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. 7304 _____ Соколов И.Д.

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 18

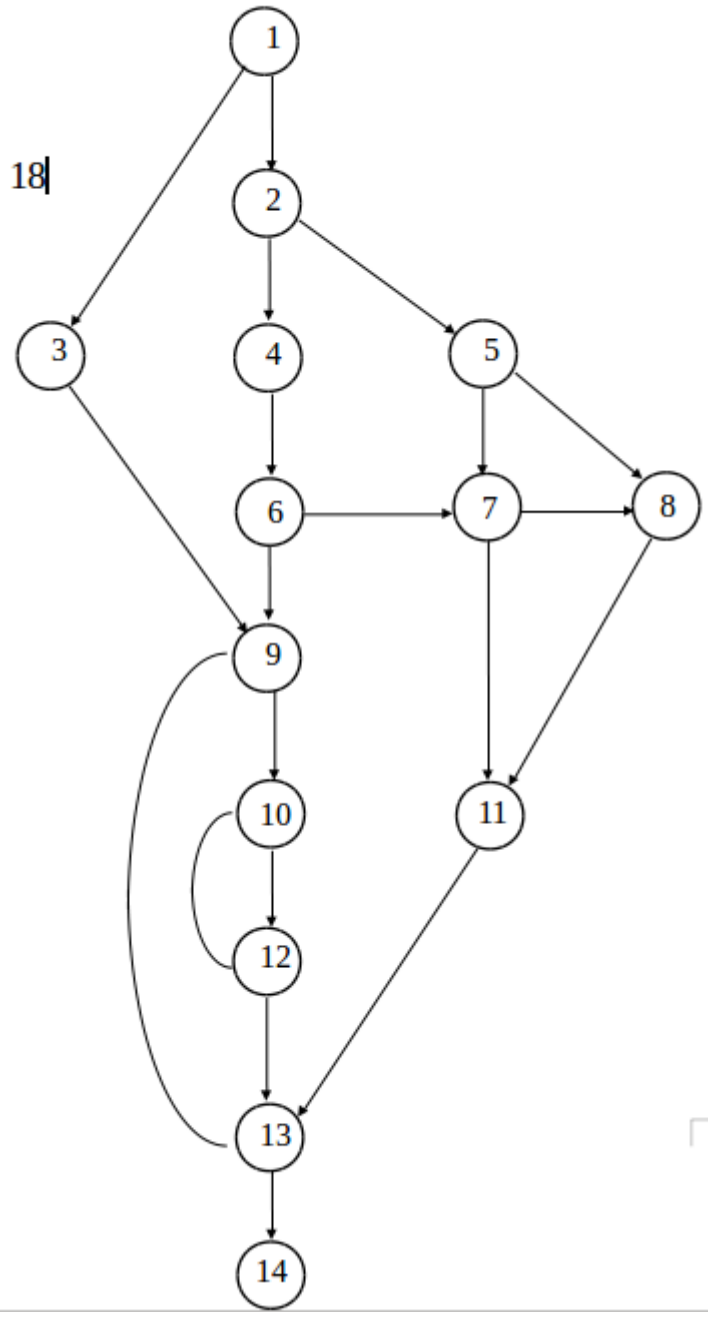


Рисунок 1 - Исходный граф

2. Ручной расчет

2.1. Первый критерий

M1: **1**-3-9-10-**12**-**13**-9-10-**12**-**13**-14 = 5

M2: **1**-**2**-4-**6**-9-10-**12**-10-**12**-**13**-14 = 6

M3: **1**-**2**-4-**6**-**7**-11-**13**-14 = 5

M4: **1**-**2**-**5**-**7**-8-11-**13**-14= 5

M5: **1**-**2**-**5**-8-11-**13**-14= 4

S = 5+6+5+5+4 = 25 – Сложность по первому критерию

Кол-во маршрутов, необходимое для прохождения по каждой дуге и посещения каждой вершины – 5

2.2. Второй критерий

Y = 20

N = 14

P = 1 (дуга 14-1)

Z = 20 - 14 + 2 * 1 = 8 – Цикломатическое число

m1: 10-**12** = 1

m2: 9-10-**12**-**13** = 2

m3: **1**-3-9-10-**12**-**13**-14 = 3

m4: **1**-**2**-4-**6**-9-10-**12**-**13**-14 = 5

m5: **1**-**2**-4-**6**-**7**-11-**13**-14 = 5

m6: **1**-**2**-4-**6**-**7**-8-11-**13**-14 = 5

m7: **1**-**2**-**5**-**7**-11-**13**-14= 5

m8: **1**-**2**-**5**-8-11-**13**-14= 4

S = 1+2+3+5+5+5+5+4 = ***30*** – Сложность по второму критерию

3. Автоматический расчет

Граф для программы представлен в виде:

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

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

Результаты работы представлены на рис. 2, рис. 3 и рис. 4

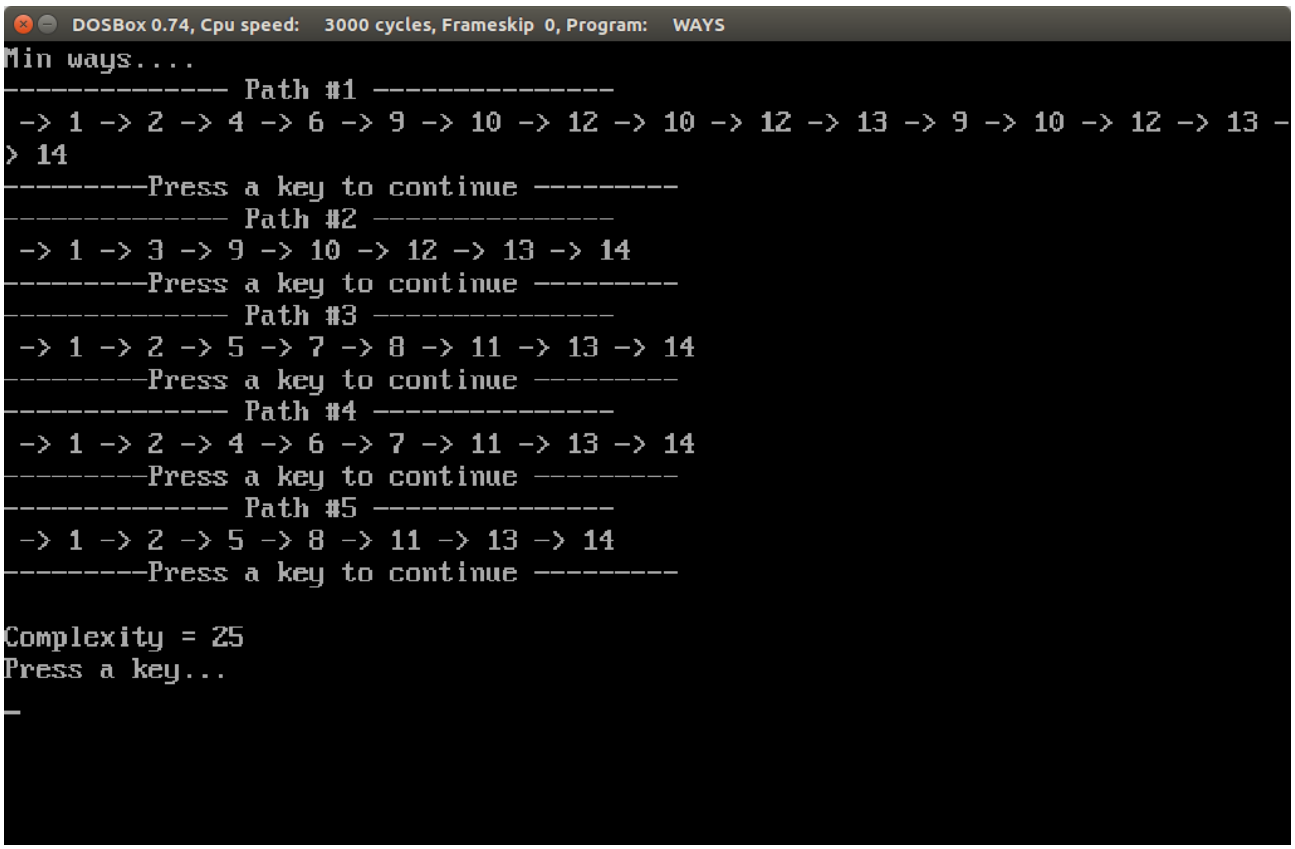


Рисунок 2 - Расчет по первому критерию

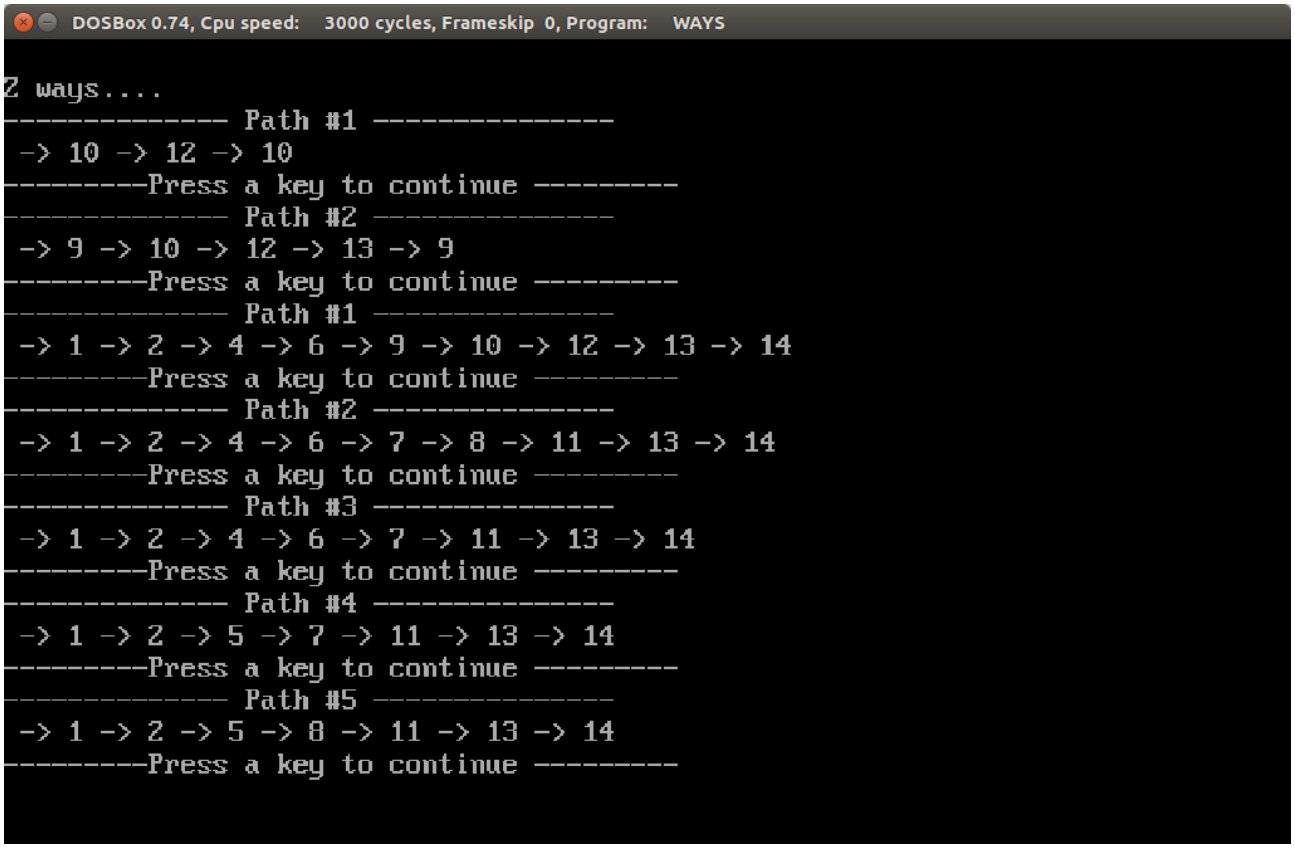


Рисунок 3 - Расчет по второму критерию

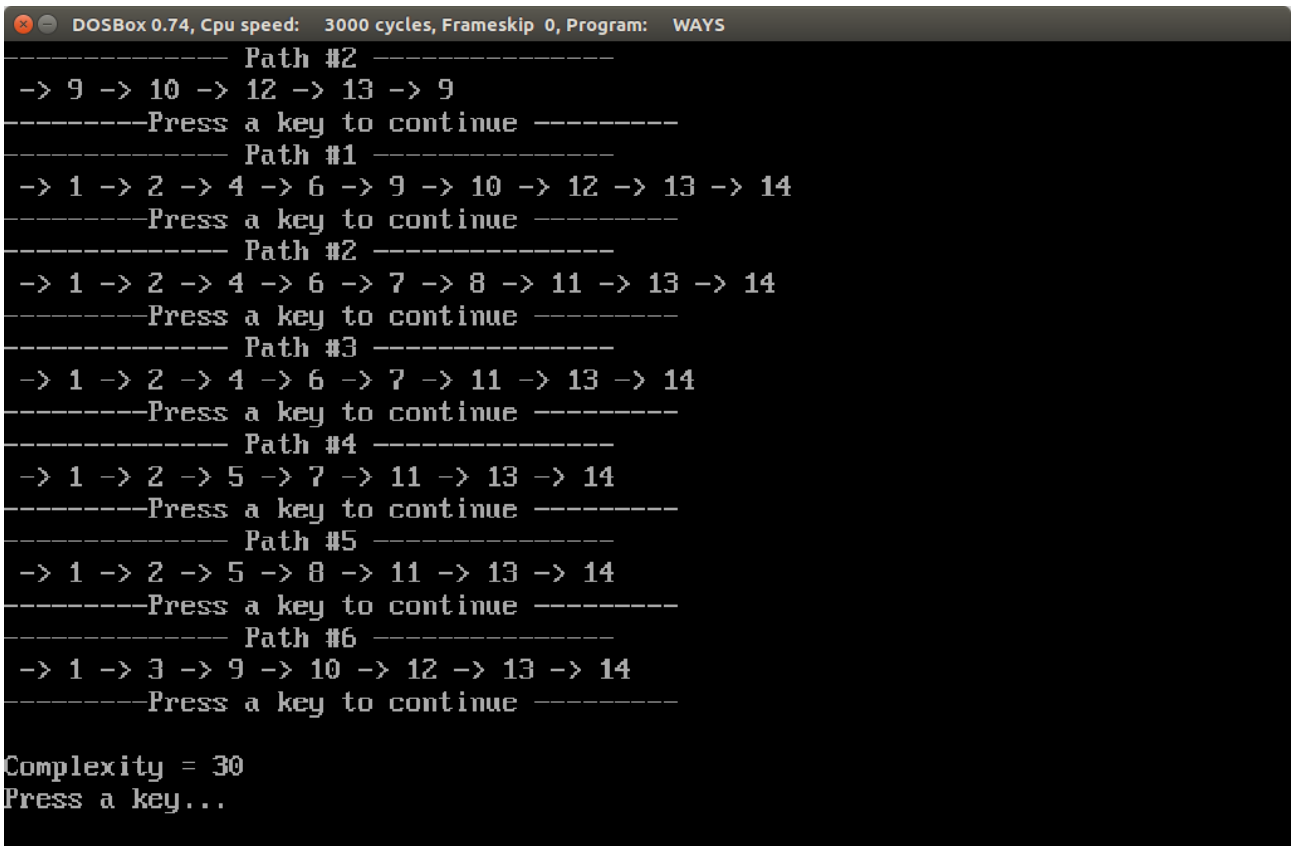


Рисунок 4 - Расчет по второму критерию (продолжение)

Сложности совпали с ручным расчетом

4. Код из ЛР №1:

```
float x, er, ec;
unsigned char done;
float erf(float x)
  /* infinite series expansion of the Gaussian error function */
{
