Практическая работа 8

Функциональное программирование

1. Напишите функцию, которая принимает список на вход, и возвращает сумму всех элементов этого списка.

2. Напишите функцию, которая получает на вход список целых чисел и возвращает разность самого большого и самого маленького из них.

3. Создайте функцию для объединения двух списков целых чисел.

4. Создайте функцию, которая принимает три аргумента prob, prize, pay и возвращает True, если prob * prize > pay, в противном случае возвращает False.

5. Создайте функцию, которая принимает три аргумента prob, prize, pay и возвращает True, если prob * prize > pay, в противном случае возвращает False.

6. Функция получает на вход два числа. Она должна вернуть True, если сумма этих чисел меньше 100 и False в противном случае.

7. Напишите функцию, которая принимает целое число и возвращает True, если оно делится на 100. В противном случае функция должна вернуть False.

8. Напишите функцию, которая принимает количество минут и частоту кадров (FPS) и возвращает, сколько за это время кадров показывает компьютер при этом FPS.

9. Напишите функцию, которая возвращает True, если k^k == n для входных данных (n, k), и возвращает False в противном случае.

10. Создайте рекурсивную функцию, которая принимает два параметра и повторяет строку n количество раз. Первый параметр txt - это строка, которую нужно повторить, а второй параметр - количество повторений строки.

11. Создайте функцию, которая принимает уравнение (например, "1+1") и возвращает ответ.

12. Напишите функцию, которая принимает число number, и возвращает слово Google с количеством букв o, равным number.

13. Приветствие: Напишите функцию, которая выводит "Привет, мир!" на экран.

14. Сумма двух чисел: Создайте функцию, которая принимает два числа и возвращает их сумму.

15. Сравнение чисел: Напишите функцию, которая принимает два числа и возвращает большее из них.

16. Определение четности: Создайте функцию, которая принимает одно число и возвращает true, если оно четное, и false, если нечетное.

17. Факториал числа: Реализуйте функцию, которая вычисляет факториал введенного числа.

18. Проверка на простоту: Напишите функцию, которая проверяет, является ли заданное число простым.

19. Сумма чисел в массиве: Создайте функцию, которая принимает массив целых чисел и возвращает их сумму.

20. Наибольшее число в массиве: Напишите функцию, которая находит и возвращает максимальное значение в массиве.

21. Сортировка массива: Создайте функцию, которая сортирует массив чисел в порядке возрастания.

22. Проверка палиндрома: Напишите функцию, которая проверяет, является ли строка палиндромом.

23. Количество символов: Создайте функцию, которая принимает строку и возвращает количество символов в ней.

24. Конвертация в верхний регистр: Напишите функцию, которая принимает строку и возвращает её в верхнем регистре.

25. Объединение строк: Создайте функцию, которая принимает две строки и возвращает их объединение.

26. Возвращение последнего элемента массива: Напишите функцию, которая возвращает последний элемент переданного массива.

27. Проверка наличия элемента: Создайте функцию, которая проверяет, присутствует ли элемент в массиве.

28. Создание массива от 1 до N: Напишите функцию, которая создает массив целых чисел от 1 до N.

29. Максимум и минимум: Создайте функцию, которая находит одновременно максимальное и минимальное значения в массиве.

30. Сумма чисел от 1 до N: Напишите функцию, которая рассчитывает сумму всех чисел от 1 до N.

31. Преобразование Celsius в Fahrenheit: Создайте функцию, которая конвертирует температуру из Цельсия в Фаренгейт.

32. Обратный порядок строки: Напишите функцию, которая принимает строку и возвращает её в обратном порядке.

33. Поиск элемента по индексу: Создайте функцию, которая находит элемент по индексу в массиве.

34. Удаление пробелов из строки: Напишите функцию, которая удаляет все пробелы из переданной строки.

35. Сумма первых N натуральных чисел: Создайте функцию, которая возвращает сумму первых N натуральных чисел.

36. Проверка строки на наличие подстроки: Напишите функцию, которая проверяет, содержится ли одна строка в другой.

37. Печать таблицы умножения: Создайте функцию, которая выводит таблицу умножения от 1 до 10 для заданного числа.

38. Нахождение длины строки: Напишите функцию, которая возвращает длину переданной ей строки.

39. Переворот массива: Создайте функцию, которая переворачивает массив целых чисел.

40. Копирование массива: Напишите функцию, которая копирует массив и возвращает новый массив.

41. Количество гласных в строке: Создайте функцию, которая считает количество гласных в строке.

