Лабораторная №6

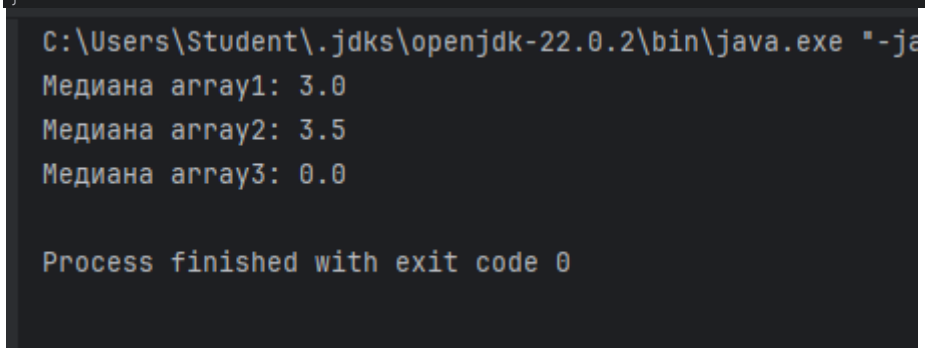
Кучеренко

1)

```
1.Задание
fun main() {

    val numbers = arrayOf(1, 2, 3, 4, 5)

    for (number in numbers) {
        println(number)
```

2)

```
fun main() {
    val numbers = arrayOf(1, 2, 3, 4, 5)

    val sum = numbers.sum()

    println("Сумма элементов массива: $sum")
}
```



3)

```
fun main() {

    val numbers = arrayOf(5, 3, 8, 1, 2, 7, 4, 10, 6, 9)

    val maxNumber = numbers.maxOrNull()
    val minNumber = numbers.minOrNull()

    println("Максимальное значение: $maxNumber")
    println("Минимальное значение: $minNumber")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Commu
Медиана array1: 3.0
Медиана array2: 3.5
Медиана array3: 0.0

Process finished with exit code 0
```

4)

```
fun bubbleSort(arr: IntArray): IntArray {
    val n = arr.size
    var swapped: Boolean

    do {
        swapped = false
        for (i in 1 until n) {
            if (arr[i - 1] > arr[i]) {

                val temp = arr[i - 1]
                arr[i - 1] = arr[i]
                arr[i] = temp
                swapped = true
            }
        }
    } while (swapped)

    return arr
}

fun main() {
    val numbers = intArrayOf(64, 34, 25, 12, 22, 11, 90)
    val sortedNumbers = bubbleSort(numbers)

    println("Отсортированный массив: ${sortedNumbers.joinToString(", ")}")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe
Медиана array1: 3.0
Медиана array2: 3.5
Медиана array3: 0.0

Process finished with exit code 0
```

5)

```
fun main() {
    val array = arrayOf(1, 2, 2, 3, 4, 4, 5, 5, 6)

    val uniqueElements = array.toSet()

    println("Уникальные элементы: $uniqueElements")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.
Медиана array1: 3.0
Медиана array2: 3.5
Медиана array3: 0.0

Process finished with exit code 0
```

6)

```
fun main() {

    val numbers = arrayOf(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

    val evenNumbers = numbers.filter { it % 2 == 0 }.toTypedArray()

    val oddNumbers = numbers.filter { it % 2 != 0 }.toTypedArray()

    println("Четные числа: ${evenNumbers.joinToString(", ")}")
    println("Нечетные числа: ${oddNumbers.joinToString(", ")}")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Pro
Медиана array1: 3.0
Медиана array2: 3.5
Медиана array3: 0.0

Process finished with exit code 0
```

7)

```
fun main() {

    val array = intArrayOf(1, 2, 3, 4, 5)

    println("Исходный массив: ${array.joinToString(", ")}")

    val reversedArray = array.reversedArray()

    println("Реверсированный массив: ${reversedArray.joinToString(", ")}")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Comm
Медиана array1: 3.0
Медиана array2: 3.5
Медиана array3: 0.0

Process finished with exit code 0
```

