **33**, 34, 35, **36**, 37
17. 1, 2, **3**, 4, 5, 6, **7**, **8**, 9, 10, **11**, **12**, 13, **14**, 15, 16, **17**, **18**, 19, **20**, 26, 27, **28**, **29**, 32, **33**, 34, 35, **36**, 37

Сложность: 99

3.2.3. Программный анализ

Граф задан в нотации программы. Результаты в приложении Г.

Лог работы программы в приложении Д.

Результаты вычисления минимального покрытия соответствуют полученным в п. 3.2.1.

Вычисление базовых маршрутов заканчивается некорректным завершением программы-анализатора.

4. Выводы

Изучено применение:

- критерия минимального покрытия,

- анализа базовых маршрутов

для оценки структорной сложности программ.

Проведено ручное и автоматизированное вычисление метрик для двух примеров, получены похожие результаты.

Установлено, что программа-анализатор не отличается устойчивостью.

ПРИЛОЖЕНИЕ А

Граф из задания

```
Nodes { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16 }

Top { 1 }

Last { 15 }

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

ПРИЛОЖЕНИЕ Б

Лог работы для графа из задания

```
Min ways....
------------- Path #1 --------------
 -> 1 -> 2 -> 4 -> 8 -> 12 -> 14 -> 15
---------Press a key to continue ---------
------------- Path #2 --------------
 -> 1 -> 3 -> 6 -> 7 -> 11 -> 7 -> 11 -> 16 -> 13 -> 14 -> 15
---------Press a key to continue ---------
------------- Path #3 --------------
 -> 1 -> 2 -> 5 -> 8 -> 12 -> 14 -> 15
---------Press a key to continue ---------
------------- Path #4 --------------
 -> 1 -> 2 -> 5 -> 9 -> 10 -> 12 -> 14 -> 15
---------Press a key to continue ---------
------------- Path #5 --------------
 -> 1 -> 2 -> 5 -> 9 -> 16 -> 13 -> 14 -> 15
---------Press a key to continue ---------
------------- Path #6 --------------
 -> 1 -> 3 -> 7 -> 11 -> 16 -> 13 -> 14 -> 15
---------Press a key to continue ---------
------------- Path #7 --------------
 -> 1 -> 3 -> 6 -> 9 -> 10 -> 12 -> 14 -> 15
---------Press a key to continue ---------

Complexity = 25
Press a key...

Z ways....
------------- Path #1 --------------
 -> 7 -> 11 -> 7
---------Press a key to continue ---------
------------- Path #1 --------------
```

```
 -> 1 -> 2 -> 4 -> 8 -> 12 -> 14 -> 15
---------Press a key to continue ---------
-------------- Path #2 ---------------
 -> 1 -> 2 -> 5 -> 8 -> 12 -> 14 -> 15
---------Press a key to continue ---------
-------------- Path #3 ---------------
 -> 1 -> 2 -> 5 -> 9 -> 10 -> 12 -> 14 -> 15
---------Press a key to continue ---------
-------------- Path #4 ---------------
 -> 1 -> 2 -> 5 -> 9 -> 16 -> 13 -> 14 -> 15
---------Press a key to continue ---------
-------------- Path #5 ---------------
 -> 1 -> 3 -> 6 -> 7 -> 11 -> 16 -> 13 -> 14 -> 15
---------Press a key to continue ---------
-------------- Path #6 ---------------
 -> 1 -> 3 -> 6 -> 9 -> 10 -> 12 -> 14 -> 15
---------Press a key to continue ---------
-------------- Path #7 ---------------
 -> 1 -> 3 -> 7 -> 11 -> 16 -> 13 -> 14 -> 15
---------Press a key to continue ---------

Complexity = 25
Press a key...
```

ПРИЛОЖЕНИЕ В

Программа из ЛР1

```
#include ”stdio.h”
#include ”stdlib.h”
#include ”stdbool.h”

#define RMAX 3
#define CMAX 3

float** _alloc_matr(int a, int b) {
    float** m = (float**)malloc(a * sizeof(float*));
    for (int i = 0; i < CMAX; i ++) {
        m[i] = (float*)malloc(b * sizeof(float));
    }
    return m;
}

void _free_matr(float** m, int a) {
    for (int i = 0; i < a; i ++) {
        free(m[i]);
    }
    free(m);
}


/* print out the answers */
void print_matr(float** a, float* y) {
    for (int i = 0; i < RMAX; i++) {
        for (int j = 0; j < CMAX; j ++) {
            printf(”%f  ”, a[i][j]);
        }
        printf(”: %f\n”, y[i]);
    }
```

```
32  }
33
34  /* get the values for n, and arrays a,y */
35  void get_data(float** a, float* y) {
36      for (int i = 0; i < RMAX; i++) {
37          printf("Equation %d\n", i);
38          for (int j = 0; j < CMAX; j++) {
39              printf("%d: ", j);
40              scanf("%f", &a[i][j]);
41          }
42          printf("C: ");
43          scanf("%f", &y[i]);
44      }
45      print_matr(a, y);
46      printf("\n");
47  }
48
49  /* pascal program to calculate the determinant of a 3-by-3matrix */
50  float deter(float** a) {
51      return a[0][0] * (a[1][1] * a[2][2] - a [2][1] * a[1][2])
52           - a[0][1] * (a[1][0] * a[2][2] - a [2][0] * a[1][2])
53           + a[0][2] * (a[1][0] * a[2][1] - a [2][0] * a[1][1]);
54  }
55
56  void setup(float** a, float** b, float* coef, float* y, int j,
    ↪  float det) {
57      for (int i = 0; i < RMAX; i++) {
58          b[i][j] = y[i];
59          if (j > 0) {
60              b[i][j-1] = a[i][j-1];
61          }
62      }
63      coef[j] = deter(b) / det;
64  }
```