42. Индекс первого вхождения: Напишите функцию, которая возвращает индекс первого вхождения элемента в массиве, или -1, если элемент не найден.

```
import kotlin.random.Random
import java.util.Scanner
import kotlin.math.pow


val scanner = Scanner(System.`in`)

fun sumList(list: List<Int>): Int {
    return list.sum()
}

fun maxMinDifference(list: List<Int>): Int {
    return (list.maxOrNull() ?: 0) - (list.minOrNull() ?: 0)
}

fun combineLists(list1: List<Int>, list2: List<Int>): List<Int> {
    return list1 + list2
}

fun checkProfit(prob: Double, prize: Double, pay: Double): Boolean {
    return prob * prize > pay
}

fun sumLessThan100(a: Double, b: Double): Boolean {
    return a + b < 100
}


fun divisibleBy100(number: Int): Boolean {
    return number % 100 == 0
}


fun calculateFrames(minutes: Double, fps: Double): Double {
    return minutes * 60 * fps
}


fun checkPowerEquality(n: Int, k: Int): Boolean {
    return k.toDouble().pow(k) == n.toDouble()
}


fun repeatString(txt: String, n: Int): String {
    return if (n == 0) "" else txt + repeatString(txt, n - 1)
}


fun evaluateExpression(expr: String): Double {
    return expr.split("+").sumOf { it.trim().toDouble() }
}


fun createCustomGoogle(number: Int): String {
    val o = "o".repeat(number)
    return "G${o}gle"
}
```

```kotlin
fun helloWorld() {
    println("Привет, мир!")
}


fun sumTwoNumbers(a: Double, b: Double): Double {
    return a + b
}

fun findLargerNumber(a: Double, b: Double): Double {
    return maxOf(a, b)
}


fun isEven(number: Int): Boolean {
    return number % 2 == 0
}


fun factorial(n: Int): Long {
    return (1..n).fold(1L) { acc, i -> acc * i }
}


fun isPrime(n: Int): Boolean {
    return when {
        n < 2 -> false
        else -> (2 until n).none { n % it == 0 }
    }
}


fun sumArray(arr: List<Int>): Int {
    return arr.sum()
}


fun findMax(arr: List<Int>): Int? {
    return arr.maxOrNull()
}


fun sortArray(arr: MutableList<Int>): List<Int> {
    arr.sort()
    return arr
}


fun isPalindrome(s: String): Boolean {
    return s == s.reversed()
}


fun stringLength(s: String): Int {
    return s.length
}

fun toUpper(s: String): String {
    return s.uppercase()
}

fun concatenate(s1: String, s2: String): String {
    return s1 + s2
```

```kotlin
}


fun lastElement(arr: List<Int>): Int? {
    return arr.lastOrNull()
}


fun containsElement(arr: List<Int>, element: Int): Boolean {
    return element in arr
}


fun createSequence(n: Int): List<Int> {
    return (1..n).toList()
}


fun findMinMax(arr: List<Int>): Pair<Int?, Int?> {
    return Pair(arr.minOrNull(), arr.maxOrNull())
}


fun sumFrom1ToN(n: Int): Int {
    return n * (n + 1) / 2
}


fun celsiusToFahrenheit(c: Double): Double {
    return c * 9 / 5 + 32
}


fun reverseString(s: String): String {
    return s.reversed()
}


fun elementAtIndex(arr: List<Int>, index: Int): Int? {
    return arr.getOrNull(index)
}


fun removeSpaces(s: String): String {
    return s.replace(" ", "")
}


fun sumNaturalNumbers(n: Int): Int {
    return n * (n + 1) / 2
}


fun containsSubstring(s1: String, s2: String): Boolean {
    return s1.contains(s2)
}


fun multiplicationTable(n: Int) {
    (1..10).forEach { println("$n x $it = ${n * it}") }
}


fun getStringLength(s: String): Int {
```

```
    return s.length
}


fun reverseArray(arr: List<Int>): List<Int> {
    return arr.reversed()
}


fun copyArray(arr: List<Int>): List<Int> {
    return arr.toList()
}


fun countVowels(s: String): Int {
    val vowels = setOf('a', 'e', 'i', 'o', 'u', 'а', 'е', 'ё', 'и', 'о', 'у',
'ы', 'э', 'ю', 'я')
    return s.count { it.lowercaseChar() in vowels }
}


fun findFirstIndex(arr: List<Int>, target: Int): Int {
    return arr.indexOf(target)
}


fun findLastIndex(arr: List<Int>, target: Int): Int {
    return arr.lastIndexOf(target)
}

fun main() {
    val menu = """
        Выберите функцию (1-42, 0 для выхода):
    """.trimIndent()

    while (true) {
        print(menu)
        when (scanner.nextInt()) {
            1 -> {
                println("Введите размер списка:")
                val size = scanner.nextInt()
                val randomList = List(size) { Random.nextInt(1, 101) }
                println("Сгенерированный список: $randomList")
                println("Сумма всех элементов: ${sumList(randomList)}")
            }
            2 -> {
                println("Введите размер списка:")
                val numbers = MutableList(scanner.nextInt()) {
                    println("Введите элемент ${it + 1}:")
                    scanner.nextInt()
                }
                println("Разность между максимальным и минимальным:
${maxMinDifference(numbers)}")
            }
            3 -> {
                fun readList(name: String): List<Int> {
                    println("Введите размер списка $name:")
                    return MutableList(scanner.nextInt()) {