8)

```
fun main() {
    val array = arrayOf(5, 3, 7, 1, 9, 4)
```

```
    val elementToFind = 7

    val index = linearSearch(array, elementToFind)

    if (index != -1) {
        println("Элемент $elementToFind найден на индексе $index.")
    } else {
        println("Элемент $elementToFind не найден в массиве.")
    }
}

fun linearSearch(array: Array<Int>, element: Int): Int {
    for (i in array.indices) {
        if (array[i] == element) {
            return i
        }
    }
    return -1
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Progra
Медиана array1: 3.0
Медиана array2: 3.5
Медиана array3: 0.0

Process finished with exit code 0
```

9)

```
fun main() {
    val originalArray = arrayOf(1, 2, 3, 4, 5)
    val copiedArray = originalArray.copyOf()

    println("Original Array: ${originalArray.joinToString()}")
    println("Copied Array: ${copiedArray.joinToString()}")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Program
Медиана array1: 3.0
Медиана array2: 3.5
Медиана array3: 0.0

Process finished with exit code 0
```

10)

```
fun main() {
    val array = intArrayOf(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
    val sumOfEvens = sumOfEvenNumbers(array)
    println("Сумма четных чисел: $sumOfEvens")
}

fun sumOfEvenNumbers(arr: IntArray): Int {
    var sum = 0
    for (num in arr) {
```

```
        if (num % 2 == 0) {
            sum += num
        }
    }
    return sum
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.ex
Медиана array1: 3.0
Медиана array2: 3.5
Медиана array3: 0.0

Process finished with exit code 0
```

11)

```
fun main() {

    val array1 = arrayOf(1, 2, 3, 4, 5)
    val array2 = arrayOf(4, 5, 6, 7, 8)

    val intersection = array1.intersect(array2.toSet())

    val intersectionArray = intersection.toTypedArray()

    println("Пересечение массивов: ${intersectionArray.joinToString(", ")}")
}
```

```
un Alt+4  ers\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Progr
Медиана array1: 3.0
Медиана array2: 3.5
Медиана array3: 0.0

Process finished with exit code 0
```

12)

```
fun <T> swap(array: Array<T>, index1: Int, index2: Int) {
    if (index1 < 0 || index2 < 0 || index1 >= array.size || index2 >=
array.size) {
        throw IndexOutOfBoundsException("Неверные индексы: $index1 или
$index2")
    }
    val temp = array[index1]
    array[index1] = array[index2]
    array[index2] = temp
}

fun main() {
    val myArray = arrayOf(1, 2, 3, 4, 5)
    println("Массив до перестановки: ${myArray.joinToString(", ")}")

    swap(myArray, 1, 3) // Меняем местами элементы с индексами 1 и 3
```

```
    println("Массив после перестановки: ${myArray.joinToString(", ")}")
}
```

Services Alt+8

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\Int
Медиана array1: 3.0
Медиана array2: 3.5
Медиана array3: 0.0

Process finished with exit code 0
```

13)

```
import kotlin.random.Random

fun main() {

    val randomNumbers = IntArray(20) { Random.nextInt(1, 101) }

    println("Случайные числа: ${randomNumbers.joinToString(", ")}")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Commu
Медиана array1: 3.0
Медиана array2: 3.5
Медиана array3: 0.0

Process finished with exit code 0
```

14)

```
fun main() {

    val numbers = arrayOf(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)

    val divisibleByThree = numbers.filter { it % 3 == 0 }

    println("Числа, делящиеся на 3: $divisibleByThree")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Program
Медиана array1: 3.0
Медиана array2: 3.5
Медиана array3: 0.0

Process finished with exit code 0
```

15)

```
fun isPalindrome(array: IntArray): Boolean {
    val n = array.size
    for (i in 0 until n / 2) {
        if (array[i] != array[n - i - 1]) {
            return false
        }
    }
```