  static const float sqrtpi = 1.7724538;
  static const float tol = 1.0E-4;
  float x2, sum, sum1, term;
  int i;
  float erf_result;

  x2 = x * x;
  sum = x;
  term = x;
  i = 0;
  do {
```

```
    i = i + 1;
    sum1 = sum;
    term = 2.0 * term * x2 / (1.0 + 2.0 * i);
    sum = term + sum1;
  } while (term >= tol * sum);
  erf_result = 2.0 * sum * exp(-x2) / sqrtpi;
  return erf_result;
}     /* erf */

float erfc(float x)
  /* complement of error function */
{
  static const float sqrtpi = 1.7724538;
  int terms = 12;
  float x2, u, v, sum;
  int i;
  float erfc_result;

  x2 = x * x;
  v = 1.0 / (2.0 * x2);
  u = 1.0 + v * (terms + 1.0);
  for( i = terms; i >= 1; i --)
  {
    sum = 1.0 + i * v / u;
    u = sum;
  }
  erfc_result = exp(-x2) / (x * sum * sqrtpi);
  return erfc_result;
}       /* ercf */

int main()
{        /* main */
  done = 0;
  x = 2;
  do {
    if (x < 0.0)
      done = 1;
    else
    {
      if (x == 0.0)
      {
        er = 0.0;
        ec = 1.0;
      }
      else if (x < 1.5)
      {
        er = erf(x);
        ec = 1.0 - er;
      }
      else
      {
```

```
        ec = erfc(x);
        er = 1.0 - ec;
      }  /* if */
      x = x - 1;
    }   /* if */
  } while (!done);
  return 0;
}
```