                        println("Введите элемент ${it + 1} для списка
$name:")
                        scanner.nextInt()
                    }
                }
                val list1 = readList("1")
```

```
            val list2 = readList("2")
            println("Объединенный список: ${combineLists(list1, list2)}")
        }
        4 -> {
            println("Введите prob, prize, pay через пробел:")
            val (prob, prize, pay) = List(3) { scanner.nextDouble() }
            println("Результат: ${checkProfit(prob, prize, pay)}")
        }
        5 -> {
            println("Введите два числа через пробел:")
            val (a, b) = List(2) { scanner.nextDouble() }
            println("Результат: ${sumLessThan100(a, b)}")
        }
        6 -> {
            println("Введите число:")
            println("Результат: ${divisibleBy100(scanner.nextInt())}")
        }
        7 -> {
            println("Введите минуты и FPS через пробел:")
            val (minutes, fps) = List(2) { scanner.nextDouble() }
            println("Количество кадров: ${calculateFrames(minutes,
fps)}")
        }
        8 -> {
            println("Введите n и k через пробел:")
            val (n, k) = List(2) { scanner.nextInt() }
            println("Результат: ${checkPowerEquality(n, k)}")
        }
        9 -> {
            scanner.nextLine()
            println("Введите строку:")
            val txt = scanner.nextLine()
            println("Введите количество повторений:")
            val n = scanner.nextInt()
            println("Результат: ${repeatString(txt, n)}")
        }
        10 -> {
            scanner.nextLine()
            println("Введите уравнение (только с +):")
            val expr = scanner.nextLine()
            try {
                println("Результат: ${evaluateExpression(expr)}")
            } catch (e: Exception) {
                println("Ошибка вычисления: ${e.message}")
            }
        }
        11 -> {
            println("Введите количество 'o':")
            val oCount = scanner.nextInt()
            println(createCustomGoogle(oCount))
        }
        12 -> helloWorld()
        13 -> {
            println("Введите два числа через пробел:")
            val (a, b) = List(2) { scanner.nextDouble() }
            println("Сумма: ${sumTwoNumbers(a, b)}")
        }
        14 -> {
            println("Введите два числа через пробел:")
            val (a, b) = List(2) { scanner.nextDouble() }
            println("Большее число: ${findLargerNumber(a, b)}")
        }
        15 -> {
            println("Введите число:")
```

```
                println("Результат: ${isEven(scanner.nextInt())}")
            }
            16 -> {
                println("Введите число:")
                val n = scanner.nextInt()
                println("Факториал: ${factorial(n)}")
            }
            17 -> {
                println("Введите число:")
                val n = scanner.nextInt()
                println("Результат: ${isPrime(n)}")
            }
            18 -> {
                println("Введите размер массива:")
                val size = scanner.nextInt()
                val arr = List(size) { Random.nextInt(1, 101) }
                println("Массив: $arr\nСумма: ${sumArray(arr)}")
            }
            19 -> {
                println("Введите размер массива:")
                val size = scanner.nextInt()
                val arr = List(size) { Random.nextInt(1, 101) }
                println("Массив: $arr\nМаксимум: ${findMax(arr)}")
            }
            20 -> {
                println("Введите размер массива:")
                val size = scanner.nextInt()
                val arr = MutableList(size) { Random.nextInt(1, 101) }
                println("Исходный массив: $arr")
                println("Отсортированный: ${sortArray(arr)}")
            }
            21 -> {
                scanner.nextLine()
                println("Введите строку:")
                val s = scanner.nextLine()
                println("Результат: ${isPalindrome(s)}")
            }
            22 -> {
                scanner.nextLine()
                println("Введите строку:")
                val s = scanner.nextLine()
                println("Длина: ${stringLength(s)}")
            }
            23 -> {
                scanner.nextLine()
                println("Введите строку:")
                val s = scanner.nextLine()
                println("Результат: ${toUpper(s)}")
            }
            24 -> {
                scanner.nextLine()
                println("Введите первую строку:")
                val s1 = scanner.nextLine()
                println("Введите вторую строку:")
                val s2 = scanner.nextLine()
                println("Результат: ${concatenate(s1, s2)}")
            }
            25 -> {
                println("Введите размер массива:")
                val size = scanner.nextInt()
