1.

```kotlin
fun sumOfNumbers(numbers: List<Int>): Int {
    return numbers.sum() // Сумма всех элементов списка
}

fun main() {
    val myList = listOf(1, 2, 3, 4, 5) // Пример списка
    val result = sumOfNumbers(myList) // Вызов функции
    println("Сумма всех элементов списка: $result") // Вывод результата
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-
Сумма всех элементов списка: 15

Process finished with exit code 0
```

2.

```kotlin
fun differenceBetweenMaxMin(numbers: List<Int>): Int {
    if (numbers.isEmpty()) {
        throw IllegalArgumentException("Список не должен быть пустым") // Проверка на пустой список
    }
    val max = numbers.maxOrNull() ?: 0 // Находим максимальный элемент
    val min = numbers.minOrNull() ?: 0 // Находим минимальный элемент
    return max - min // Возвращаем разность
}

fun main() {
    val myList = listOf(10, 2, 5, 8, 3) // Пример списка
    val result = differenceBetweenMaxMin(myList) // Вызов функции
    println("Разность между самым большим и самым маленьким элементами: $result") // Вывод результата
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community
Разность между самым большим и самым маленьким элементами: 8

Process finished with exit code 0
```

3.

```kotlin
fun mergeLists(list1: List<Int>, list2: List<Int>): List<Int> {
    return list1 + list2 // Объединяем два списка
}

fun main() {
    val firstList = listOf(1, 2, 3) // Первый список
    val secondList = listOf(4, 5, 6) // Второй список
    val mergedList = mergeLists(firstList, secondList) // Вызов функции
    println("Объединенный список: $mergedList") // Вывод результата
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C
Разность между самым большим и самым маленьким элементами: 8

Process finished with exit code 0
```

4-5.

```kotlin
fun isProfitable(prob: Double, prize: Double, pay: Double): Boolean {
    return prob * prize > pay // Возвращаем результат сравнения
}

fun main() {
    // Примеры использования
    val prob1 = 0.8
    val prize1 = 100.0
    val pay1 = 70.0
    println("Прибыльно? ${isProfitable(prob1, prize1, pay1)}") // Ожидается True

    val prob2 = 0.5
    val prize2 = 50.0
    val pay2 = 30.0
    println("Прибыльно? ${isProfitable(prob2, prize2, pay2)}") // Ожидается False
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent
Прибыльно? true
Прибыльно? false

Process finished with exit code 0
```

6

```
fun isSumLessThan100(num1: Int, num2: Int): Boolean {
    return (num1 + num2) < 100
}

fun main() {
    val number1 = 45 // Пример первого числа
    val number2 = 50 // Пример второго числа

    val result = isSumLessThan100(number1, number2) // Вызов функции
    println("Сумма меньше 100?: $result") // Вывод результата
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin
Сумма меньше 100?: true

Process finished with exit code 0
```

7

```
fun isDivisibleBy100(number: Int): Boolean {
    return number % 100 == 0
}

fun main() {
    val testNumber = 250 // Пример числа для проверки

    val result = isDivisibleBy100(testNumber) // Вызов функции
    println("Число $testNumber делится на 100?: $result") // Вывод результата
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Program
Число 250 делится на 100?: false

Process finished with exit code 0
```

8

```kotlin
fun calculateFrames(minutes: Int, fps: Int): Int {
    // Переводим минуты в секунды и умножаем на FPS
    return minutes * 60 * fps
}

fun main() {
    val minutes = 5 // Пример количества минут
    val fps = 30    // Пример частоты кадров

    val totalFrames = calculateFrames(minutes, fps) // Вызов функции
    println("За $minutes минут при $fps FPS компьютер покажет $totalFrames кадров.")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\P
За 5 минут при 30 FPS компьютер покажет 9000 кадров.

Process finished with exit code 0
```

