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 _____ Шарапенков И.И.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2021

Цель работы:

Изучение структурной сложности графовых моделей программ и метрик ее оценки.

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, выбираемой из файла zadan_struct.doc в соответствии с номером в списке группы;
- Программа из 1-ой лабораторной работы (управляющий граф составить самостоятельно).

Оцениваемые характеристики структурной сложности:

- Числа учитываемых маршрутов проверки программы для заданного критерия;
- Цикломатическое число;
- Суммарное число ветвлений по всем маршрутам – оценка структурной сложности.

Ход работы

1. Был выбран вариант №18 для первой программы, у которой необходимо определить структурную сложность. Управляющий граф первой программы приведен на Рисунке 1:

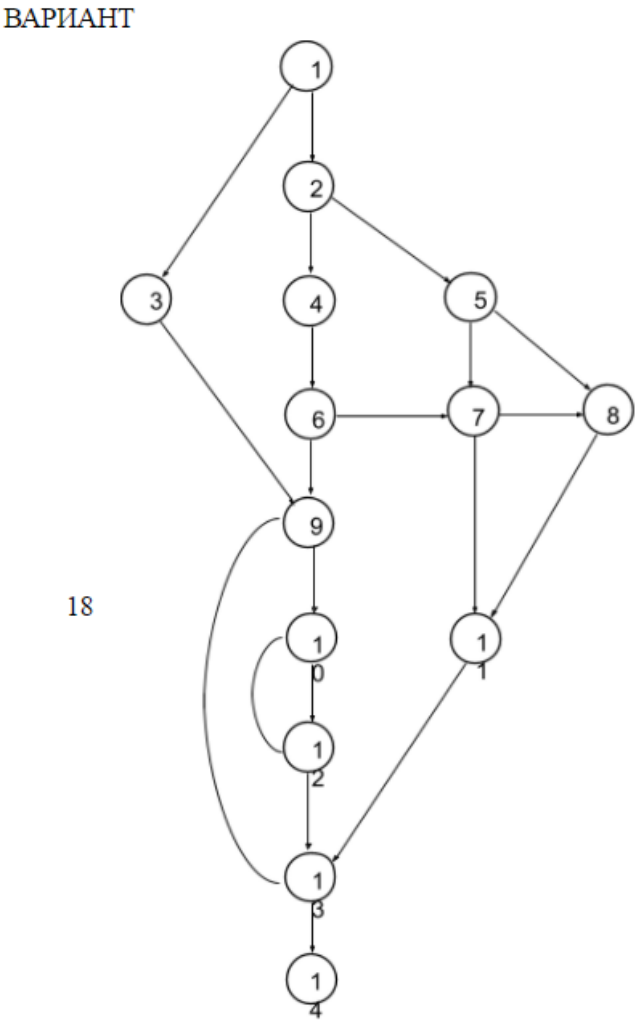

Рисунок 1: Управляющий граф первой программы для определения структурной сложности

2. Была рассчитана структурная сложность первой программы вручную по первому критерию, а именно по минимальному покрытию вершин и дуг графа управления.

   M1: **1**-3-9-10-**12**-10-**12**-**13**-9-10-**12**-**13**-14 (6 ветвлений)

M2: **1**-**2**-4-**6**-9-10-**12**-**13**-14 (5 ветвлений)

M3: **1**-**2**-**5**-8-11-**13**-14 (4 ветвления)

M4: **1**-**2**-**5**-**7**-8-11-**13**-14 (5 ветвлений)

M5: **1**-**2**-4-**6**-**7**-11-**13**-14 (5 ветвлений)

Количество маршрутов: $M = 5$

Сложность: $S = 6 + 5 + 4 + 5 + 5 = 25$

3. Было рассчитано вручную цикломатическое число графа первой программы для дальнейшего применения второго критерия. Для данной программы число вершин в графе равно 14, число дуг – 20, число связных компонент графа 1, тогда: $Z = Y - N + 2 * P = 20 - 14 + 2 * 1 = 8$, то есть цикломатическое число равно 8.
4. Для управляющего графа первой программы вручную были построены 8 линейно-независимых циклов и линейно-независимых маршрутов, после чего была подсчитана структурная сложность по второму критерию.