```
    return true
}

fun main() {
    val array = intArrayOf(1, 2, 3, 2, 1)
    if (isPalindrome(array)) {
        println("Массив является палиндромом.")
    } else {
        println("Массив не является палиндромом.")
    }
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community E
Медиана array1: 3.0
Медиана array2: 3.5
Медиана array3: 0.0

Process finished with exit code 0
```

16)

```
fun main() {

    val array1 = arrayOf(1, 2, 3)
    val array2 = arrayOf(4, 5, 6)

    val concatenatedArray = array1 + array2

    println(concatenatedArray.joinToString(", "))
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java
1, 2, 3, 4, 5, 6

Process finished with exit code 0
```

17)

```
fun main() {

    val array = intArrayOf(1, 2, 3, 4, 5)

    val sum = array.sum()
    val product = array.fold(1) { acc, i -> acc * i }

    println("Сумма: $sum")
    println("Произведение: $product")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java
Сумма: 15
Произведение: 120

Process finished with exit code 0
```

18)

```kotlin
fun main() {
    val numbers = (1..20).toList()
    val groupedNumbers = numbers.chunked(5)

    for (group in groupedNumbers) {
        println(group)
    }
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\ja
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 17, 18, 19, 20]

Process finished with exit code 0
```

19)

```kotlin
fun mergeSortedArrays(array1: IntArray, array2: IntArray): IntArray {
    val mergedArray = IntArray(array1.size + array2.size)
    var index1 = 0
    var index2 = 0
    var mergedIndex = 0


    while (index1 < array1.size && index2 < array2.size) {
        if (array1[index1] <= array2[index2]) {
            mergedArray[mergedIndex] = array1[index1]
            index1++
        } else {
            mergedArray[mergedIndex] = array2[index2]
            index2++
        }
        mergedIndex++
    }


    while (index1 < array1.size) {
        mergedArray[mergedIndex] = array1[index1]
        index1++
        mergedIndex++
    }
```

```
    while (index2 < array2.size) {
        mergedArray[mergedIndex] = array2[index2]
        index2++
        mergedIndex++
    }

    return mergedArray
}

fun main() {
    val array1 = intArrayOf(1, 3, 5, 7)
    val array2 = intArrayOf(2, 4, 6, 8)

    val mergedArray = mergeSortedArrays(array1, array2)
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.String(", ")}")
Слитый массив: 1, 2, 3, 4, 5, 6, 7, 8

Process finished with exit code 0
```

20)

```
fun main() {
    val start = 1
    val step = 2
    val count = 10

    val arithmeticProgression = IntArray(count) { start + it * step }

    println("Арифметическая прогрессия:
${arithmeticProgression.joinToString()}")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-javaagent:C
Арифметическая прогрессия: 1, 3, 5, 7, 9, 11, 13, 15, 17, 19

Process finished with exit code 0
```

21)

```
fun removeElement(list: MutableList<Int>, element: Int): MutableList<Int> {
    list.remove(element)
    return list
}

fun main() {
    val numbers = mutableListOf(1, 2, 3, 4, 5)
    println("Исходный список: $numbers")

    val elementToRemove = 3
    val updatedList = removeElement(numbers, elementToRemove)
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe
Исходный список: [1, 2, 3, 4, 5]
Обновленный список: [1, 2, 4, 5]

Process finished with exit code 0
```

22)

```
fun findSecondMax(arr: IntArray): Int? {

    val distinctArr = arr.distinct().sorted()


    return if (distinctArr.size < 2) {
        null
    } else {
        distinctArr[distinctArr.size - 2]
    }
}

fun main() {
    val array = intArrayOf(3, 5, 7, 2, 5, 2, 7, 8)
    val secondMax = findSecondMax(array)

    if (secondMax != null) {
        println("Второй по величине элемент: $secondMax")
    } else {
        println("В массиве недостаточно уникальных элементов для нахождения
второго максимального.")
    }
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\ja
Второй по величине элемент: 7