9

```kotlin
fun isKToPowerKEqualN(n: Int, k: Int): Boolean {
    // Проверяем, является ли k положительным
    if (k <= 0) {
        return false
    }
    // Вычисляем k^k
    val result = Math.pow(k.toDouble(), k.toDouble()).toInt()
    // Сравниваем с n
    return result == n
}

fun main() {
    val n = 27 // Пример n
    val k = 3  // Пример k

    val result = isKToPowerKEqualN(n, k) // Вызов функции
    println("k^k == n для n = $n и k = $k: $result")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:
За 5 минут при 30 FPS компьютер покажет 9000 кадров.

Process finished with exit code 0
```

```kotlin
fun repeatString(txt: String, n: Int): String {
    // Базовый случай: если n равно 0, возвращаем пустую строку
    if (n <= 0) {
        return ""
    }
    // Рекурсивный вызов: добавляем строку к результату
    return txt + repeatString(txt, n: n - 1)
}

fun main() {
    val txt = "Hello " // Строка для повторения
    val n = 3          // Количество повторений

    val result = repeatString(txt, n) // Вызов функции
    println(result) // Вывод результата
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaage
Hello Hello Hello

Process finished with exit code 0
```

```kotlin
import java.util.Stack

fun evaluateExpression(expression: String): Double {
    val tokens = expression.replace( oldValue: " ",  newValue: "").toCharArray()
    val values = Stack<Double>()
    val ops = Stack<Char>()

    var i = 0
    while (i < tokens.size) {
        when {
            tokens[i].isDigit() -> {
                val sb = StringBuilder()
                while (i < tokens.size && (tokens[i].isDigit() || tokens[i] == '.')) {
                    sb.append(tokens[i++])
                }
                values.push(sb.toString().toDouble())
                i--
            }
            tokens[i] == '(' -> ops.push(tokens[i])
            tokens[i] == ')' -> {
                while (ops.peek() != '(') {
                    values.push(applyOp(ops.pop(), values.pop(), values.pop()))
                }
                ops.pop()
            }
            tokens[i] in "+-*/" -> {
                while (!ops.empty() && hasPrecedence(tokens[i], ops.peek())) {
                    values.push(applyOp(ops.pop(), values.pop(), values.pop()))
                }
                ops.push(tokens[i])
            }
        }
        i++
    }

    while (!ops.empty()) {
        values.push(applyOp(ops.pop(), values.pop(), values.pop()))
    }

    return values.pop()
}
```

```kotlin
private fun hasPrecedence(op1: Char, op2: Char): Boolean {
    if (op2 == '(' || op2 == ')') return false
    return (op1 != '*' && op1 != '/') || (op2 != '+' && op2 != '-')
}

private fun applyOp(op: Char, b: Double, a: Double): Double {
    return when (op) {
        '+' -> a + b
        '-' -> a - b
        '*' -> a * b
        '/' -> a / b
        else -> throw UnsupportedOperationException("Неподдерживаемая операция: $op")
    }
}

fun main() {
    val testCases = listOf(
        "1 + 1" to 2.0,
        "2 * 3" to 6.0,
        "5 / 2" to 2.5,
        "10 - 4" to 6.0,
        "2 * (3 + 4)" to 14.0
    )

    testCases.forEach { (expr, expected) ->
        try {
            val result = evaluateExpression(expr)
            println("'$expr' = $result (ожидается: $expected) ${if (result == expected) "✓" else "×"}")
        } catch (e: Exception) {
            println("Ошибка вычисления '$expr': ${e.message}")
        }
    }
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\j
'1 + 1' = 2.0 (ожидается: 2.0) ✓
'2 * 3' = 6.0 (ожидается: 6.0) ✓
'5 / 2' = 2.5 (ожидается: 2.5) ✓
'10 - 4' = 6.0 (ожидается: 6.0) ✓
'2 * (3 + 4)' = 14.0 (ожидается: 14.0) ✓

Process finished with exit code 0
```

```kotlin
fun generateGoogleWithO(number: Int): String {
    // Генерируем строку с нужным количеством "o"
    val oCount = if (number < 0) 0 else number // Не допускаем отрицательных значений
    return "G" + "o".repeat(oCount) + "ogle"
}

fun main() {
    val number = 3 // Пример числа
    val result = generateGoogleWithO(number) // Вызов функции
    println(result) // Вывод результата
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaager
Goooogle

Process finished with exit code 0
```

13

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

fun main() {
    greet() // Вызов функции для вывода приветствия
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java
Привет, мир!

