Практическая работа № 6.

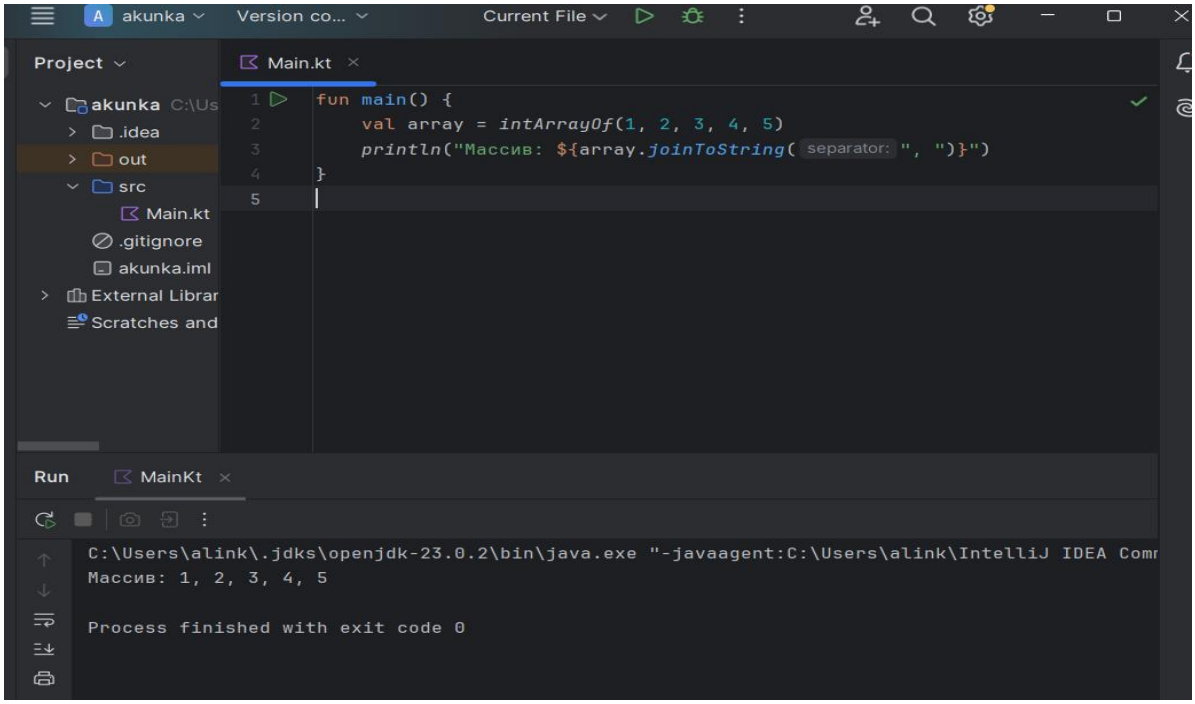
Выполнили : Андрухова , Загородняя.

1.fun main() {

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

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

}



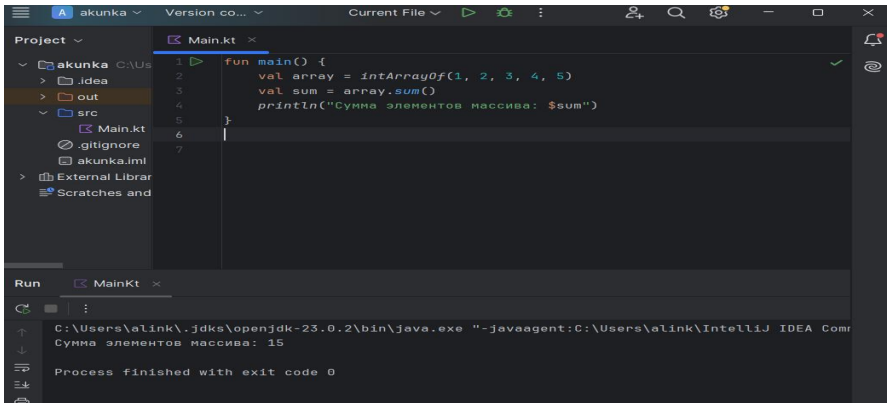
2.fun main() {

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

    val sum = array.sum()

    println("Сумма элементов массива: $sum")

}



3.fun main() {

    val array = intArrayOf(10, 5, 8, 3, 9, 7, 2, 1, 6, 4)