   M1: 10-**12** (1 ветвление)

   M2: 9-10-**12**-**13** (2 ветвления)

   M3: **1**-3-9-10-**12**-**13**-14 (3 ветвления)

   M4: **1**-**2**-4-**6**-9-10-**12**-**13**-14 (5 ветвлений)

   M5: **1**-**2**-4-**6**-**7**-11-**13**-14 (5 ветвлений)

   M6: **1**-**2**-**5**-**7**-8-11-**13**-14 (5 ветвлений)

   M7: **1**-**2**-**5**-8-11-**13**-14 (4 ветвления)

   M8: **1**-**2**-4-**6**-**7**-8-11-**13**-14 (5 ветвлений)

   Сложность: $S = 1 + 2 + 3 + 5 + 5 + 5 + 4 + 5 = 30$
5. Для первой программы была подсчитана структурная сложность по двум критериям с помощью программы ways.exe. Результаты расчета структурной сложности по первому критерию приведены на Рисунке 2, по второму критерию на Рисунке 3:

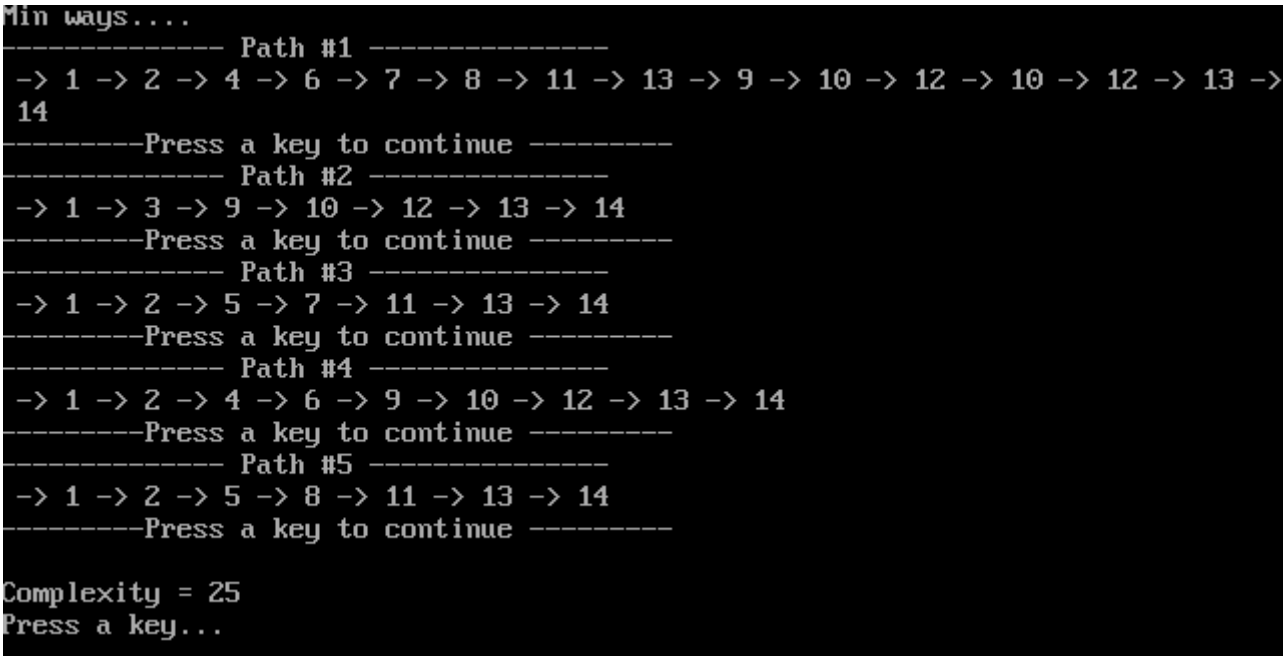

Рисунок 2: Программный расчет структурной сложности первой программы по первому критерию