Графовое представление на рис. 5

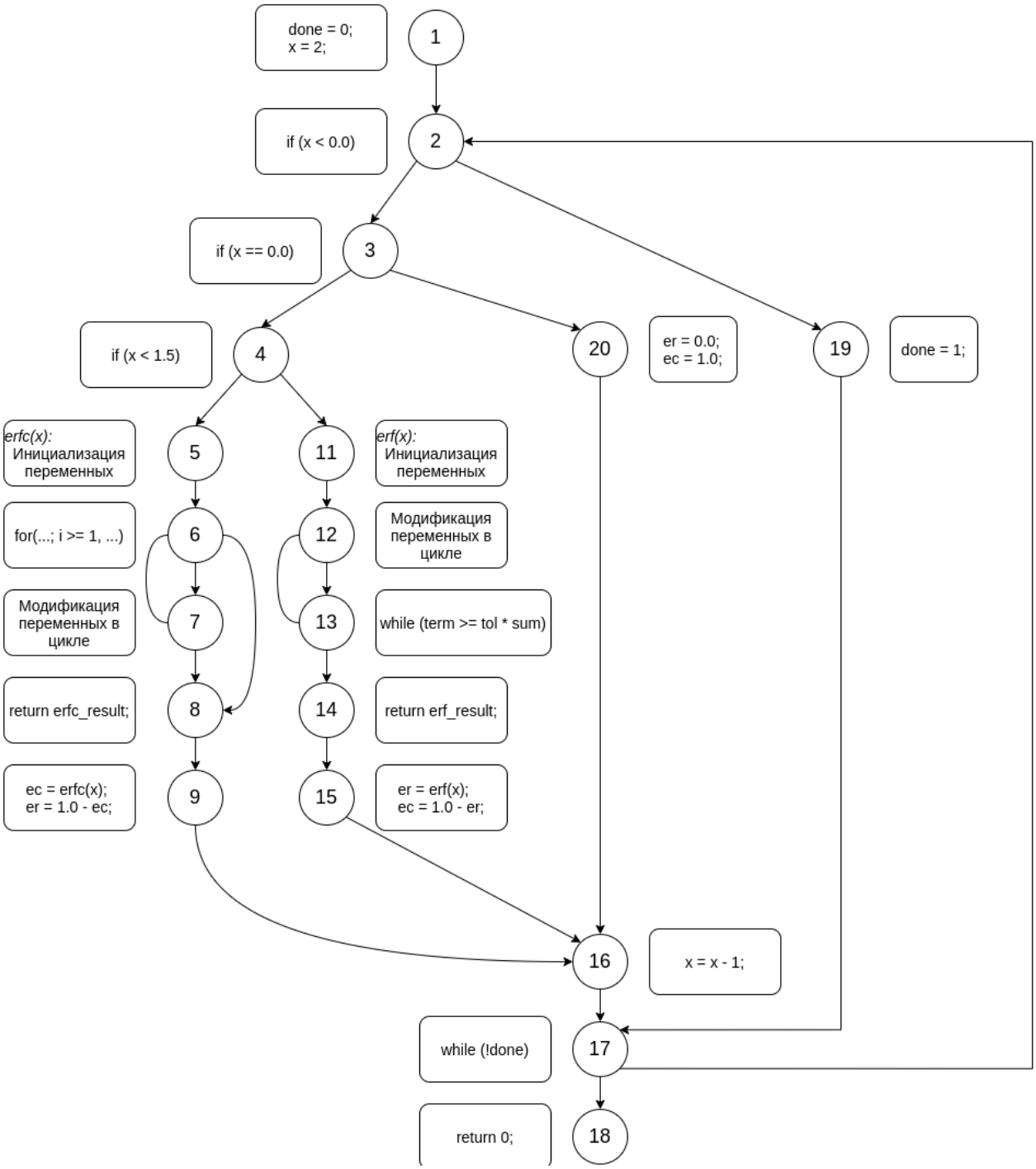


Рисунок 5 - Графовое представление программы из ЛР №1

Так как граф, представленный на рис. 5 не структурированный, программа ways.exe не может его обработать (6-7-8). Структурированная версия представлена на рис. 6