Process finished with exit code 0
```

14

```kotlin
fun sum(a: Int, b: Int): Int {
    return a + b
}

fun main() {
    val number1 = 5
    val number2 = 10
    val result = sum(number1, number2) // Вызов функции для получения суммы
    println("Сумма: $result") // Вывод результата
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaage
Сумма: 15

Process finished with exit code 0
```

15

```kotlin
fun max(a: Int, b: Int): Int {
    return if (a > b) a else b
}

fun main() {
    val number1 = 7
    val number2 = 12
    val result = max(number1, number2) // Вызов функции для нахождения большего числа
    println("Большее число: $result") // Вывод результата
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaage
Большее число: 12

Process finished with exit code 0
```

16

```kotlin
fun isEven(number: Int): Boolean {
    return number % 2 == 0 // Возвращает true, если число четное
}

fun main() {
    val number = 8
    val result = isEven(number) // Вызов функции для проверки четности
    println("Число $number четное? $result") // Вывод результата
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:
Число 8 четное? true

Process finished with exit code 0
```

17

```kotlin
fun factorial(n: Int): Int {
    return if (n == 0) 1 else n * factorial( n: n - 1) // Рекурсивное вычисление факториала
}

fun main() {
    val number = 5
    val result = factorial(number)
    println("Факториал числа $number равен $result")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-java
Факториал числа 5 равен 120

Process finished with exit code 0
```

18

```kotlin
fun isPrime(number: Int): Boolean {
    if (number <= 1) return false
    for (i in 2 ≤ until < number) {
        if (number % i == 0) return false // Проверка делимости
    }
    return true
}

fun main() {
    val number = 7
    val result = isPrime(number)
    println("Число $number простое? $result")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-j
Число 7 простое? true

Process finished with exit code 0
```

19

```kotlin
fun sumOfArray(arr: IntArray): Int {
    var sum = 0
    for (num in arr) {
        sum += num // Суммирование элементов массива
    }
    return sum
}

fun main() {
    val array = intArrayOf(1, 2, 3, 4, 5)
    val result = sumOfArray(array)
    println("Сумма элементов массива равна $result")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-java
Сумма элементов массива равна 15

Process finished with exit code 0
```

20

```kotlin
fun findMax(arr: IntArray): Int {
    var max = arr[0] // Предполагаем, что первый элемент - максимальный
    for (num in arr) {
        if (num > max) {
            max = num // Обновляем максимум
        }
    }
    return max
}

fun main() {
    val array = intArrayOf(1, 5, 3, 9, 2)
    val result = findMax(array)
    println("Наибольшее число в массиве: $result")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:
Наибольшее число в массиве: 9

Process finished with exit code 0
```

21

```kotlin
fun sortArray(arr: IntArray): IntArray {
    return arr.sortedArray() // Используем встроенный метод сортировки
}

fun main() {
    val array = intArrayOf(5, 3, 8, 1, 2)
    val sortedArray = sortArray(array)
    println("Отсортированный массив: ${sortedArray.joinToString()}")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Program File
Отсортированный массив: 1, 2, 3, 5, 8

Process finished with exit code 0
```

22

```kotlin
fun isPalindrome(str: String): Boolean {
    val cleanStr = str.replace(Regex( pattern: "[^A-Za-z0-9]"),  replacement: "").lowercase() // Убираем лишние символы и меняем на нижний регистр
    return cleanStr == cleanStr.reversed() // Сравниваем с обратной строкой
}

fun main() {
    val inputStr = "A man, a plan, a canal, Panama"
    val result = isPalindrome(inputStr)
    println("Строка \"$inputStr\" является палиндромом? $result")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Program Files\Je
Строка "A man, a plan, a canal, Panama" является палиндромом? true

Process finished with exit code 0
```

23

```kotlin
fun countCharacters(str: String): Int {
    return str.length // Просто возвращаем длину строки
}

fun main() {
    val inputStr = "Hello, Kotlin!"
    val count = countCharacters(inputStr)
    println("Количество символов в строке: $count")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.ex
Количество символов в строке: 14

Process finished with exit code 0
```

24

```kotlin
fun toUpperCase(str: String): String {
    return str.uppercase() // Преобразуем строку в верхний регистр
}

fun main() {
    val inputStr = "hello, kotlin!"
    val result = toUpperCase(inputStr)
    println("Верхний регистр: $result")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Ed
Верхний регистр: HELLO, KOTLIN!