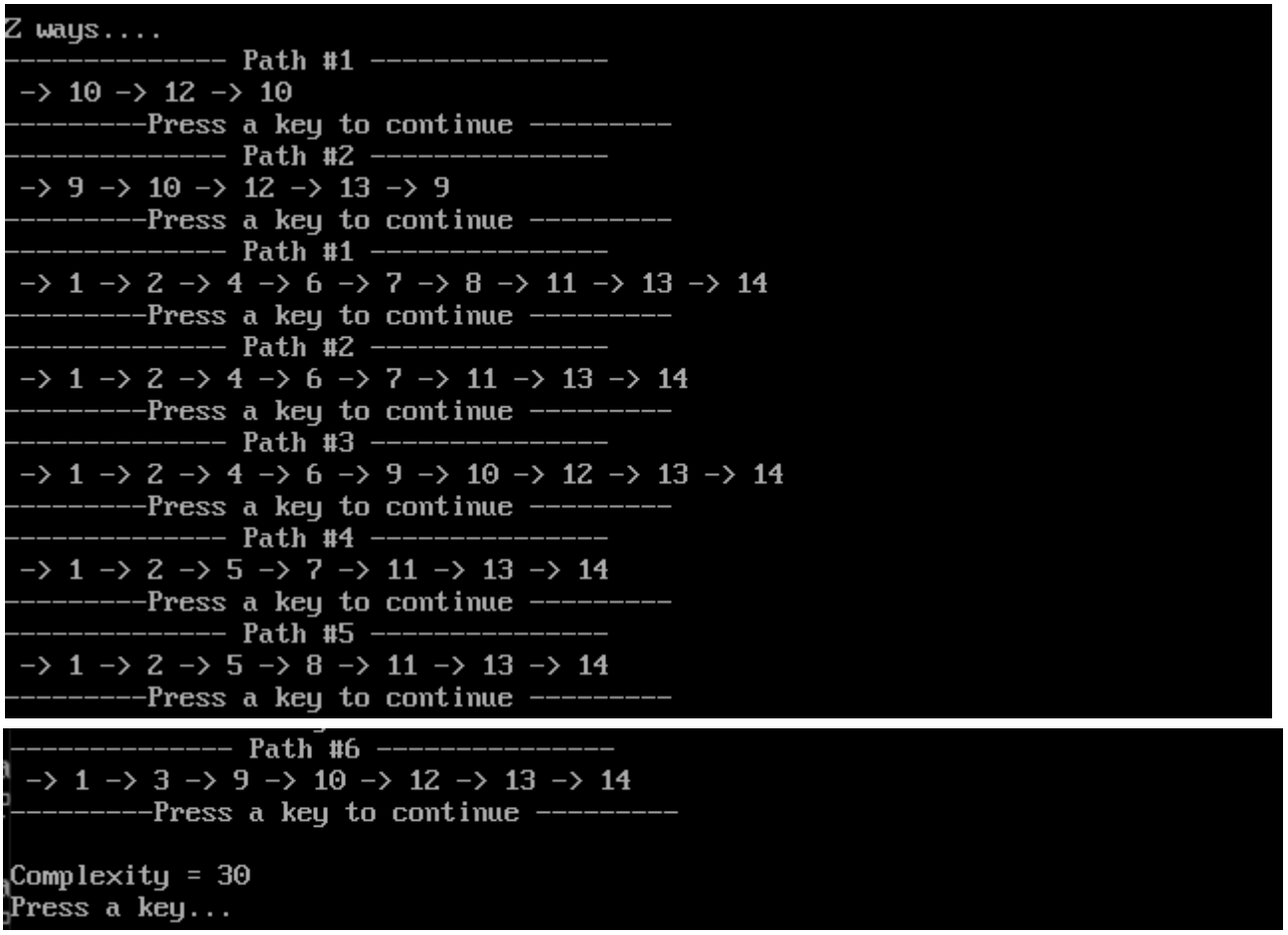

Рисунок 3: Программный расчет структурной сложности первой программы по второму критерию

6. Для программы из первой лабораторной работы был составлен управляющий граф. Данный управляющий граф представлен на Рисунке 4:

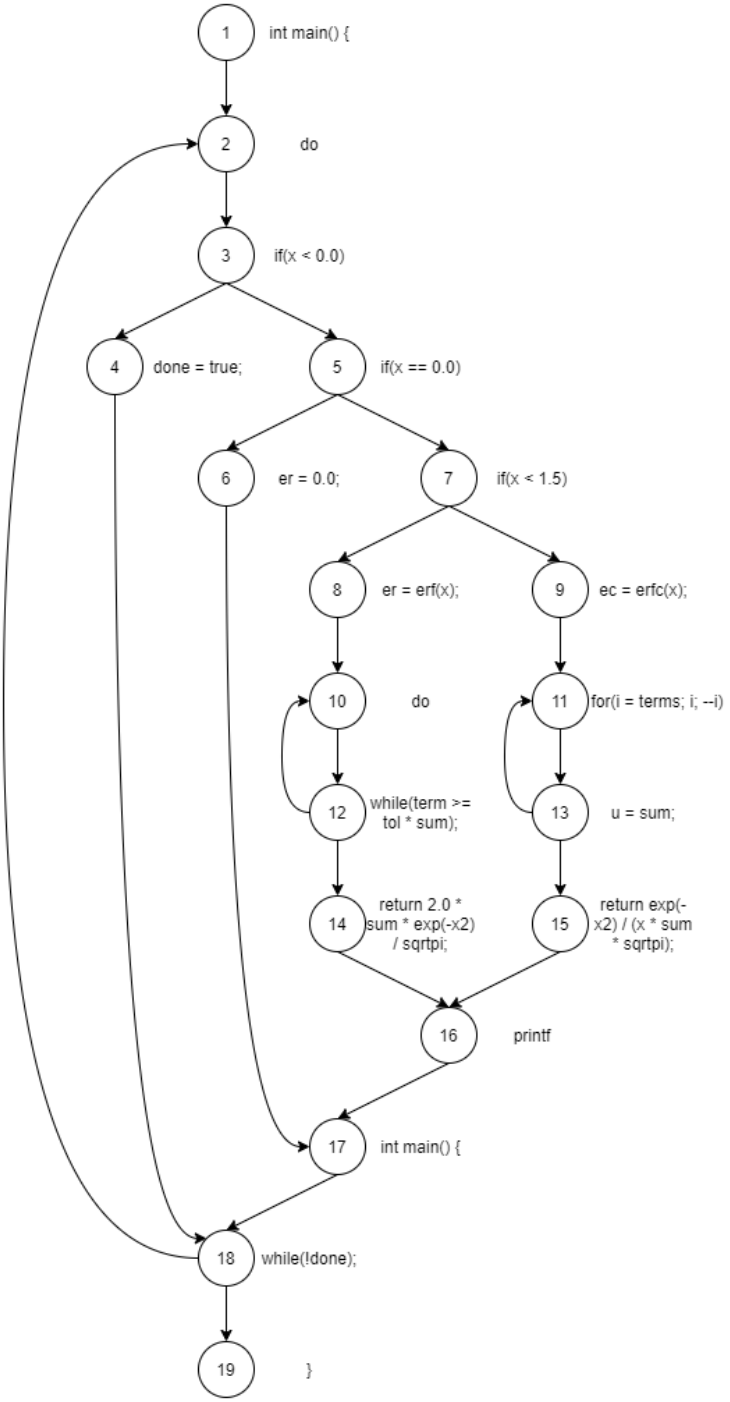

Рисунок 4: Управляющий граф программы из первой лабораторной работы для определения структурной сложности

7. Была рассчитана структурная сложность программы из первой лабораторной работы вручную по первому критерию, а именно по минимальному покрытию вершин и дуг графа управления.

   M1:

   1-2-**3**-4-**18**-2-**3**-**5**-6-17-**18**-2-**3**-**5**-**7**-8-10-**12**-10-**12**-14-16-17-**18**-2-**3**-**5**-**7**-9-11-**13**-11-**13**-15-16-17-**18**-19 (17 ветвлений)

   Количество маршрутов: $\mathrm{M} = 1$

   Сложность: $S = 17$

8. Было рассчитано вручную цикломатическое число графа программы из первой лабораторной работы для дальнейшего применения второго критерия. Для данной программы число вершин в графе равно 19, число дуг – 24, число связных компонент графа 1, тогда: $Z = Y - N + 2 * P = 24 - 19 + 2 * 1 = 7$, то есть цикломатическое число равно 7.

9. Для управляющего графа программы из первой лабораторной работы вручную были построены 7 линейно-независимых цикла и линейно – независимых маршрута, после чего была подсчитана структурная сложность по второму критерию.

