Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей
Кафедра программного обеспечения и информационных технологий
Дисциплина «Метрология, стандартизация и сертификация»

ОТЧЕТ
к лабораторной работе №2
на тему:
«МЕТРИКИ СЛОЖНОСТИ ПОТОКА УПРАВЛЕНИЯ ПРОГРАММ»
БГУИР 6-05-0612-02

Выполнили студенты группы 353503
КОХАН Артём Игоревич
ШЕМЕТКОВ Ян Игоревич

_____
(дата, подпись студента)

Проверил ассистент каф. ПОИТ
БОЛТАК Светлана Владимировна

_____
(дата, подпись преподавателя)

Минск 2025

1 ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

Задание

1 Изучить метрики сложности потока управления.

2 Взять код программы в консольном режиме (например, найти в Интернет) на указанном преподавателем языке программирования.

3 Написать свою программу (на любом языке), которая будет анализировать программу из пункта 2. На основе анализа программа должна рассчитать метрику Джилба: абсолютную сложность, относительную сложность, а также расширение метрики Джилба - максимальный уровень вложенности. Результат работы программы: вывод на экран метрики Джилба и расширение метрики Джилба.

4 В анализируемой программе выделить и распечатать часть кода с максимальным количеством ветвлений (наличие циклов обязательно). Достаточно взять часть кода с пятью-шестью вложенными циклами и несколькими операторами if. Наличие оператора выбора приветствуется. Изобразить схему данной части кода с максимальным уровнем детализации по ГОСТ 19.701-90 (например, в Visio). Это означает, что каждому блоку схемы алгоритма должен соответствовать один оператор языка программирования.

На основании составленной граф-схемы алгоритма рассчитать абсолютную Sa и относительную So граничную сложность программы по метрике граничных значений (по аналогии с примерами, приведенными в теоретических сведениях). Результаты расчетов метрики граничных значений должны быть представлены в виде таблиц, аналогичных таблицам 3 и 4.

2 ВЫПОЛНЕНИЕ РАБОТЫ

Ниже приведён код для расчёта метрик Джилбы

```
fun main() {
    // Ввод данных
    print("Введите n1: ")
    val n1 = readln().toInt()
    print("Введите n2: ")
    var n2 = readln().toInt() ; var i = 0

    while (i < n1) {
        when (n2) {
            1 -> {
                if (n1 % 2 == 0) {
                    n2 *= 2
                } else {
                    n2 /= 2
                }
            }
            2 -> {
                if (n1 % 2 == 0) {
                    n2 *= 2
                } else {
                    n2 /= 2
                }
            }
            3 -> {
                when {
                    n1 <= 1 -> n2 *= 2
                    n1 <= 2 -> n2 /= 2
                    n1 <= 3 -> n2 += 2
                    n1 <= 4 -> n2 -= 2
                    n1 == 5 -> n2++
                    else -> n2 = 0
                }
            }
        }
        i++
    }

    println("Результат: n2 = $n2")
}
```

