

1. Создание и вывод элементов: Создайте массив из 5 целых чисел и выведите их на экран.

```
fun main() {  
    val array = intArrayOf(10, 20, 30, 40, 500)  
    println("Элементы массива: ${array.joinToString()}")  
}
```

```
Элементы массива: 10, 20, 30, 40, 500  
Process finished with exit code 0
```

2. Сумма элементов массива: Напишите программу, которая находит сумму всех элементов массива чисел.

```
fun main() {  
    // Создаем массив чисел  
    val numbers = arrayOf(10, 20, 30, 40, 500)  
  
    // Вычисляем сумму элементов массива  
    val sum = numbers.sum()  
  
    // Выводим результат  
    println("Сумма элементов массива: $sum")  
}
```

```
Сумма элементов массива: 600  
Process finished with exit code 0
```

3. Максимальное и минимальное значение: Создайте массив из 10 чисел, найдите и выведите максимальное и минимальное значение.

```
fun main() {  
    // Создаем массив из 10 чисел
```

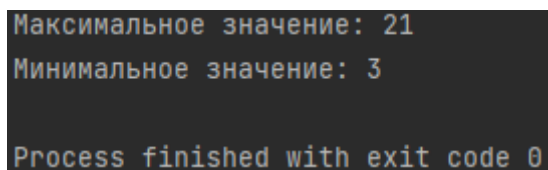
```

val numbers = arrayOf(12, 5, 8, 21, 3, 17, 9, 14, 6, 11)

// Находим максимальное и минимальное значения
val max = numbers.maxOrNull()
val min = numbers.minOrNull()

// Выводим результаты
println("Максимальное значение: $max")
println("Минимальное значение: $min")
}

```



```

Максимальное значение: 21
Минимальное значение: 3
Process finished with exit code 0

```

4. Сортировка массива: Реализуйте алгоритм сортировки для массива чисел и выведите отсортированный массив.

```

fun main() {
    // Создаем массив чисел
    val numbers = arrayOf(64, 34, 25, 12, 22, 11, 90)

    // Сортировка пузырьком
    for (i in numbers.indices) {
        for (j in 0 until numbers.size - i - 1) {
            if (numbers[j] > numbers[j + 1]) {
                // Меняем местами
                val temp = numbers[j]
                numbers[j] = numbers[j + 1]
                numbers[j + 1] = temp
            }
        }
    }
}

```

```

    }
}

// Выводим отсортированный массив
println("Отсортированный массив: ${numbers.joinToString(", ")}")
}

```

```

Отсортированный массив: 11, 12, 22, 25, 34, 64, 90
Process finished with exit code 0

```

5. Уникальные элементы: Напишите программу, которая выводит уникальные элементы из массива.

```

fun main() {
    // Создаем массив с дублирующимися элементами
    val numbers = arrayOf(1, 2, 3, 2, 410, 5, 1, 6, 4, 7)

    // Находим уникальные элементы с помощью множества
    val uniqueNumbers = numbers.toSet()

    // Выводим уникальные элементы
    println("Уникальные элементы: ${uniqueNumbers.joinToString(", ")}")
}

```

```

Уникальные элементы: 1, 2, 3, 410, 5, 6, 4, 7
Process finished with exit code 0

```

6. Четные и нечетные числа: Создайте массив и разделите его на четные и нечетные числа, сохранив их в разные массивы.

```

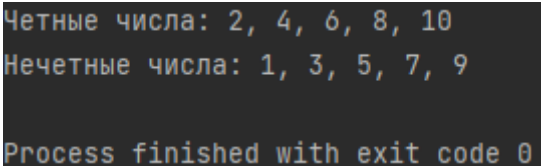
fun main() {
    // Создаем массив чисел
    val numbers = arrayOf(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

    // Создаем списки для четных и нечетных чисел
    val evenNumbers = mutableListOf<Int>()
    val oddNumbers = mutableListOf<Int>()

    // Разделяем числа на четные и нечетные
    for (number in numbers) {
        if (number % 2 == 0) {
            evenNumbers.add(number)
        } else {
            oddNumbers.add(number)
        }
    }

    // Выводим четные и нечетные числа
    println("Четные числа: ${evenNumbers.joinToString(", ")}")
    println("Нечетные числа: ${oddNumbers.joinToString(", ")}")
}