```
65
66  bool solve(float** a, float* y, float* coef) {
67      float** b = _alloc_matr(RMAX, CMAX);
68      float det = 0;
69      for (int i = 0; i < RMAX; i++) {
70          for (int j = 0; j < CMAX; j++) {
71              b[i][j] = a[i][j];
72          }
73      }
74      det = deter(b);
75      if (det == 0) {
76          printf("ERROR: matrix is singular.");
77          return true;
78      }
79      setup(a, b, coef, y, 0, det);
80      setup(a, b, coef, y, 1, det);
81      setup(a, b, coef, y, 2, det);
82      _free_matr(b, RMAX);
83      return false;
84  }
85
86  void write_data(float* coef) {
87      for (int i = 0; i < CMAX; i++) {
88          printf("%f ", coef[i]);
89      }
90      printf("\n");
91  }
92
93  int main() {
94      float** a = _alloc_matr(RMAX, CMAX);
95      float* y = (float*)malloc(CMAX * sizeof(float));
96      float* coef = (float*)malloc(CMAX * sizeof(float));
97      bool error;
98      char scan;
```

```
 99      while (true) {
100          get_data(a, y);
101          error = solve(a, y, coef);
102          if (!error) {
103              write_data(coef);
104          }
105          printf("More? ");
106          scanf(" %c", &scan);
107          if (scan != 'y') {
108              break;
109          }
110      }
111      free(y);
112      free(coef);
113      _free_matr(a, RMAX);
114      return 0;
115  }
```

ПРИЛОЖЕНИЕ Г

Граф ЛР1 в нотации программы-анализатора

```
1  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, 28, 29, 30, 31, 32, 33,
   ↪  34, 35, 36, 37 }
2
3  Top { 1 }
4
5  Last { 37 }
6
7  Arcs {
8      arc(1, 2);
9      arc(2, 3);
10     arc(3, 2);
11     arc(3, 4);
12     arc(4, 5);
13     arc(5, 6);
14     arc(6, 7);
15     arc(7, 6);
16     arc(7, 8);
17     arc(8, 5);
18     arc(8, 9);
19     arc(9, 10);
20     arc(10, 11);
21     arc(11, 10);
22     arc(11, 12);
23     arc(12, 9);
24     arc(12, 13);
25     arc(13, 14);
26     arc(14, 13);
27     arc(14, 15);
28     arc(15, 16);
29     arc(16, 17);
```

```
30      arc(17, 16);
31      arc(17, 18);
32      arc(18, 15);
33      arc(18, 19);
34      arc(19, 20);
35      arc(20, 21);
36      arc(20, 26);
37      arc(21, 22);
38      arc(22, 23);
39      arc(22, 24);
40      arc(23, 24);
41      arc(24, 25);
42      arc(25, 21);
43      arc(25, 26);
44      arc(26, 27);
45      arc(27, 28);
46      arc(28, 27);
47      arc(28, 29);
48      arc(29, 30);
49      arc(29, 32);
50      arc(30, 31);
51      arc(31, 30);
52      arc(31, 32);
53      arc(32, 33);
54      arc(33, 4);
55      arc(33, 34);
56      arc(34, 35);
57      arc(35, 36);
58      arc(36, 35);
59      arc(36, 37);
60  }
```

ПРИЛОЖЕНИЕ Д

Лог работы программы для графа ЛР1

```
1  Min ways....
2  -------------- Path #1 ---------------
3   -> 1 -> 2 -> 3 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 6 -> 7 -> 8 -> 5
    ↪  -> 6 -> 7 -> 8 -> 9 -> 10 -> 11 -> 10 -> 11 -> 12 -> 9 -> 10
    ↪  -> 11 -> 12 -> 13 -> 14 -> 13 -> 14 -> 15 -> 16 -> 17 -> 16 ->
    ↪  17 -> 18 -> 15 -> 16 -> 17 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23
    ↪  -> 24 -> 25 -> 21 -> 22 -> 24 -> 25 -> 26 -> 27 -> 28 -> 27 ->
    ↪  28 -> 29 -> 30 -> 31 -> 30 -> 31 -> 32 -> 33 -> 4 -> 5 -> 6 ->
    ↪  7 -> 8 -> 9 -> 10 -> 11 -> 12 -> 13 -> 14 -> 15 -> 16 -> 17 ->
    ↪  18 -> 19 -> 20 -> 26 -> 27 -> 28 -> 29 -> 32 -> 33 -> 34 -> 35
    ↪  -> 36 -> 35 -> 36 -> 37
4  ---------Press a key to continue ---------
5
6  Complexity = 43
7  Press a key...
8
9  Z ways....
10 Abnormal program termination
```