Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Метрология, стандартизация и сертификация (в информационных технологиях)

ОТЧЕТ

по лабораторной работе №2

Тема работы: Метрики сложности потока управления программ

Выполнили

студенты: гр. 251003 Зданович Н.В.

Дедов Н.Ю.

Проверил: Болтак С.В.

Минск 2024

Тестовый код

fun main() {

var maxValue = Int.MIN\_VALUE

var sum = 0

var count = 0

var k = 0

if (count==0) {

println("Выполнение при правильных исходных данных")

if (i % 2 == 0) {

println("Выполнение условия при четном числе")

for (i in 1..3) {

println("Цикл for, уровень $i")

while (count < 3) {

println("Начало цикла while")

if (i == 2) {

println("Специальная обработка для числа 2")

k = 2

}

else {

println("Обработка иных чисел")

k = 3 }

when (i) {

1 -> {

println(" Выполнение when: число 1")

k += 1

}

2 -> {

println(" Выполнение when: число 2")

k += 2

}

3 -> {

println(" Выполнение when: число 3")

k += 3

}

4 -> {

println(" Выполнение when: число 4")

k += 4

}

else -> println(" Выполнение when: другое число")

}

count++

}

println("Выход из цикла while")

}

println("Выход из цикла for")

}

else {

println("Нечетное число $i")

k = 1

}

println("Конец проверки на четность или нечетность")

}

else {

println ("Проверьте исходные данные")

k = -1

}

println("Результат $k")

if (count >= 3) {

println("Специальная обработка для итерации 2 внешнего while")

var j = 0

while (j < 2) {

j++

println("Вложенный цикл while, итерация $j")

sum += j \* i

}

}

println("Максимальное значение: $maxValue")

println("Сумма: $sum")

}

Расчет базовых метрик Джилба для данной программы.

Таблица 1

|  |  |  |
| --- | --- | --- |
| **Оператор** | ***f1j*** | |
| main | 1 | |
| = | 9 | |
| if | 4 | |
| == | 3 | |
| println | 21 | |
| % | 1 | |
| for | 1 | |
| .. | 1 | |
| while | 2 | |
| < | 2 | |
| when | 1 | |
| -> | 5 | |
| += | 5 | |
| ++ | 2 | |
| - | 1 | |
| >= | 1 | |
| \* | 1 | |
|  | | ***N =* 61** | |

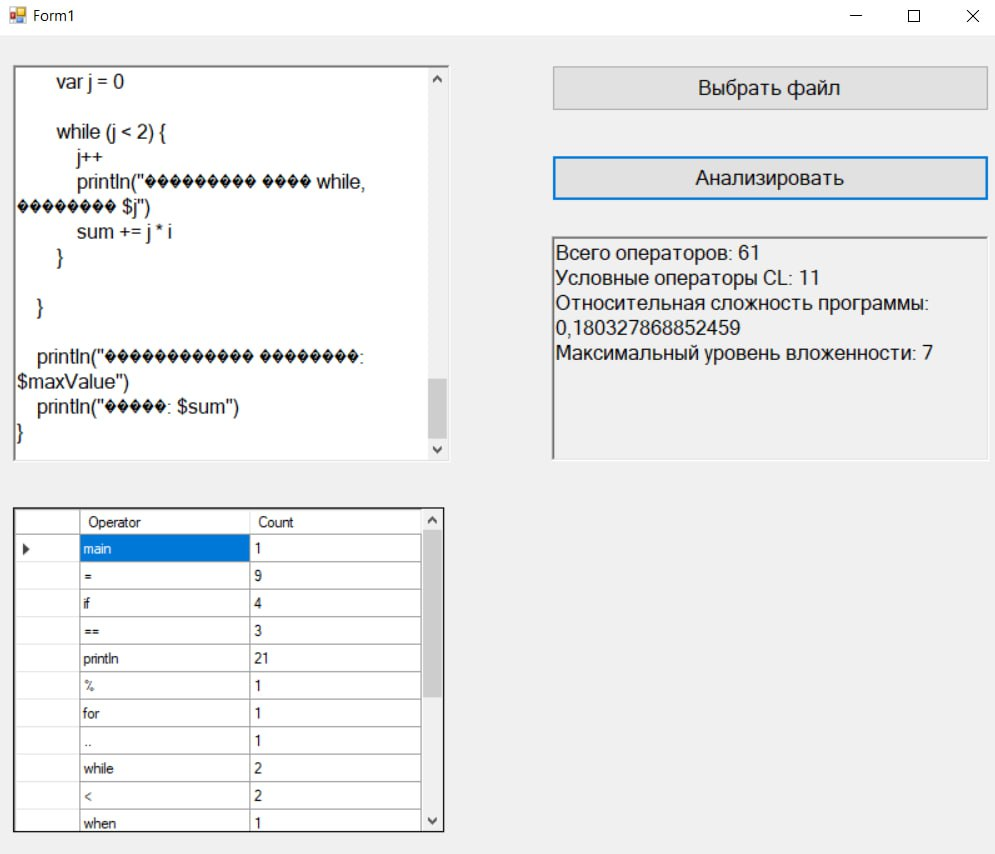
Максимальный уровень вложенности условногооператора **CLI** = 7.