                val arr = List(size) { Random.nextInt(1, 101) }
                println("Массив: $arr\nПоследний элемент:
${lastElement(arr)}")
            }
```

```
            26 -> {
                println("Введите размер массива:")
                val size = scanner.nextInt()
                val arr = List(size) { Random.nextInt(1, 101) }
                println("Массив: $arr")
                println("Введите искомый элемент:")
                val element = scanner.nextInt()
                println("Результат: ${containsElement(arr, element)}")
            }
            27 -> {
                println("Введите N:")
                val n = scanner.nextInt()
                println("Массив: ${createSequence(n)}")
            }
            28 -> {
                println("Введите размер массива:")
                val size = scanner.nextInt()
                val arr = List(size) { Random.nextInt(1, 101) }
                println("Массив: $arr")
                val (min, max) = findMinMax(arr)
                println("Min: $min, Max: $max")
            }
            29 -> {
                println("Введите N:")
                val n = scanner.nextInt()
                println("Сумма: ${sumFrom1ToN(n)}")
            }
            30 -> {
                println("Введите температуру в °C:")
                val c = scanner.nextDouble()
                println("Результат: ${celsiusToFahrenheit(c)} °F")
            }
            31 -> {
                scanner.nextLine()
                println("Введите строку:")
                val s = scanner.nextLine()
                println("Результат: ${reverseString(s)}")
            }
            32 -> {
                println("Введите размер массива:")
                val size = scanner.nextInt()
                val arr = List(size) { Random.nextInt(1, 101) }
                println("Массив: $arr")
                println("Введите индекс:")
                val index = scanner.nextInt()
                println("Элемент: ${elementAtIndex(arr, index) ?: "Не
найден"}")
            }
            33 -> {
                scanner.nextLine()
                println("Введите строку:")
                val s = scanner.nextLine()
                println("Результат: ${removeSpaces(s)}")
            }
            34 -> {
                println("Введите N:")
                val n = scanner.nextInt()
                println("Сумма: ${sumNaturalNumbers(n)}")
            }
            35 -> {
                scanner.nextLine()
                println("Введите основную строку:")
                val s1 = scanner.nextLine()
                println("Введите подстроку:")
```

```
                val s2 = scanner.nextLine()
                println("Результат: ${containsSubstring(s1, s2)}")
            }
            36 -> {
                println("Введите число:")
                val n = scanner.nextInt()
                multiplicationTable(n)
            }
            37 -> {
                scanner.nextLine()
                println("Введите строку:")
                val s = scanner.nextLine()
                println("Длина: ${getStringLength(s)}")
            }
            38 -> {
                println("Введите размер массива:")
                val size = scanner.nextInt()
                val arr = List(size) { Random.nextInt(1, 101) }
                println("Исходный массив: $arr")
                println("Перевернутый: ${reverseArray(arr)}")
            }
            39 -> {
                println("Введите размер массива:")
                val size = scanner.nextInt()
                val arr = List(size) { Random.nextInt(1, 101) }
                println("Оригинал: $arr\nКопия: ${copyArray(arr)}")
            }
            40 -> {
                scanner.nextLine()
                println("Введите строку:")
                val s = scanner.nextLine()
                println("Количество гласных: ${countVowels(s)}")
            }
            41 -> {
                println("Введите размер массива:")
                val size = scanner.nextInt()
                val arr = List(size) { Random.nextInt(1, 101) }
                println("Массив: $arr")
                println("Введите искомый элемент:")
                val target = scanner.nextInt()
                println("Индекс: ${findFirstIndex(arr, target)}")
            }
            42 -> {
                println("Введите размер массива:")
                val size = scanner.nextInt()
                val arr = List(size) { Random.nextInt(1, 101) }
                println("Массив: $arr")
                println("Введите искомый элемент:")
                val target = scanner.nextInt()
                println("Последний индекс: ${findLastIndex(arr, target)}")
            }
            0 -> return
            else -> println("Неверный ввод!")
        }
        scanner.nextLine()
        println("\nНажмите Enter для продолжения...")
        scanner.nextLine()
    }
}
```