Process finished with exit code 0
```

23)

```
fun mergeArrays(vararg arrays: IntArray): List<Int> {
    return arrays.flatMap { it.asIterable() }
}

fun main() {
    val array1 = intArrayOf(1, 2, 3)
    val array2 = intArrayOf(4, 5, 6)
    val array3 = intArrayOf(7, 8, 9)

    val result = mergeArrays(array1, array2, array3)

    println(result)
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.e
[1, 2, 3, 4, 5, 6, 7, 8, 9]

Process finished with exit code 0
```

24)

```
fun main() {

    val matrix = arrayOf(
        intArrayOf(1, 2, 3),
        intArrayOf(4, 5, 6),
        intArrayOf(7, 8, 9)
    )


    println("Исходная матрица:")
    printMatrix(matrix)


    val transposedMatrix = transpose(matrix)


    println("Транспонированная матрица:")
    printMatrix(transposedMatrix)
}


fun transpose(matrix: Array<IntArray>): Array<IntArray> {
    val rows = matrix.size
    val cols = matrix[0].size
    val transposed = Array(cols) { IntArray(rows) }

    for (i in 0 until rows) {
        for (j in 0 until cols) {
            transposed[j][i] = matrix[i][j]
        }
    }

    return transposed
}


fun printMatrix(matrix: Array<IntArray>) {
    for (row in matrix) {
        for (value in row) {
            print("$value ")
        }
        println()
    }
}
```

```
Транспонированная матрица:
1 4 7
2 5 8
3 6 9

Process finished with exit code 0
```

25)

```
fun linearSearch(array: IntArray, target: Int): Boolean {
    for (element in array) {
        if (element == target) {
            return true
        }
    }
    return false
}

fun main() {
    val array = intArrayOf(1, 5, 2, 8, 3, 9, 4)
    val target1 = 8
    val target2 = 10

    println("Элемент $target1 найден: ${linearSearch(array, target1)}")
    println("Элемент $target2 найден: ${linearSearch(array, target2)}")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\ja
Элемент 8 найден: true
Элемент 10 найден: false

Process finished with exit code 0
```

26)

```
fun average(numbers: DoubleArray): Double {
    if (numbers.isEmpty()) {
        return 0.0
    }
    return numbers.sum() / numbers.size
}

fun main() {
    val numbers = doubleArrayOf(1.0, 2.0, 3.0, 4.0, 5.0)
    val avg = average(numbers)
    println("Среднее арифметическое: $avg")


    val emptyArray = doubleArrayOf()
    val emptyAvg = average(emptyArray)
    println("Среднее арифметическое пустого массива: $emptyAvg")

    val mixedArray = doubleArrayOf(1.0, 2.5, 0.0, -1.5, 5.2)
    val mixedAvg = average(mixedArray)
    println("Среднее арифметическое смешанного массива: $mixedAvg")
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe
Среднее арифметическое: 3.0
Среднее арифметическое пустого массива: 0.0
Среднее арифметическое смешанного массива: 1.44

Process finished with exit code 0
```

27)

```
fun maxSequence(array: IntArray): Int {
    if (array.isEmpty()) return 0

    var maxCount = 1
    var currentCount = 1
    for (i in 1 until array.size) {
        if (array[i] == array[i - 1]) {
            currentCount++
        } else {
            maxCount = maxOf(maxCount, currentCount)
            currentCount = 1
        }
    }
    return maxOf(maxCount, currentCount)
}

fun main() {
    val array1 = intArrayOf(1, 1, 1, 2, 2, 3, 3, 3, 3, 4)
    val array2 = intArrayOf(1, 2, 3, 4, 5)
    val array3 = intArrayOf(1,1,1,1,1,1)
    val array4 = intArrayOf()

    println("Максимальная последовательность в array1:
${maxSequence(array1)}")
    println("Максимальная последовательность в array2:
${maxSequence(array2)}")
    println("Максимальная последовательность в array3:
${maxSequence(array3)}")
    println("Максимальная последовательность в array4:
${maxSequence(array4)}")