```



```

Четные числа: 2, 4, 6, 8, 10
Нечетные числа: 1, 3, 5, 7, 9
Process finished with exit code 0

```

7. Реверс массива: Напишите программу, которая реверсирует массив чисел.

```

fun main() {
    // Создаем массив чисел
    val numbers = arrayOf(1, 2, 3, 4, 5)

    // Реверсируем массив
    val reversedNumbers = numbers.reversedArray()

    // Выводим реверсированный массив
    println("Реверсированный массив: ${reversedNumbers.joinToString(", ")}")
}

```

```

Реверсированный массив: 5, 4, 3, 2, 1
Process finished with exit code 0

```

8. Поиск элемента: Реализуйте поиск элемента в массиве и выводите его индекс.

```

fun main() {
    // Создаем массив чисел
    val numbers = arrayOf(10, 20, 30, 40, 50)

    // Элемент для поиска
    val target = 30

    // Поиск элемента и вывод индекса
    val index = numbers.indexOf(target)

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

```

```
}  
}
```

```
Элемент 30 найден на индексе 2.  
Process finished with exit code 0
```

9. Копирование массива: Создайте новый массив, скопировав в него элементы из другого массива.

```
fun main() {  
    // Создаем исходный массив  
    val originalArray = arrayOf(1, 2, 3, 4, 5)  
  
    // Копируем элементы в новый массив  
    val copiedArray = originalArray.copyOf()  
  
    // Выводим оригинальный и скопированный массивы  
    println("Исходный массив: ${originalArray.joinToString(", ")}")  
    println("Скопированный массив: ${copiedArray.joinToString(", ")}")  
}
```

```
Исходный массив: 1, 2, 3, 4, 5  
Скопированный массив: 1, 2, 3, 4, 5  
Process finished with exit code 0
```

10. Сумма четных чисел: Напишите программу, которая находит сумму всех четных чисел в массиве.

```
fun sumOfEvenNumbers(arr: IntArray): Int {
```

```

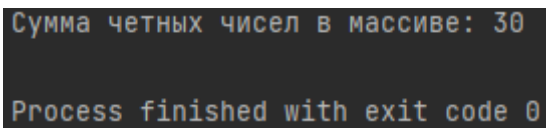
var totalSum = 0

for (num in arr) {
    if (num % 2 == 0) {
        totalSum += num
    }
}

return totalSum
}

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

```



```

Сумма четных чисел в массиве: 30
Process finished with exit code 0

```

11. Пересечение массивов: Напишите программу, которая находит пересечение двух массивов и выводит результат.

```

fun intersection(arr1: IntArray, arr2: IntArray): IntArray {
    return arr1.intersect(arr2.asIterable()).toIntArray()
}

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

```

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

```
Пересечение массивов: 4, 5
Process finished with exit code 0
```

12. Перестановка элементов: Реализуйте функцию, которая меняет местами два элемента в массиве.

```
fun swapElements(arr: IntArray, index1: Int, index2: Int) {
    val temp = arr[index1]
    arr[index1] = arr[index2]
    arr[index2] = temp
}

fun main() {
    val array = intArrayOf(1, 2, 3, 4, 5)
    swapElements(array, 1, 3)
    println("Массив после перестановки: ${array.joinToString()}")
}
```

```
Массив после перестановки: 1, 4, 3, 2, 5
Process finished with exit code 0
```

13. Заполнение случайными числами: Создайте массив из 20 случайных чисел от 1 до 100 и выведите его на экран.

```
import kotlin.random.Random
```



```

fun main() {
    val randomArray = IntArray(20) { Random.nextInt(1, 101) }
    println("Случайные числа: ${randomArray.joinToString()}")
}