Process finished with exit code 0
```

25

```kotlin
fun concatenateStrings(str1: String, str2: String): String {
    return str1 + str2 // Объединяем две строки
}

fun main() {
    val str1 = "Hello, "
    val str2 = "world!"
    val result = concatenateStrings(str1, str2)
    println("Объединенные строки: $result")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C
Объединенные строки: Hello, world!

Process finished with exit code 0
```

26

```
fun getLastElement(arr: IntArray): Int? {
    return if (arr.isNotEmpty()) arr.last() else null // Проверяем массив на пустоту
}

fun main() {
    val array = intArrayOf(1, 2, 3, 4, 5)
    val lastElement = getLastElement(array)
    println("Последний элемент массива: $lastElement")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-j
Последний элемент массива: 5

Process finished with exit code 0
```

27

```
fun containsElement(arr: IntArray, element: Int): Boolean {
    return arr.contains(element) // Проверяем наличие элемента в массиве
}

fun main() {
    val array = intArrayOf(1, 2, 3, 4, 5)
    val elementToCheck = 3
    val exists = containsElement(array, elementToCheck)
    println("Элемент $elementToCheck присутствует в массиве? $exists")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\Int
Элемент 3 присутствует в массиве? true

Process finished with exit code 0
```

28

```kotlin
fun createArray(n: Int): IntArray {
    return IntArray(n) { it + 1 } // Создаем массив от 1 до N
}

fun main() {
    val n = 5
    val array = createArray(n)
    println("Массив от 1 до $n: ${array.joinToString()}")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.e
Массив от 1 до 5: 1, 2, 3, 4, 5

Process finished with exit code 0
```

29

```kotlin
fun findMinMax(arr: IntArray): Pair<Int?, Int?> {
    return if (arr.isNotEmpty()) arr.minOrNull() to arr.maxOrNull() else null to null // Находим минимум и максимум
}

fun main() {
    val array = intArrayOf(3, 1, 4, 1, 5, 9)
    val (min, max) = findMinMax(array)
    println("Минимум: $min, Максимум: $max")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent
Минимум: 1, Максимум: 9

Process finished with exit code 0
```

30

```kotlin
fun sumFromOneToN(n: Int): Int {
    return (n * (n + 1)) / 2 // Формула суммы первых N чисел
}

fun main() {
    val n = 5
    val sum = sumFromOneToN(n)
    println("Сумма от 1 до $n: $sum")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:
Сумма от 1 до 5: 15

Process finished with exit code 0
```

31

```kotlin
fun celsiusToFahrenheit(celsius: Double): Double {
    return (celsius * 9 / 5) + 32 // Конвертация температуры
}

fun main() {
    val celsius = 25.0
    val fahrenheit = celsiusToFahrenheit(celsius)
    println("$celsius °C = $fahrenheit °F")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Program Files
25.0 °C = 77.0 °F

Process finished with exit code 0
```

32

```kotlin
fun reverseString(str: String): String {
    return str.reversed() // Обратный порядок строки
}

fun main() {
    val inputStr = "Hello, Kotlin!"
    val reversed = reverseString(inputStr)
    println("Обратный порядок: $reversed")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent
Обратный порядок: !niltoK ,olleH

Process finished with exit code 0
```

33

```kotlin
fun getElementAtIndex(arr: IntArray, index: Int): Int? {
    return if (index in arr.indices) arr[index] else null // Получаем элемент по индексу
}

fun main() {
    val array = intArrayOf(10, 20, 30, 40)
    val index = 2
    val element = getElementAtIndex(array, index)
    println("Элемент по индексу $index: $element")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Program Files\
Элемент по индексу 2: 30

Process finished with exit code 0
```

34

```kotlin
fun removeSpaces(str: String): String {
    return str.replace( oldValue: " ",  newValue: "") // Удаляем пробелы
}

fun main() {
    val inputStr = "Hello World!"
    val noSpaces = removeSpaces(inputStr)
    println("Строка без пробелов: $noSpaces")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-ja
Строка без пробелов: HelloWorld!