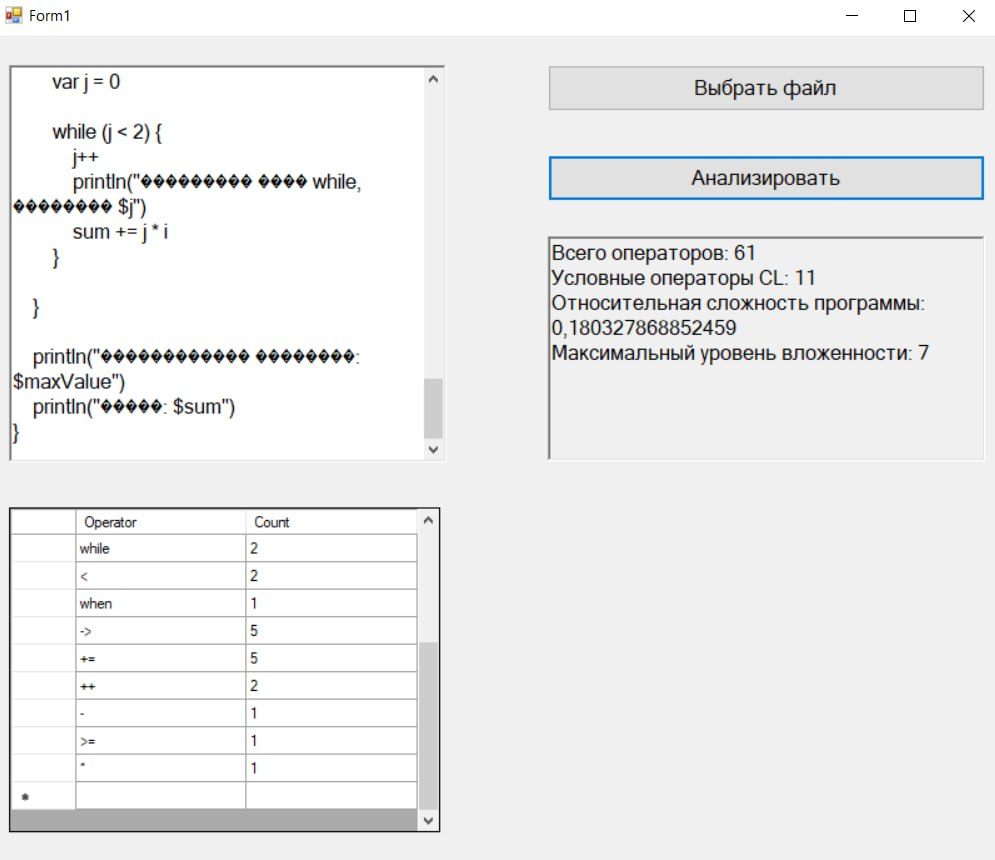
Общее количество операторов программы ***N*** = 61.

Количество условных операторов ***CL*** = 11

Насыщенность программы условными операторами **cl =** 11 / 61 = 0.18

cl = CL / N





Расчет метрики граничных значений для данной программы.

var count = 0

var k = 0

if (count==0) {

println("Выполнение при правильных исходных данных")

if (i % 2 == 0) {

println("Выполнение условия при четном числе")

for (i in 1..3) {

println("Цикл for, уровень $i")

while (count < 3) {

println("Начало цикла while")

if (i == 2) {

println("Специальная обработка для числа 2")

k = 2

}

else {

println("Обработка иных чисел")

k = 3 }

when (i) {

1 -> {

println(" Выполнение when: число 1")

k += 1

}

2 -> {

println(" Выполнение when: число 2")

k += 2

}

3 -> {

println(" Выполнение when: число 3")

k += 3

}

4 -> {

println(" Выполнение when: число 4")

k += 4 }

else -> println(" Выполнение when: другое число")

}

count++

}

println("Выход из цикла while")

}

println("Выход из цикла for") }

else {

println("Нечетное число $i")

k = 1 }

println("Конец проверки на четность или нечетность") }

else {

println ("Проверьте исходные данные")

k = -1 }

println("Результат $k")



Рисунок 1 – Схема алгоритма (часть 1)



Рисунок 2 – Схема алгоритма (часть 2)

Относительная граничная сложность программы *So*

****

*Sa —* абсолютная граничная сложность программы;

*ʋ –* общее число вершин графа программы.

Таблица 2

**Свойства подграфов программы**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Свойства подграфов программы | Номер вершины выбора | | | | | |
| 4 | 6 | 8 | 10 | 12 | 15 |
| Номера вершин перехода | 5, 30 | 7, 32 | 9, 27 | 11, 26 | 13, 34 | 16,18,20,22,24 |
| Скорректированная сложность вершины выбора | 27 | 24 | 19 | 16 | 5 | 10 |
| Номера вершин подграфа | 5-28, 30, 31 | 7-27, 32,33 | 9-26 | 11-25 | 13,14,34,35 | 16,17,18,19,20,21,22,23,24 |
| Номер нижней границы подграфа | 29 | 28 | 27 | 26 | 15 | 25 |

Таблица 3

**Скорректированные сложности вершин графа программы (рис. 1, рис. 2)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Номер вершины графа программы |  |  |  |  |  |  |  |  |  |  |  |  | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| Скорректированная сложность вершины графа | 1 | 1 | 1 | 27 | 1 | 24 | 1 | 19 | 1 | 16 | 1 | 5 | 1 | 1 | 10 | 1 | 1 | 1 | 1 | 1 | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Номер вершины графа программы | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 |  |
| Скорректированная сложность вершины графа | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | ***Sa* = 130** | |

*So*= l – (36 – 1)/130 = 0,731.