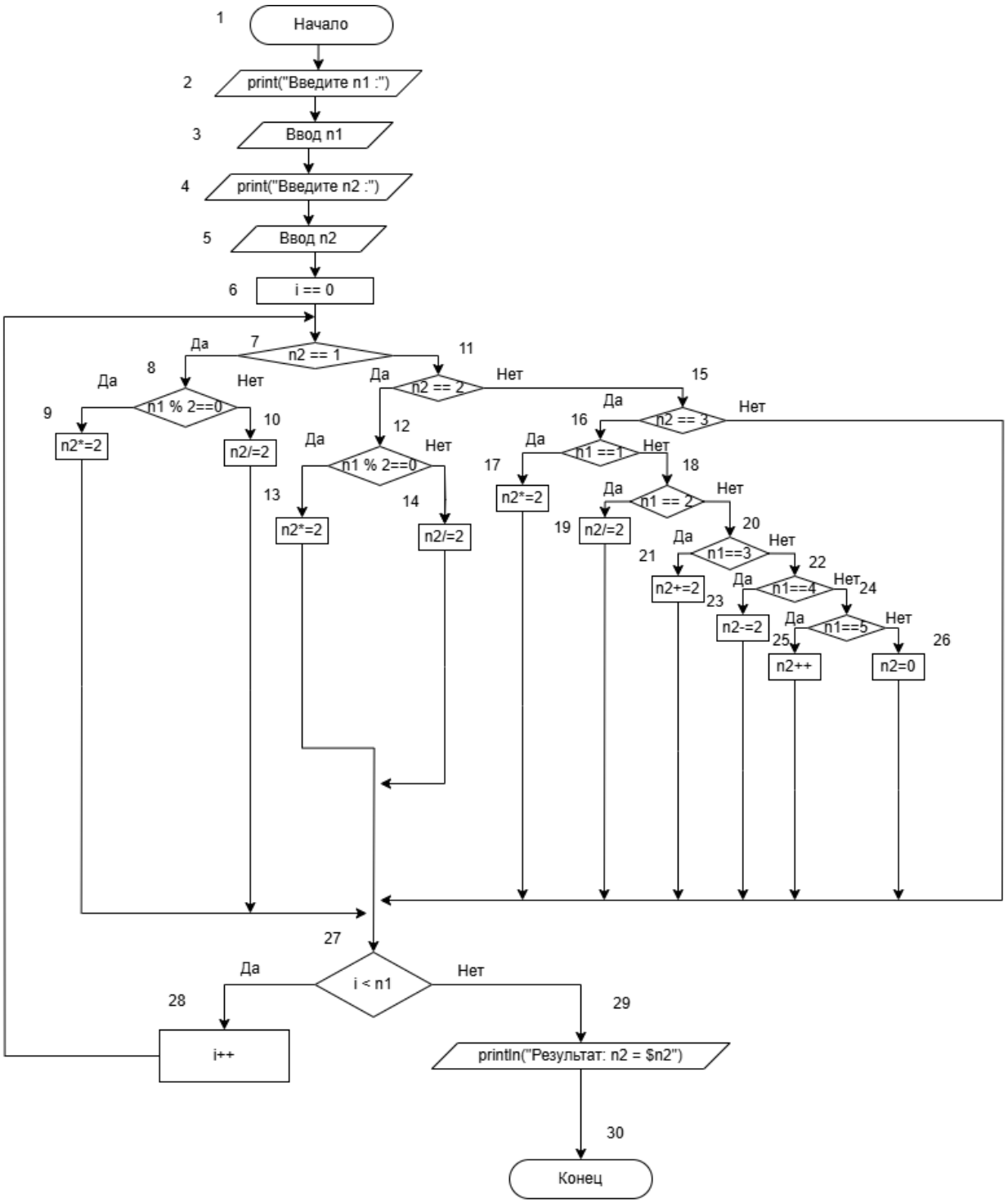

Рисунок 1 – Блок-схема с максимальной вложенностью

Свойства подграфов программы

Свойства подграфов программы	Номер вершины выбора					
	7	8	11	12	15	16

Номера вершин перехода	8,11	9,10	12,15	13,14	16,27	17,18
Скорректированная сложность вершины выбора	23	23	23	23	23	23
Номера вершин подграфа	7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29	7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29	7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29	7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29	7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29	7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29
Номер нижней границы	30	30	30	30	30	30

Свойства подграфов программы	Номер вершины выбора				
	18	20	22	24	27
Номера вершин перехода	19,20	21,22	23,24	25,26	28,29
Скорректированная сложность вершины выбора	23	23	23	23	23
Номера вершин подграфа	7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29	7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29	7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29	7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29	7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29

Номер нижней границы	30	30	30	30	30

Скорректированные сложности вершин графа программы

Номер вершины графа программы	1	2	3	4	5	6	7	8	9
Скорректированная сложность вершины графа	1	1	1	1	1	1	23	23	23

Номер вершины графа программы	10	11	12	13	14	15	16	17	18
Скорректированная сложность вершины графа	23	23	23	23	23	23	23	23	23

Номер вершины графа программы	19	20	21	22	23	24	25	26	27
Скорректированная сложность вершины графа	23	23	23	23	23	23	23	23	23

Номер вершины графа программы	28	29	30

Скорректированная сложность вершины графа	23	1	0	S_a= 513

Таким образом, абсолютная граничная сложность S_a программы, схема алгоритма которой приведена на рис. 1, равна 50. Относительная граничная сложность данной программы равна $S_o = 1 – (30 – 1)/513 \approx 0{,}9435$.

```
Абсолютная сложность (CL): 5
Общее количество операторов: 48
Относительная сложность (cl): 0.1
Максимальный уровень вложенности (CLI): 8

{ } -> 11
function_call -> 7
= -> 4
; -> 1
while -> 1
< -> 1
when -> 2
if..else -> 2
% -> 2
== -> 3
*= -> 3
/= -> 3
<= -> 4
+= -> 1
-= -> 1
++ -> 2
```

Рисунок 2 – Работа программы

ВЫВОД

В ходе лабораторной работы был проанализирован исходный текст программы и изучены базовые и расширенные метрики сложности потока управления программ на примере языка программирования Kotlin.