

Лабораторная работа №6

Харахардин А. ИС233

The image shows a Kotlin program in IntelliJ IDEA. The code performs various operations on arrays, including creation, summing, finding min/max, sorting, deduplication, partitioning by parity, reversing, indexing, copying, filtering, intersecting, swapping, and randomization.

```
1 import kotlin.random.Random
2
3 fun main() {
4     // 1. Создание и вывод элементов
5     val arr1 = IntArrayOf(5, 3, 8, 1, 9)
6     println("1. Массив: ${arr1.joinToString()}")
7
8     // 2. Сумма элементов массива
9     val sum = arr1.sum()
10    println("2. Сумма элементов: $sum")
11
12    // 3. Максимальное и минимальное значение
13    val arr3 = IntArrayOf(5, 3, 8, 1, 9, 4, 7, 2, 6, 10)
14    println("3. Максимальное: ${arr3.maxOrNull()}, Минимальное: ${arr3.minOrNull()}")
15
16    // 4. Сортировка массива
17    val arr4 = arr3.copyOf()
18    arr4.sort()
19    println("4. Отсортированный массив: ${arr4.joinToString()}")
20
21    // 5. Уникальные элементы
22    val arr5 = IntArrayOf(1, 2, 2, 3, 4, 4, 5)
23    val unique = arr5.distinct()
24    println("5. Уникальные элементы: ${unique.joinToString()}")
25
26    // 6. Четные и нечетные числа
27    val (even, odd) = arr5.partition { it % 2 == 0 }
28    println("6. Четные: ${even.joinToString()}, Нечетные: ${odd.joinToString()}")
29
30    // 7. Реверс массива
31    val arr7 = arr5.copyOf()
32    arr7.reverse()
33    println("7. Реверс массива: ${arr7.joinToString()}")
34
35    // 8. Поиск элемента
36    val index = arr3.indexOf(8)
37    println("8. Индекс элемента 8: $index")
38
39    // 9. Копирование массива
40    val arr9 = arr3.copyOf()
41    println("9. Скопированный массив: ${arr9.joinToString()}")
42
43    // 10. Сумма четных чисел
44    val sumEven = arr3.filter { it % 2 == 0 }.sum()
45    println("10. Сумма четных чисел: $sumEven")
46
47    // 11. Пересечение массивов
48    val arr1a = IntArrayOf(1, 2, 3, 4, 5)
49    val arr1b = IntArrayOf(4, 5, 6, 7, 8)
50    val intersect = arr1a.intersect(arr1b.toList())
51    println("11. Пересечение массивов: ${intersect.joinToString()}")
52
53    // 12. Перестановка элементов
54    val arr12 = arr3.copyOf()
55    arr12.swap(0, 4)
56    println("12. После перестановки: ${arr12.joinToString()}")
57
58    // 13. Заполнение случайными числами
```

The Run console shows the following output:

```
7. Реверс массива: 10, 6, 2, 7, 4, 9, 1, 8, 3, 5
8. Индекс элемента 8: 2
9. Скопированный массив: 5, 3, 8, 1, 9, 4, 7, 2, 6, 10
10. Сумма четных чисел: 30
11. Пересечение массивов: 4, 5
12. После перестановки: 5, 8, 1, 9, 4, 7, 2, 6, 18
13. Случайные числа: 93, 18, 58, 39, 1, 26, 31, 62, 52, 59, 28, 65, 1, 31, 67, 56, 68, 32, 58, 55
14. Числа, делящиеся на 3: 3, 9, 6
```

```
1 fun main() {
2     // 13. Случайные числа
3     val arr13 = IntArray(20) { Random.nextInt(1000, 1001)}
4     println("13. Случайные числа: ${arr13.joinToString()}")
5
6     // 14. Числа, делимые на 3
7     val divisibleBy3 = arr13.filter { it % 3 == 0 }
8     println("14. Числа, делимые на 3: ${divisibleBy3.joinToString()}")
9
10    // 15. Проверка на палиндром
11    val arr15 = IntArrayOf(1, 2, 3, 2, 1)
12    val isPalindrome = arr15.contentEquals(arr15.reversedArray())
13    println("15. Массив ${if (isPalindrome) "является" else "не является"} палиндромом")
14
15    // 16. Конкатенация двух массивов
16    val arr16a = IntArrayOf(1, 2, 3)
17    val arr16b = IntArrayOf(4, 5, 6)
18    val concatenated = arr16a + arr16b
19    println("16. Конкатенация: ${concatenated.joinToString()}")
20
21    // 17. Сумма и произведение
22    val sum17 = arr3.sum()
23    val product17 = arr3.fold(1) { acc, i -> acc * i }
24    println("17. Сумма: $sum17, Произведение: $product17")
25
26    // 18. Группировка чисел
27    val arr18 = IntArray(25) { it + 1 }
28    val groups = arr18.toList().chunked(5)
29    println("18. Группы по 5 элементов:")
30    groups.forEachIndexed { i, group -> println("    группа ${i + 1}: ${group.joinToString()}") }
31}
```