   M1: 10-**12** (1 ветвление)

   M2: 11-**13** (1 ветвление)

   M3: 2-**3**-4-**18** (2 ветвления)

   M4: 1-2-**3**-**5**-6-17-**18**-19 (3 ветвления)

   M5: 1-2-**3**-**5**-**7**-8-10-**12**-14-16-17-**18**-19 (5 ветвлений)

   M6: 1-2-**3**-**5**-**7**-9-11-**13**-15-16-17-**18**-19 (5 ветвлений)

   M7: 1-2-**3**-4-**18**-19 (2 ветвления)

   Сложность: $S = 1 + 1 + 2 + 3 + 5 + 5 + 2 = 19$

10. Для программы из первой лабораторной работы была подсчитана структурная сложность по двум критериям с помощью программы

ways.exe. Результаты расчета структурной сложности по первому критерию приведены на Рисунке 5, по второму критерию на Рисунке 6.

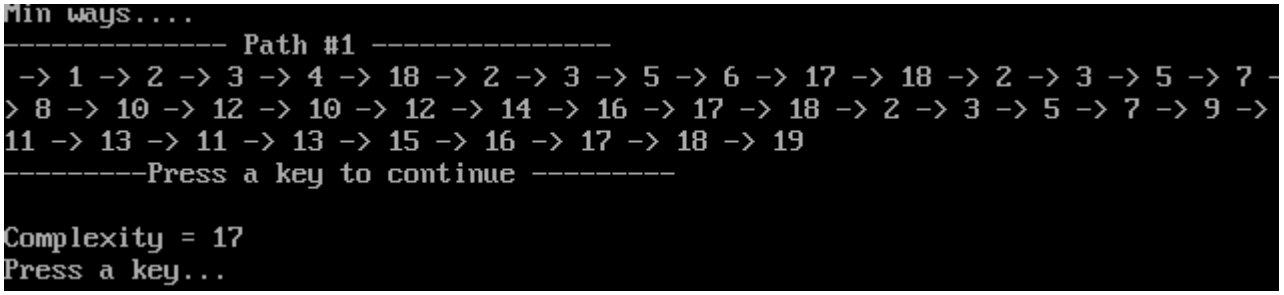

Рисунок 5: Программный расчет структурной сложности программы из первой лабораторной работы по первому критерию