Process finished with exit code 0
```

35

```kotlin
fun sumOfFirstNNaturalNumbers(n: Int): Int {
    return (n * (n + 1)) / 2 // Формула для суммы первых N натуральных чисел
}

fun main() {
    val n = 5
    val sum = sumOfFirstNNaturalNumbers(n)
    println("Сумма первых $n натуральных чисел: $sum")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Progr
Сумма первых 5 натуральных чисел: 15

Process finished with exit code 0
```

36

```kotlin
fun containsSubstring(mainString: String, subString: String): Boolean {
    return mainString.contains(subString) // Проверяем наличие подстроки
}

fun main() {
    val mainStr = "Hello, Kotlin!"
    val subStr = "Kotlin"
    val result = containsSubstring(mainStr, subStr)
    println("Содержится ли '$subStr' в '$mainStr': $result")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent
Содержится ли 'Kotlin' в 'Hello, Kotlin!': true

Process finished with exit code 0
```

37

```kotlin
fun printMultiplicationTable(number: Int) {
    for (i in 1 ≤ .. ≤ 10) {
        println("$number * $i = ${number * i}") // Выводим таблицу умножения
    }
}

fun main() {
    val number = 7
    println("Таблица умножения для $number:")
    printMultiplicationTable(number)
}
```

```
Таблица умножения для 7:
7 * 1 = 7
7 * 2 = 14
7 * 3 = 21
7 * 4 = 28
7 * 5 = 35
7 * 6 = 42
7 * 7 = 49
7 * 8 = 56
7 * 9 = 63
7 * 10 = 70

Process finished with exit code 0
```

38

```kotlin
fun stringLength(str: String): Int {
    return str.length // Возвращаем длину строки
}

fun main() {
    val inputStr = "Hello, Kotlin!"
    val length = stringLength(inputStr)
    println("Длина строки '$inputStr': $length")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C
Длина строки 'Hello, Kotlin!': 14

Process finished with exit code 0
```

39

```
fun reverseArray(arr: IntArray): IntArray {
    return arr.reversedArray() // Переворачиваем массив
}

fun main() {
    val array = intArrayOf(1, 2, 3, 4, 5)
    val reversed = reverseArray(array)
    println("Перевернутый массив: ${reversed.joinToString()}")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-java
Перевернутый массив: 5, 4, 3, 2, 1

Process finished with exit code 0
```

40

```
fun copyArray(arr: IntArray): IntArray {
    return arr.copyOf() // Копируем массив
}

fun main() {
    val originalArray = intArrayOf(1, 2, 3, 4, 5)
    val copiedArray = copyArray(originalArray)
    println("Копированный массив: ${copiedArray.joinToString()}")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Pro
Копированный массив: 1, 2, 3, 4, 5

Process finished with exit code 0
```

41

```kotlin
fun countVowels(str: String): Int {
    val vowels = "aeiouAEIOU" // Гласные буквы
    return str.count { it in vowels } // Считаем гласные
}

fun main() {
    val inputStr = "Hello, Kotlin!"
    val vowelCount = countVowels(inputStr)
    println("Количество гласных в строке '$inputStr': $vowelCount")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\
Количество гласных в строке 'Hello, Kotlin!': 4

Process finished with exit code 0
```

42

```kotlin
fun indexOfFirstOccurrence(arr: IntArray, element: Int): Int {
    return arr.indexOf(element) // Находим индекс первого вхождения
}

fun main() {
    val array = intArrayOf(1, 2, 3, 4, 2, 5)
    val elementToFind = 2
    val index = indexOfFirstOccurrence(array, elementToFind)
    println("Индекс первого вхождения элемента $elementToFind: $index")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Pro
Индекс первого вхождения элемента 2: 1

Process finished with exit code 0
```