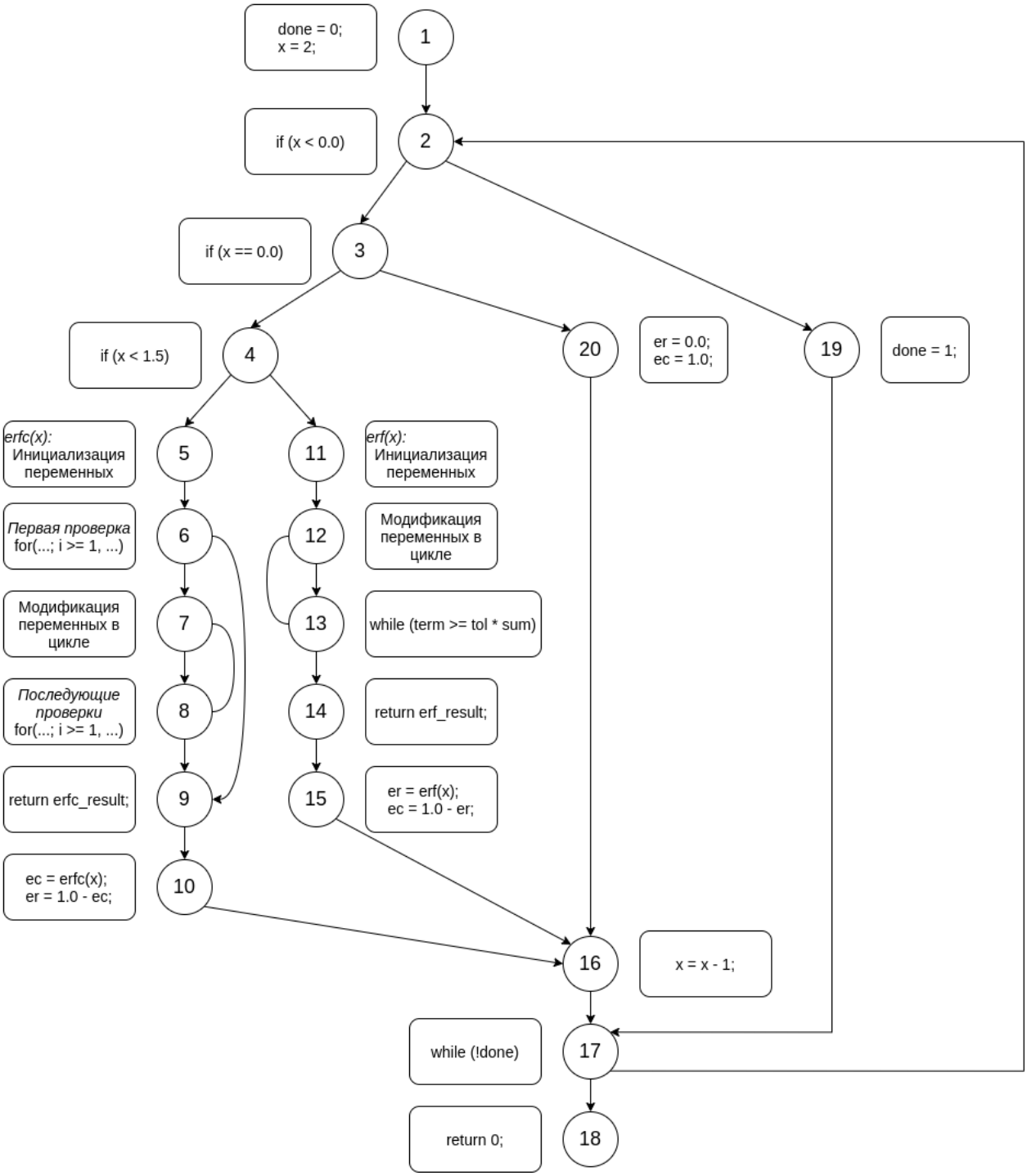


Рисунок 6 — Структурированный граф для программы из ЛР №1

5. Ручной подсчет

5.1. Первый критерий

M1: 1-**2**-**3**-**4**-5-**6**-7-**8**-7-**8**-9-10-16-**17**-

-**2**-**3**-**4**-5-**6**-9-10-16-**17**-

-**2**-**3**-**4**-11-12-**13**-12-**13**-14-15-16-**17**-
-**2**-**3**-20-16-**17**-
-**2**-19-**17**-18 = 23

S = 23

5.2. Второй критерий
Так как граф структурированный.

Так как граф структурированный.

$n_в = 7$

Z = 1 + 7 = 8

Циклы:
m1: 7-**8** = 1
m2: 12-**13** = 1
m3: **2**-19-**17** = 2

Пути:
m4: 1-**2**-**3**-**4**-5-**6**-7-**8**-9-10-16-**17**-18 = 6
m5: 1-**2**-**3**-**4**-5-**6**-9-10-16-**17**-18 = 5
m6: 1-**2**-**3**-**4**-11-12-**13**-14-15-16-**17**-18 = 5
m7: 1-**2**-**3**-20-16-**17**-18= 3
m8: 1-**2**-19-**17**-18= 2

S = 1 + 1 + 2 + 6 + 5 + 5 + 3 + 2 = 25

6. Автоматический подсчет
Представление для ways.exe:

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

```
arc(12,13);
arc(13,12);
arc(7,8);
arc(8,7);
arc(6,9);
arc(13,14);
arc(20,16);
arc(19,17);
arc(17,2);
arc(8,9);
arc(14,15);
arc(9,10);
arc(10,16);
arc(15,16);
arc(16,17);
arc(17,18);
}
```

Расчет программой представлен на рисунках 7, 8 и 9.