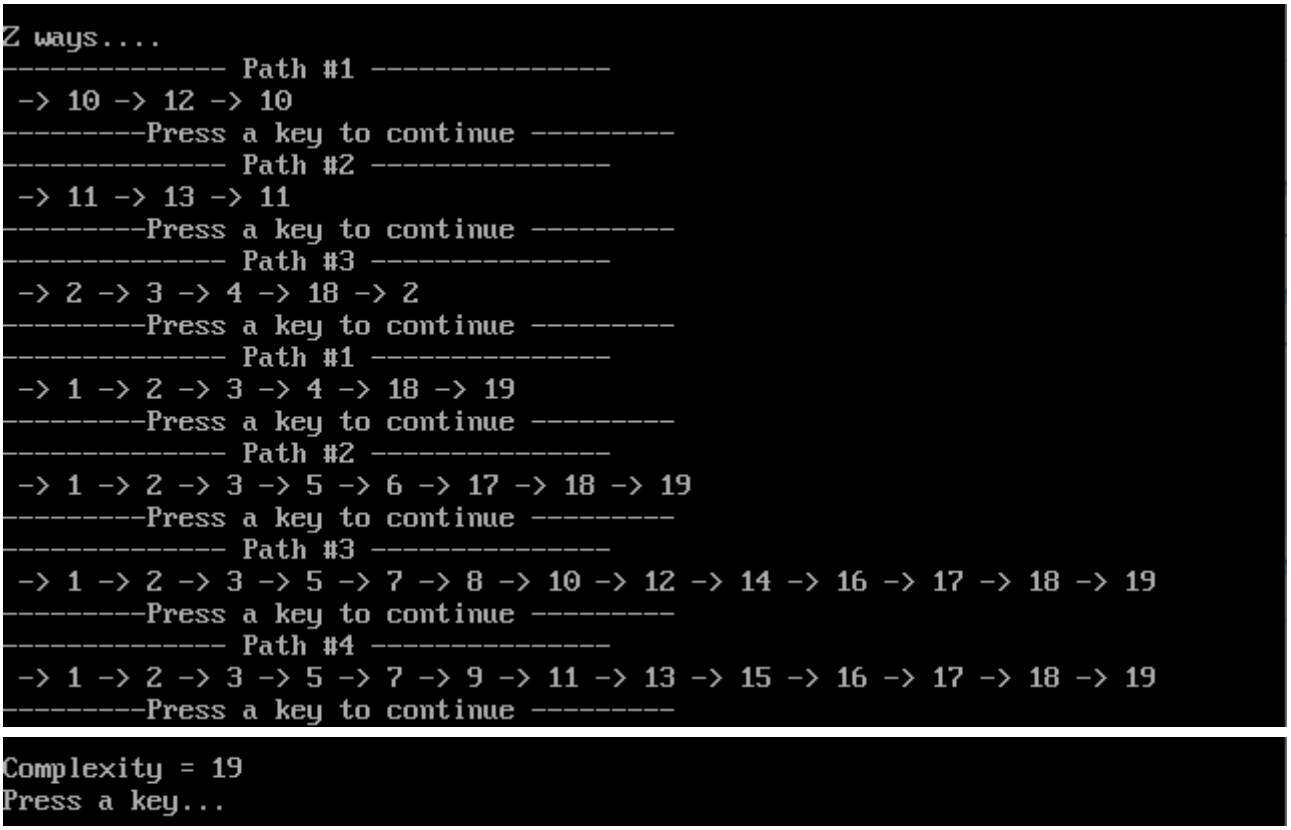

Рисунок 6: Программный расчет структурной сложности программы из первой лабораторной работы по второму критерию

Выводы.

В ходе выполнения лабораторной работы были изучены методы оценки структурной сложности программы на основе его управляющего графа, была рассчитана структурная сложность двух программ по двух критериям: минимальное покрытие дуг графа и выбор маршрутов на основе цикломатического числа графа. Расчеты были проведены как ручным, там и программным способом.

ПРИЛОЖЕНИЕ А.

КОД ПРОГРАММЫ НА ЯЗЫКЕ ПАСКАЛЬ.

```
PROGRAM SIMP1;
{ INTEGRATION BY SIMPSON'S METHOD }
CONST TOL = 1.0E-6;
VAR SUM,UPPER,LOWER      : REAL;
FUNCTION FX(X: REAL): REAL;
BEGIN
    FX:=EXP(-X/2)
END;
{ FUNCTION FX }
FUNCTION DFX(X: REAL): REAL;
BEGIN
    DFX:=-(EXP(-X/2))/2
END;
{ FUNCTION DFX }
PROCEDURE SIMPS(LOWER,UPPER,TOL      : REAL; VAR SUM    : REAL);
{ NUMERICAL INTEGRATION BY SIMPSON'S RULE }
{ FUNCTION IS FX, LIMITS ARE LOWER AND UPPER }
{ WITH NUMBER OF REGIONS EQUAL TO PIECES }
{ PARTITION IS DELTA_X, ANSWER IS SUM }
VAR I : INTEGER;
X,DELTA_X,EVEN_SUM, ODD_SUM,END_SUM, END_COR,SUM1 : REAL;
PIECES: INTEGER;
BEGIN
    PIECES:=2;
    DELTA_X:=(UPPER-LOWER)/PIECES;
    ODD_SUM:=FX(LOWER+DELTA_X);
    EVEN_SUM:=0.0;
    END_SUM:=FX(LOWER)+FX(UPPER);
    END_COR:=DFX(LOWER)-DFX(UPPER);
    SUM:=(END_SUM+4.0*ODD_SUM)*DELTA_X/3.0;
    REPEAT
        PIECES:=PIECES*2;
        SUM1:=SUM;
        DELTA_X:=(UPPER-LOWER)/PIECES;
        EVEN_SUM:=EVEN_SUM+ODD_SUM;
        ODD_SUM:=0.0;
        FOR I:=1 TO PIECES DIV 2 DO
            BEGIN
                X:=LOWER+DELTA_X*(2.0*I-1.0);
                ODD_SUM:=ODD_SUM+FX(X)
            END;
SUM:=(7.0*END_SUM+14.0*EVEN_SUM+16.00*ODD_SUM+END_COR*DELTA_X)*DELTA_X/15.0;
    UNTIL (SUM<>SUM1) AND (ABS(SUM-SUM1)<=ABS(TOL*SUM))
END; { SIMPS }
BEGIN { MAIN PROGRAM }
    LOWER:=1.0;
    UPPER:=9.0;
    SIMPS(LOWER,UPPER,TOL,SUM);
    WRITELN;
END.
```