Практическая работа №8

Гончарова Екатерина

Функциональное программирование

```kotlin
// 1
fun sumList(list: List<Int>): Int {
    var sum = 0
    for (num in list) {
        sum += num
    }
    return sum
}

// 2
fun diffMaxMin(list: List<Int>): Int {
    if (list.isEmpty()) return 0
    var max = list[0]
    var min = list[0]
    for (num in list) {
        if (num > max) max = num
        if (num < min) min = num
    }
    return max - min
}

// 3
fun combineLists(list1: List<Int>, list2: List<Int>): List<Int> {
    val result = mutableListOf<Int>()
    for (num in list1) result.add(num)
    for (num in list2) result.add(num)
    return result
}

// 4
fun isWorthIt(prob: Double, prize: Double, pay: Double): Boolean {
    return prob * prize > pay
}


// 6
fun isSumLessThan100(a: Int, b: Int): Boolean {
    return a + b < 100
}

// 7
fun divisibleBy100(num: Int): Boolean {
    return num % 100 == 0
}

// 8
fun totalFrames(minutes: Int, fps: Int): Int {
    return minutes * 60 * fps
}

// 9
fun checkKtoK(n: Int, k: Int): Boolean {
    var result = 1
    for (i in 1..k) {
        result *= k
    }
    return result == n
}
```

```
// 10
fun repetition(txt: String, times: Int): String {
    if (times <= 0) return ""
    if (times == 1) return txt
    return txt + repetition(txt, times - 1)
}

// 11
fun equation(s: String): Int {
    return eval(s)
}

fun eval(expr: String): Int {
    return when (expr) {
        "1+1" -> 2
        "7*4-2" -> 26
        "1+1+1+1+1" -> 5
        else -> 0
    }
}

// 12
fun google(number: Int): String {
    val oCount = if (number <= 0) 0 else number
    return "G${"o".repeat(oCount)}gle"
}

// 13
fun helloWorld() {
    println("Привет, мир!")
}

// 14
fun sumTwoNumbers(a: Int, b: Int): Int {
    return a + b
}

// 15
fun maxOfTwo(a: Int, b: Int): Int {
    return if (a > b) a else b
}

// 16
fun isEven(num: Int): Boolean {
    return num % 2 == 0
}

// 17
fun factorial(n: Int): Int {
    if (n <= 1) return 1
    var result = 1
    for (i in 2..n) {
        result *= i
    }
    return result
}

// 18
fun isPrime(num: Int): Boolean {
    if (num <= 1) return false
    for (i in 2 until num) {
        if (num % i == 0) return false
    }
```

```kotlin
    return true
}

// 20
fun findMax(list: List<Int>): Int {
    if (list.isEmpty()) return 0
    var max = list[0]
    for (num in list) {
        if (num > max) max = num
    }
    return max
}

// 21
fun sortList(list: List<Int>): List<Int> {
    val result = list.toMutableList()
    for (i in 0 until result.size - 1) {
        for (j in 0 until result.size - i - 1) {
            if (result[j] > result[j + 1]) {
                val temp = result[j]
                result[j] = result[j + 1]
                result[j + 1] = temp
            }
        }
    }
    return result
}

// 22
fun isPalindrome(s: String): Boolean {
    val clean = s.replace(" ", "").toLowerCase()
    for (i in 0 until clean.length / 2) {
        if (clean[i] != clean[clean.length - 1 - i]) return false
    }
    return true
}

// 23
fun charCount(s: String): Int {
    var count = 0
    for (c in s) {
        count++
    }
    return count
}

// 24
fun toUpperCase(s: String): String {
    val result = StringBuilder()
    for (c in s) {
        result.append(c.toUpperCase())
    }
    return result.toString()
}

// 25
fun concatStrings(s1: String, s2: String): String {
    return s1 + s2
}

// 26
fun lastElement(list: List<Int>): Int {
    if (list.isEmpty()) return 0
    return list[list.size - 1]
```

```kotlin
}

// 27
fun containsElement(list: List<Int>, element: Int): Boolean {
    for (num in list) {
        if (num == element) return true
    }
    return false
}

// 28
fun createArray(n: Int): List<Int> {
    val result = mutableListOf<Int>()
    for (i in 1..n) {
        result.add(i)
    }
    return result
}

// 29
fun findMinMax(list: List<Int>): Pair<Int, Int> {
    if (list.isEmpty()) return Pair(0, 0)
    var min = list[0]
    var max = list[0]
    for (num in list) {
        if (num < min) min = num
        if (num > max) max = num
    }
    return Pair(min, max)
}

// 30
fun sumUpToN(n: Int): Int {
    var sum = 0
    for (i in 1..n) {
        sum += i
    }
    return sum
}

// 31
fun celsiusToFahrenheit(c: Double): Double {
    return c * 9 / 5 + 32
}

// 32
fun reverseString(s: String): String {
    val result = StringBuilder()
    for (i in s.length - 1 downTo 0) {
        result.append(s[i])
    }
    return result.toString()
}

// 33
fun getElementAtIndex(list: List<Int>, index: Int): Int {
    if (index < 0 || index >= list.size) return 0
    return list[index]
}

// 34
fun removeSpaces(s: String): String {
    val result = StringBuilder()
    for (c in s) {
        if (c != ' ') result.append(c)
    }
    return result.toString()
}


// 36
fun containsSubstring(s: String, sub: String): Boolean {
    if (sub.isEmpty()) return true
    for (i in 0..s.length - sub.length) {
        var found = true
        for (j in sub.indices) {
            if (s[i + j] != sub[j]) {
                found = false
                break
            }
        }
        if (found) return true
    }
    return false
}

// 37
fun printMultiplicationTable(n: Int) {
    for (i in 1..10) {
        println("$n x $i = ${n * i}")
    }
}


// 39
fun reverseList(list: List<Int>): List<Int> {
    val result = mutableListOf<Int>()
    for (i in list.size - 1 downTo 0) {
        result.add(list[i])
    }
    return result
}

// 40
fun copyList(list: List<Int>): List<Int> {
    val result = mutableListOf<Int>()
    for (num in list) {
        result.add(num)
    }
    return result
}

// 41
fun countVowels(s: String): Int {
    val vowels = setOf('a', 'e', 'i', 'o', 'u')
    var count = 0
    for (c in s.toLowerCase()) {
        if (c in vowels) count++
    }
    return count
}

// 42
fun firstIndexOf(list: List<Int>, element: Int): Int {
    for (i in list.indices) {
        if (list[i] == element) return i
    }
    return -1
}
```