}
```

```
Максимальная последовательность в array1: 4
Максимальная последовательность в array2: 1
Максимальная последовательность в array3: 6
Максимальная последовательность в array4: 0

Process finished with exit code 0
```

28)

```
fun main() {
    print("Введите количество элементов в массиве: ")
    val n = readLine()?.toIntOrNull() ?: 0

    if (n <= 0) {
        println("Количество элементов должно быть больше 0.")
```

```
        return
    }

    val numbers = DoubleArray(n)
    for (i in 0 until n) {
        while (true) {
            print("Введите элемент ${i + 1}: ")
            val input = readLine()
            val number = input?.toDoubleOrNull()
            if (number != null) {
                numbers[i] = number
                break
            } else {
                println("Некорректный ввод. Попробуйте ещё раз.")
            }
        }
    }

    println("Вы ввели массив: ${numbers.contentToString()}")
}
```

```
Введите элемент 32: 34
Вы ввели массив: [3.0, 2.0, 4.0, 5.0, 6.0, 8.0, 0.0, 9
4

Process finished with exit code 0
```

29)

```
fun median(array: DoubleArray): Double {
    if (array.isEmpty()) return 0.0

    val sortedArray = array.sortedArray()
    val mid = sortedArray.size / 2
    return if (sortedArray.size % 2 == 1) {
        sortedArray[mid]
    } else {
        (sortedArray[mid - 1] + sortedArray[mid]) / 2.0
    }
}

fun main() {
    val array1 = doubleArrayOf(1.0, 3.0, 5.0, 2.0, 4.0)
    val array2 = doubleArrayOf(1.0, 3.0, 5.0, 2.0, 4.0, 6.0)
    val array3 = doubleArrayOf()

    println("Медиана array1: ${median(array1)}")
    println("Медиана array2: ${median(array2)}")
    println("Медиана array3: ${median(array3)}")
}
```

```
Медиана array1: 3.0
Медиана array2: 3.5
Медиана array3: 0.0

Process finished with exit code 0
```

30)

```
fun groupNumbers(numbers: IntArray): List<IntArray> {
    if (numbers.size != 100) {
        throw IllegalArgumentException("Массив должен содержать 100
элементов.")
    }
    val groups = mutableListOf<IntArray>()
    for (i in 0 until 10) {
        val group = IntArray(10)
        for (j in 0 until 10) {
            group[j] = numbers[i * 10 + j]
        }
        groups.add(group)
    }
    return groups
}

fun main() {
    val numbers = IntArray(100) { it + 1 }
    val groups = groupNumbers(numbers)

    for (i in groups.indices) {
        println("Группа ${i + 1}: ${groups[i].contentToString()}")
    }
}
```

```
C:\Users\Student\.jdks\openjdk-22.0.2\bin\java.exe "-ja
Группа 1: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Группа 2: [11, 12, 13, 14, 15, 16, 17, 18, 19, 20]
Группа 3: [21, 22, 23, 24, 25, 26, 27, 28, 29, 30]
Группа 4: [31, 32, 33, 34, 35, 36, 37, 38, 39, 40]
Группа 5: [41, 42, 43, 44, 45, 46, 47, 48, 49, 50]
Группа 6: [51, 52, 53, 54, 55, 56, 57, 58, 59, 60]
Группа 7: [61, 62, 63, 64, 65, 66, 67, 68, 69, 70]
Группа 8: [71, 72, 73, 74, 75, 76, 77, 78, 79, 80]
Группа 9: [81, 82, 83, 84, 85, 86, 87, 88, 89, 90]
Группа 10: [91, 92, 93, 94, 95, 96, 97, 98, 99, 100]

Process finished with exit code 0
```