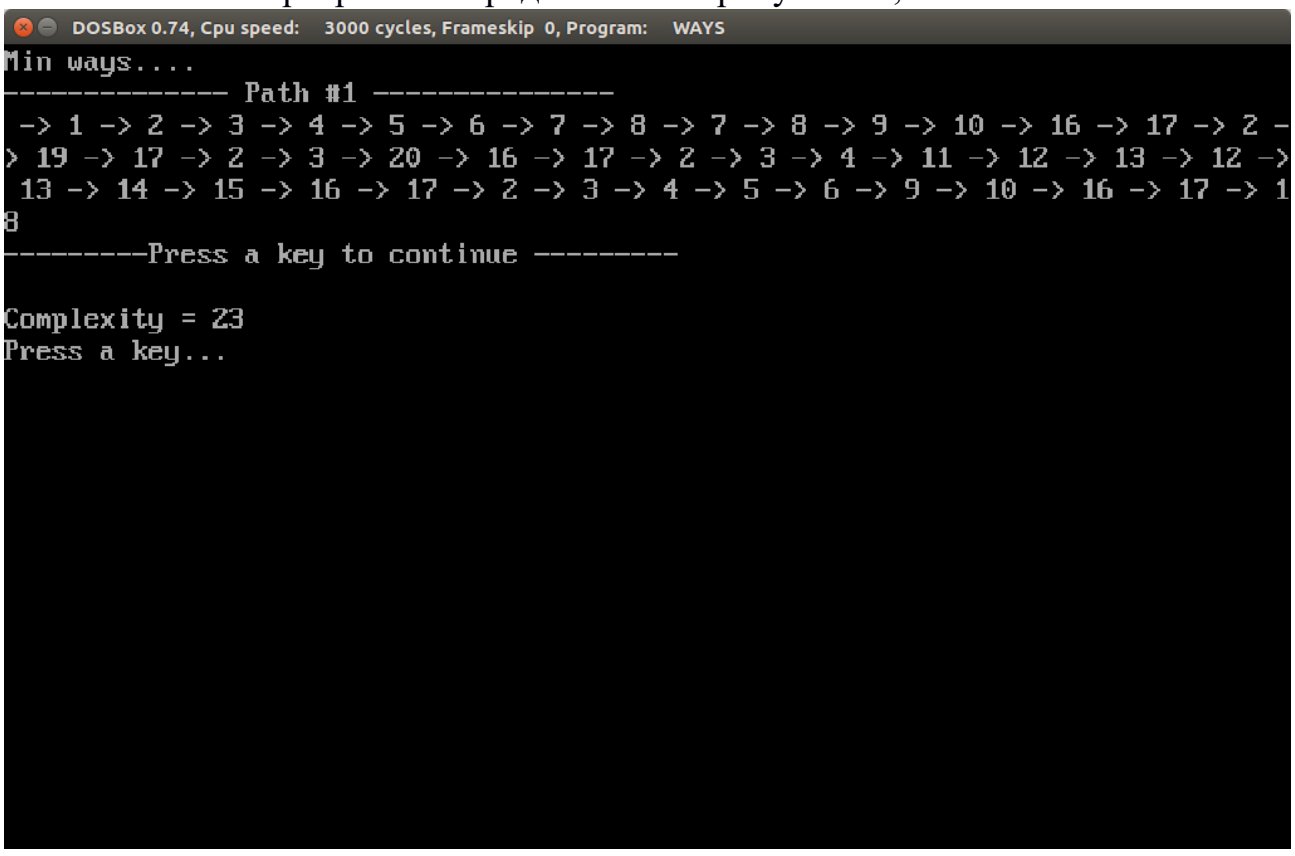


Рисунок 7 - Расчет по первому критерию

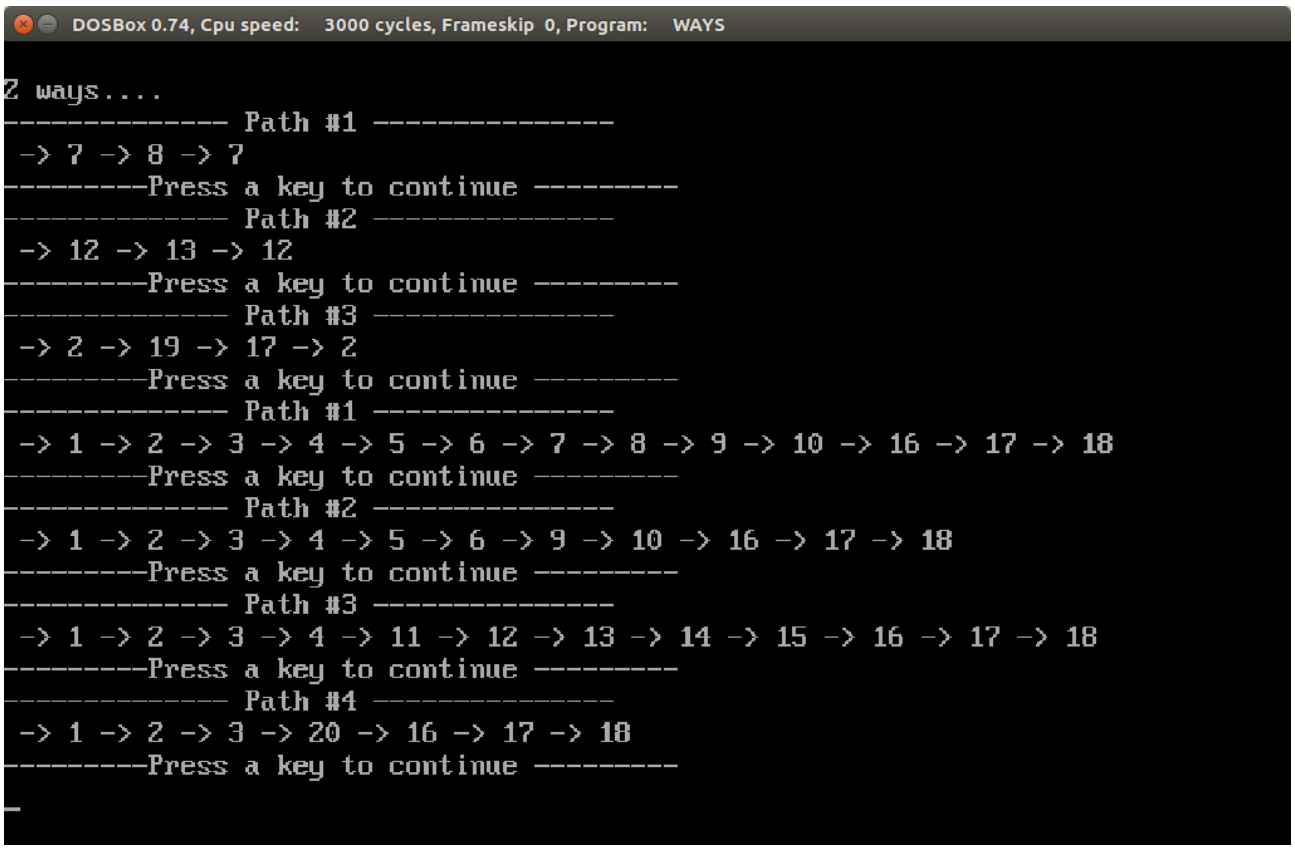
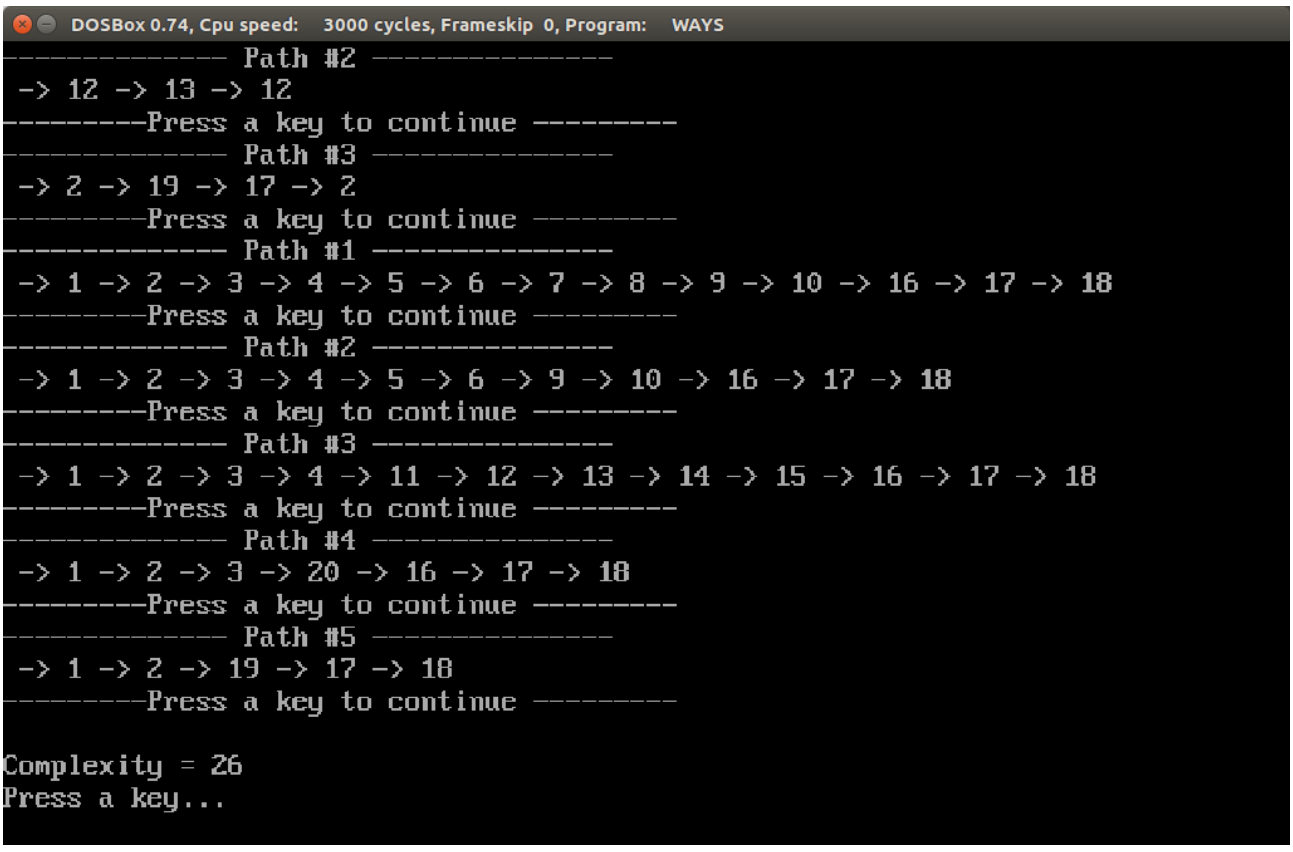


Рисунок 8 - Расчет по второму критерию

Рисунок 9 - Расчет по второму критерию (продолжение)

Для критерия 2 в ручном расчет сложность меньше на 1, так как циклы



строились как в примере из методических указаний (то есть без повторения первого узла в конце). Если бы цикл строился как в программе 2-19-17-2 (а не 2-19-17, как в указаниях), то сложность такого маршрута была бы на 1 больше и сложности бы сошлись.

Выводы.

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 автоматизировано с помощью предоставленной программы.