```

```

Случайные числа: 88, 24, 4, 25, 69, 93, 70, 38, 95, 67, 2, 71, 20, 64, 96, 57, 21, 78
Process finished with exit code 0

```

14. Числа Прокопенко: Напишите программу, которая выводит все числа в массиве, делящиеся на 3.

```

fun printDivisibleByThree(arr: IntArray) {
    val divisibleNumbers = arr.filter { it % 3 == 0 }
    if (divisibleNumbers.isNotEmpty()) {
        println("Числа, делящиеся на 3: ${divisibleNumbers.joinToString()}")
    } else {
        println("Нет чисел, делящихся на 3.")
    }
}

```

```

fun main() {
    val array = intArrayOf(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 15, 18)
    printDivisibleByThree(array)
}

```

```

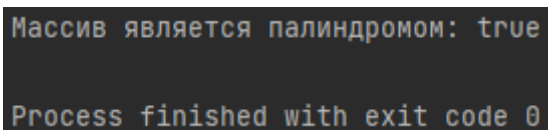
Числа, делящиеся на 3: 3, 6, 9, 12, 15, 18
Process finished with exit code 0

```

15.Проверка на палиндром: Напишите программу, которая проверяет, является ли массив палиндромом.

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

```
fun main() {  
    val array = intArrayOf(1, 2, 3, 2, 1)  
    println("Массив является палиндромом: ${isPalindrome(array)}")  
}
```



```
Массив является палиндромом: true  
Process finished with exit code 0
```

16.Конкатенация двух массивов: Создайте два массива и соедините их в один.

```
fun concatenateArrays(arr1: IntArray, arr2: IntArray): IntArray {  
    return arr1 + arr2  
}
```

```

fun main() {
    val array1 = intArrayOf(1, 2, 3)
    val array2 = intArrayOf(4, 5, 6)
    val concatenatedArray = concatenateArrays(array1, array2)
    println("Объединенный массив: ${concatenatedArray.joinToString()}")
}

```

```

Объединенный массив: 1, 2, 3, 4, 5, 6
Process finished with exit code 0

```

17. Сумма и произведение: Напишите программу, которая находит и выводит сумму и произведение всех элементов массива.

```

fun sumAndProduct(arr: IntArray): Pair<Int, Int> {
    val sum = arr.sum()
    val product = arr.fold(1) { acc, num -> acc * num }
    return Pair(sum, product)
}

```

```

fun main() {
    val array = intArrayOf(1, 2, 3, 4)
    val (sum, product) = sumAndProduct(array)
    println("Сумма: $sum, Произведение: $product")
}

```

```

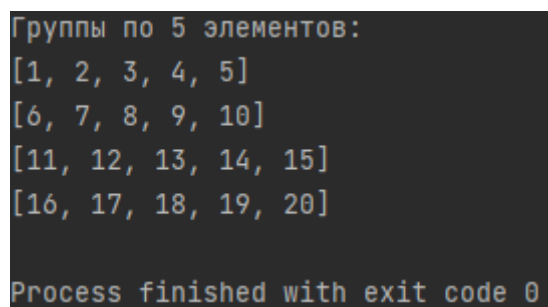
Сумма: 10, Произведение: 24
Process finished with exit code 0

```

18.Группировка чисел: Разделите массив на группы по 5 элементов и выведите их.

```
fun groupNumbers(arr: IntArray, groupSize: Int) {  
    arr.toList().chunked(groupSize).forEach { println(it) }  
}
```

```
fun main() {  
    val array = IntArray(20) { it + 1 }  
    println("Группы по 5 элементов:")  
    groupNumbers(array, 5)  
}
```



```
Группы по 5 элементов:  
[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.Слияние двух массивов: Напишите программу, которая сливает два отсортированных массива в один отсортированный массив.