Run MainKt

13. Случайные числа: 95, 16, 56, 39, 1, 26, 31, 62, 32, 59, 28, 65, 1, 51, 67, 56, 68, 32, 58, 55
14. Числа, делимые на 3: 3, 9, 6
15. Массив является палиндромом
16. Конкатенация: 1, 2, 3, 4, 5, 6
17. Сумма: 55, Произведение: 3628800
18. Группы по 5 элементов:
 группа 1: 1, 2, 3, 4, 5
 группа 2: 6, 7, 8, 9, 10

```
1 fun main() {
2     // 19. Слияние двух отсортированных массивов
3     val arr19a = IntArrayOf(1, 5, 5, 7)
4     val arr19b = IntArrayOf(2, 4, 6, 8)
5     val merged = (arr19a + arr19b).sortedArray()
6     println("19. Слияние массивов: ${merged.joinToString()}")
7
8     // 20. Арифметическая прогрессия
9     val arr20 = IntArray(10) { 1 + it * 3 }
10    println("20. Арифметическая прогрессия: ${arr20.joinToString()}")
11
12    // 21. Удаление элемента
13    val arr21 = arr3.copy().toMutableList()
14    arr21.removeAt(1000, 2)
15    println("21. После удаления элемента: ${arr21.joinToString()}")
16
17    // 22. Второй максимальный элемент
18    val sorted = arr3.sortedDescending()
19    val secondMax = sorted[1]
20    println("22. Второй максимальный элемент: $secondMax")
21
22    // 23. Объединение массивов
23    val arr23a = IntArrayOf(1, 2, 3)
24    val arr23b = IntArrayOf(4, 5)
25    val arr23c = IntArrayOf(6, 7, 8, 9)
26    val combined = arr23a + arr23b + arr23c
27    println("23. Объединение массивов: ${combined.joinToString()}")
28
29    // 24. Транспонирование матрицы
30    val matrix = arrayOf(
31        1, 2, 3, 4, 5, 6, 7, 8, 9
32    )
33}
```

Run MainKt

19. Слияние массивов: 1, 2, 3, 4, 5, 6, 7, 8
20. Арифметическая прогрессия: 1, 4, 7, 10, 13, 16, 19, 22, 25, 28
21. После удаления элемента: 5, 3, 1, 9, 4, 7, 2, 6, 18
22. Второй максимальный элемент: 9
23. Объединение массивов: 1, 2, 3, 4, 5, 6, 7, 8, 9
24. Транспонированная матрица:

```
1 fun main() {
2
3     // 24. Транспонирование матрицы
4     val matrix = arrayOf(
5         intArrayOf(1, 2, 3),
6         intArrayOf(4, 5, 6),
7         intArrayOf(7, 8, 9)
8     )
9     val transposed = transpose(matrix)
10    println("24. Транспонированная матрица:")
11    transposed.forEach { println("  ${it.joinToString()}") }
12
13    // 25. Линейный поиск
14    val found = arr3.contains(5)
15    println("25. Элемент 5 ${if (found) "найден" else "не найден"}")
16
17    // 26. Среднее арифметическое
18    val average = arr3.average()
19    println("26. Среднее арифметическое: $average")
20
21    // 27. Максимальная последовательность
22    val arr27 = intArrayOf(1, 2, 2, 3, 3, 3, 2, 2, 1)
23    val maxSequence = findMaxSequence(arr27)
24    println("27. Максимальная последовательность: ${maxSequence.joinToString()}")
25
26    // 28. Ввод и вывод массива (симуляция ввода)
27    val inputArray = intArrayOf(10, 20, 30, 40, 50) // Предположим, что пользователь ввел эти числа
28    println("28. Введенный массив: ${inputArray.joinToString()}")
29
30    // 29. Медиана
31    val arr29 = arr3.copyOf().apply { sort() }
32
33    // 30. Распределение по группам
34    val arr30 = IntArray(60) { it + 1 }
35    val groups30 = arr30.toList().chunked(10)
36    println("30. Группы по 10 элементов:")
37    groups30.forEachIndexed { i, group -> println("  группа ${i + 1}: ${group.joinToString()}") }
38
39    // 31. Вспомогательная функция для транспонирования
40    fun transpose(matrix: Array<IntArray>): Array<IntArray> {
41        val rows = matrix.size
42        val cols = matrix[0].size
43        return Array(cols) { j -> IntArray(rows) { i -> matrix[i][j] } }
44    }
45
46    // 32. Вспомогательная функция для поиска максимальной последовательности
47    fun findMaxSequence(arr: IntArray): List<Int> {
48        if (arr.isEmpty()) return emptyList()
49        var current = arr[0]
50    }
```

Run MainKt

```
24. Транспонированная матрица:
  1, 4, 7
  2, 5, 8
  3, 6, 9
25. Элемент 5 найден
26. Среднее арифметическое: 5.5
27. Максимальная последовательность: 3, 3, 3
28. Введенный массив: 10, 20, 30, 40, 50
```

```
1 fun main() {
2
3     // 29. Медиана
4     val arr29 = arr3.copyOf().apply { sort() }
5     val median = if (arr29.size % 2 == 1) arr29[arr29.size / 2].toDouble()
6     else (arr29[arr29.size / 2 - 1] + arr29[arr29.size / 2]) / 2.0
7     println("29. Медиана: $median")
8
9     // 30. Распределение по группам
10    val arr30 = IntArray(60) { it + 1 }
11    val groups30 = arr30.toList().chunked(10)
12    println("30. Группы по 10 элементов:")
13    groups30.forEachIndexed { i, group -> println("  группа ${i + 1}: ${group.joinToString()}") }
14
15    // 31. Вспомогательная функция для транспонирования
16    fun transpose(matrix: Array<IntArray>): Array<IntArray> {
17        val rows = matrix.size
18        val cols = matrix[0].size
19        return Array(cols) { j -> IntArray(rows) { i -> matrix[i][j] } }
20    }
21
22    // 32. Вспомогательная функция для поиска максимальной последовательности
23    fun findMaxSequence(arr: IntArray): List<Int> {
24        if (arr.isEmpty()) return emptyList()
25        var current = arr[0]
26    }
```

Run MainKt

```
29. Медиана: 5.5
30. Группы по 10 элементов:
  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
```