```kotlin
//1
fun sumList(list: List<Int>): Int = list.sum()  new *
//2
fun diffMaxMin(list: List<Int>): Int = list.maxOrNull()!! - list.minOrNull()!!  new *
//3
fun combineLists(list1: List<Int>, list2: List<Int>): List<Int> = list1 + list2  new *
//4-5
fun isProfitable(prob: Double, prize: Double, pay: Double): Boolean = prob * prize > pay  new *
//6
fun isSumLessThan100(a: Int, b: Int): Boolean = a + b < 100  new *
//7
fun isDivisibleBy100(num: Int): Boolean = num % 100 == 0  new *
//8
fun calculateFrames(minutes: Int, fps: Int): Int = minutes * 60 * fps  new *
//9
fun repetition(txt: String, n: Int): String = if (n == 0) "" else txt + repetition(txt,  n: n - 1)  new *
//10
fun repetition(txt: String, n: Int): String = if (n == 0) "" else txt + repetition(txt,  n: n - 1)  new *
//11
fun equation(expr: String): Int {  new *
    val engine = javax.script.ScriptEngineManager().getEngineByName( shortName: "js")
    return engine.eval(expr) as Int
}
//12
fun google(number: Int): String = "G${"o".repeat(number)}gle"  new *
//13
fun greet() = println("Привет, мир!")  new *
//14
fun sum(a: Int, b: Int): Int = a + b  new *
//15
fun maxOfTwo(a: Int, b: Int): Int = maxOf(a, b)  new *
//16
fun isEven(num: Int): Boolean = num % 2 == 0  new *
//17
fun factorial(n: Int): Int = if (n <= 1) 1 else n * factorial( n: n - 1)  new *
```

```kotlin
//18
fun isPrime(num: Int): Boolean {  new *
    if (num <= 1) return false
    for (i in 2 ≤ until < num) {
        if (num % i == 0) return false
    }
    return true
}
//19
fun sumArray(arr: IntArray): Int = arr.sum()  new *
//20
fun maxInArray(arr: IntArray): Int = arr.maxOrNull()!!  new *
//21
fun sortArray(arr: IntArray): IntArray = arr.sortedArray()  new *
//22
fun isPalindrome(s: String): Boolean = s == s.reversed()  new *
//23
fun charCount(s: String): Int = s.length  new *
//24
fun toUpperCase(s: String): String = s.uppercase()  new *
//25
fun concatStrings(s1: String, s2: String): String = s1 + s2  new *
//26
fun lastElement(arr: IntArray): Int = arr.last()  new *
//27
fun containsElement(arr: IntArray, element: Int): Boolean = element in arr  new *
//28
fun createArray(n: Int): IntArray = IntArray(n) { it + 1 }  new *
//29
fun findMinMax(arr: IntArray): Pair<Int, Int> = Pair(arr.minOrNull()!!, arr.maxOrNull()!!)  new *
//30
fun sumUpToN(n: Int): Int = n * (n + 1) / 2  new *
//31
fun celsiusToFahrenheit(c: Double): Double = c * 9/5 + 32  new *
//32
fun reverseString(s: String): String = s.reversed()  new *
//33
fun getElementAtIndex(arr: IntArray, index: Int): Int = arr[index]  new *
```

```kotlin
//34
fun removeSpaces(s: String): String = s.replace( oldValue: " ",  newValue: "")  new *
//35
fun sumNaturalNumbers(n: Int): Int = n * (n + 1) / 2  new *
//36
fun containsSubstring(s: String, sub: String): Boolean = sub in s  new *
//37
fun printMultiplicationTable(n: Int) {  new *
    for (i in 1 ≤ .. ≤ 10) {
        println("$n x $i = ${n * i}")
    }
}
//38
fun stringLength(s: String): Int = s.length  new *
//39
fun reverseArray(arr: IntArray): IntArray = arr.reversedArray()  new *
//40
fun copyArray(arr: IntArray): IntArray = arr.copyOf()  new *
//41
fun countVowels(s: String): Int = s.count { it.toLowerCase() in setOf('a', 'e', 'i', 'o', 'u') }  new *
//42
fun firstIndexOf(arr: IntArray, element: Int): Int = arr.indexOf(element)  new *
```