```
fun mergeSortedArrays(arr1: IntArray, arr2: IntArray): IntArray {  
    return (arr1 + arr2).sortedArray()  
}
```

```
fun main() {  
    val array1 = intArrayOf(1, 3, 5)  
    val array2 = intArrayOf(2, 4, 6)  
    val result = mergeSortedArrays(array1, array2)  
    println("Слитый отсортированный массив: ${result.joinToString()}")  
}
```

```
}
```

```
Слитый отсортированный массив: 1, 2, 3, 4, 5, 6  
Process finished with exit code 0
```

20.Числовая последовательность: Создайте массив целых чисел, представляющий арифметическую прогрессию, и выведите его.

```
fun arithmeticProgression(start: Int, step: Int, count: Int): IntArray {  
    return IntArray(count) { start + it * step }  
}  
  
fun main() {  
    val progression = arithmeticProgression(1, 3, 10)  
    println("Арифметическая прогрессия: ${progression.joinToString()}")  
}
```

```
Арифметическая прогрессия: 1, 4, 7, 10, 13, 16, 19, 22, 25, 28  
Process finished with exit code 0
```

21.Удаление элемента: Реализуйте функцию, которая удаляет заданный элемент из массива.

```
fun removeElement(arr: IntArray, element: Int): IntArray {  
    return arr.filter { it != element }.toIntArray()  
}
```

```
fun main() {
    val array = intArrayOf(1, 2, 3, 4, 5)
    val result = removeElement(array, 3)
    println("Массив после удаления элемента: ${result.joinToString()}")
}
```

```
Массив после удаления элемента: 1, 2, 4, 5
Process finished with exit code 0
```

22. Поиск второго максимального: Напишите программу, которая находит второй по величине элемент в массиве.

```
fun secondMax(arr: IntArray): Int? {
    val distinctElements = arr.distinct().sorted()
    return if (distinctElements.size >= 2) distinctElements[distinctElements.size - 2] else null
}
```

```
fun main() {
    val array = intArrayOf(1, 2, 3, 4, 5)
    println("Второй максимальный элемент: ${secondMax(array)}")
}
```

```
Второй максимальный элемент: 4
Process finished with exit code 0
```

23. Объединение массивов: Напишите функцию, которая объединяет несколько массивов чисел и выводит результирующий массив.

```

fun mergeMultipleArrays(vararg arrays: IntArray): IntArray {
    // Вычисляем общий размер результирующего массива
    val totalSize = arrays.sumOf { it.size }
    val result = IntArray(totalSize)

    var currentIndex = 0
    for (array in arrays) {
        for (element in array) {
            result[currentIndex++] = element
        }
    }

    return result
}

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

    val result = mergeMultipleArrays(array1, array2, array3)
    println("Объединенный массив: ${result.joinToString()}")
}

```

```

Объединенный массив: 1, 2, 3, 4, 5, 6, 7, 8, 9
Process finished with exit code 0

```

24. Транспонирование матрицы: Создайте матрицу (двумерный массив) и напишите программу, которая транспонирует её.

```

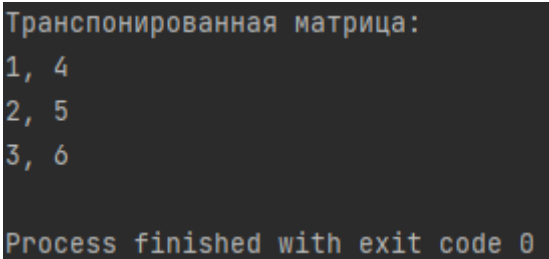
fun transposeMatrix(matrix: Array<IntArray>): Array<IntArray> {
    val rows = matrix.size
    val cols = matrix[0].size
    val transposed = Array(cols) { IntArray(rows) }

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

    return transposed
}

fun main() {
    val matrix = arrayOf(
        intArrayOf(1, 2, 3),
        intArrayOf(4, 5, 6)
    )
    val transposed = transposeMatrix(matrix)
    println("Транспонированная матрица:")
    transposed.forEach { println(it.joinToString()) }
}

```



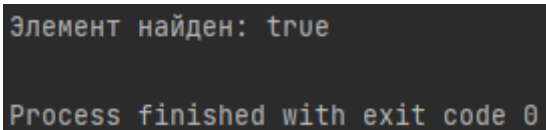
```

Транспонированная матрица:
1, 4
2, 5
3, 6
Process finished with exit code 0

```


25.Линейный поиск: Реализуйте линейный поиск элемента в массиве с возвратомBool-значения (найден или нет).

```
fun linearSearch(arr: IntArray, target: Int): Boolean {  
    return arr.contains(target)  
}  
  
fun main() {  
    val array = intArrayOf(1, 2, 3, 4, 5)  
    println("Элемент найден: ${linearSearch(array, 3)}")  
}
```



```
Элемент найден: true  
Process finished with exit code 0
```

26.Среднее арифметическое: Напишите программу, которая находит среднее арифметическое всех чисел в массиве.

```
fun average(arr: IntArray): Double {  
    return arr.average()  
}  
  
fun main() {  
    val array = intArrayOf(1, 2, 3, 4, 5)  
    println("Среднее арифметическое: ${average(array)}")  
}
```

```
Среднее арифметическое: 3.0
```

```
Process finished with exit code 0
```

27.Максимальная последовательность: Найдите максимальную последовательность одинаковых элементов в массиве.

```
fun maxSequence(arr: IntArray): Pair<Int, Int> {  
    var maxCount = 0  
    var currentCount = 1  
    var maxElement = arr[0]  
  
    for (i in 1 until arr.size) {  
        if (arr[i] == arr[i - 1]) {  
            currentCount++  
        } else {  
            if (currentCount > maxCount) {  
                maxCount = currentCount  
                maxElement = arr[i - 1]  
            }  
            currentCount = 1  
        }  
    }  
  
    if (currentCount > maxCount) {  
        maxCount = currentCount  
        maxElement = arr[arr.size - 1]  
    }  
  
    return Pair(maxElement, maxCount)  
}
```

```
fun main() {
    val array = intArrayOf(1, 1, 2, 2, 2, 3, 3)
    val (element, count) = maxSequence(array)
    println("Максимальная последовательность: элемент $element, количество $count")
}
```

```
Максимальная последовательность: элемент 2, количество 3
Process finished with exit code 0
```

28. Ввод и вывод массива: Напишите программу, которая запрашивает у пользователя ввод массива чисел и затем выводит его.

```
fun main() {
    println("Введите числа через пробел:")
    val input = readLine()!!
    val array = input.split(" ").map { it.toInt() }.toIntArray()
    println("Вы ввели массив: ${array.joinToString()}")
}
```

```
Введите числа через пробел:
10 20 30 40
Вы ввели массив: 10, 20, 30, 40
Process finished with exit code 0
```

29. Нахождение медианы: Напишите программу, которая находит медиану в массиве.

```
fun median(arr: IntArray): Double {
```

```

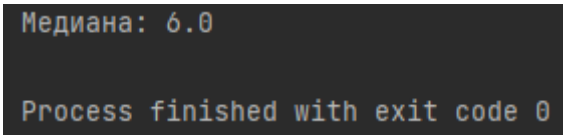
val sorted = arr.sortedArray()
return if (sorted.size % 2 == 0) {
    (sorted[sorted.size / 2 - 1] + sorted[sorted.size / 2]) / 2.0
} else {
    sorted[sorted.size / 2].toDouble()
}
}

```

```

fun main() {
    val array = intArrayOf(1, 3, 3, 6, 7, 8, 9)
    println("Медиана: ${median(array)}")
}

```



```

Медиана: 6.0

```

```

Process finished with exit code 0

```

30. Распределение по группам: Создайте массив из 100 целых чисел и разделите их на 10 групп по 10 элементов, затем выведите результаты

```

fun groupIntegers(arr: IntArray, groupSize: Int) {
    arr.toList().chunked(groupSize).forEach { println(it) }
}

```

```

fun main() {
    val array = IntArray(100) { it + 1 }
    println("Группы по 10 элементов:")
    groupIntegers(array, 10)
}

```

Группы по 10 элементов:

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

Process finished with exit code 0