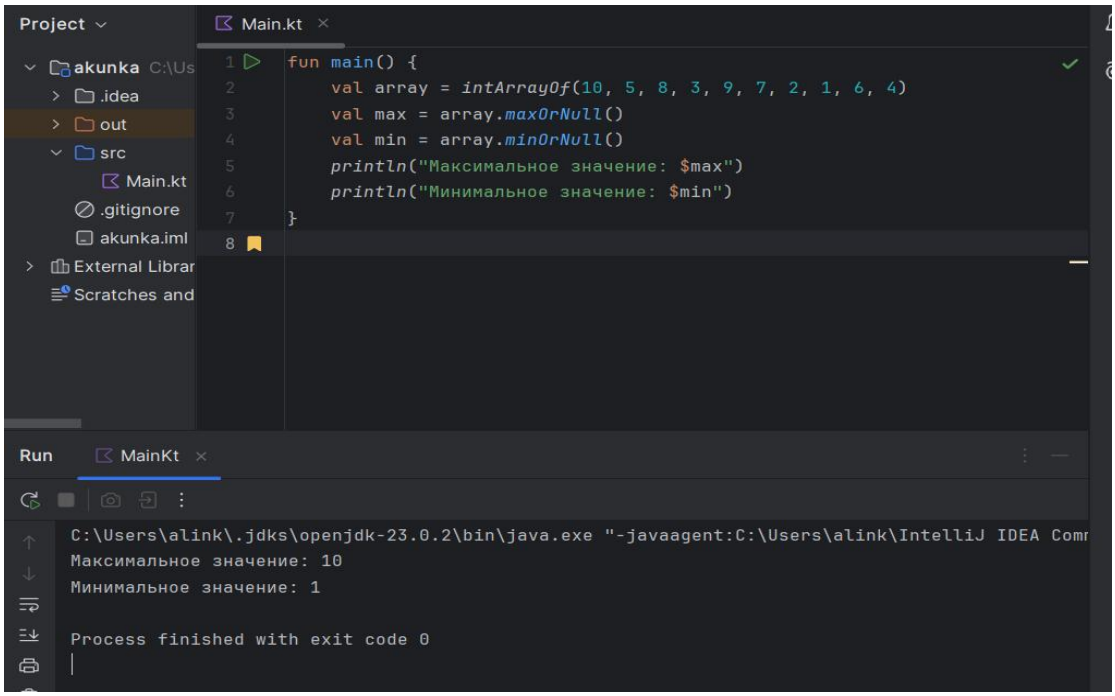
    val max = array.maxOrNull()

    val min = array.minOrNull()

    println("Максимальное значение: $max")

    println("Минимальное значение: $min")

}



4.

fun main() {

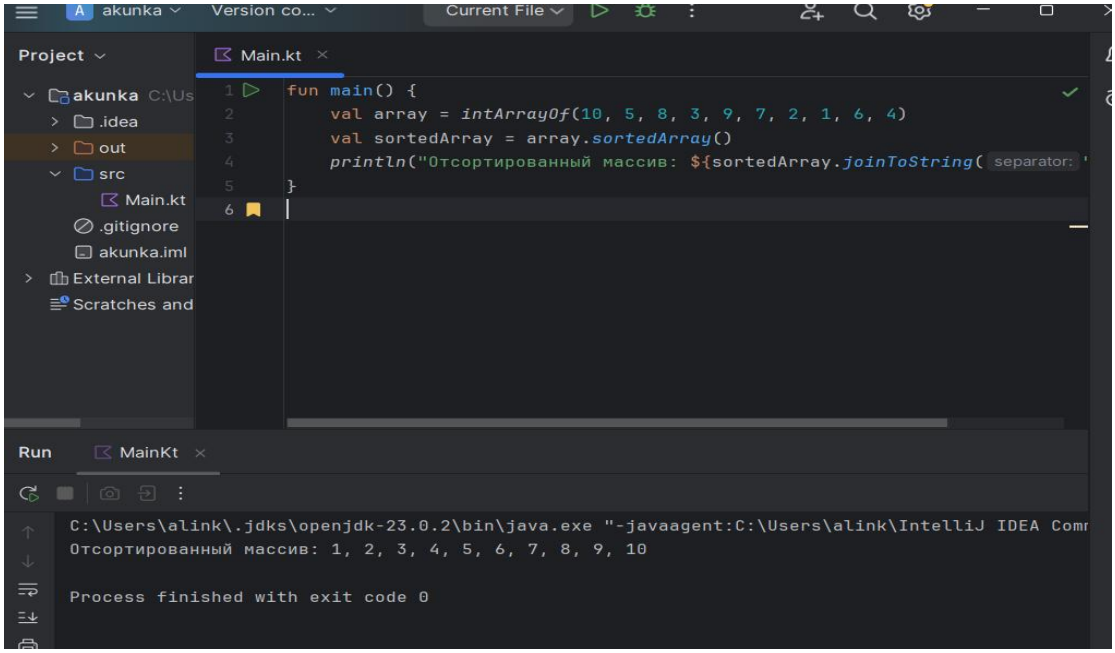
    val array = intArrayOf(10, 5, 8, 3, 9, 7, 2, 1, 6, 4)

    val sortedArray = array.sortedArray()

    println("Отсортированный массив: ${sortedArray.joinToString(",

")}")

}



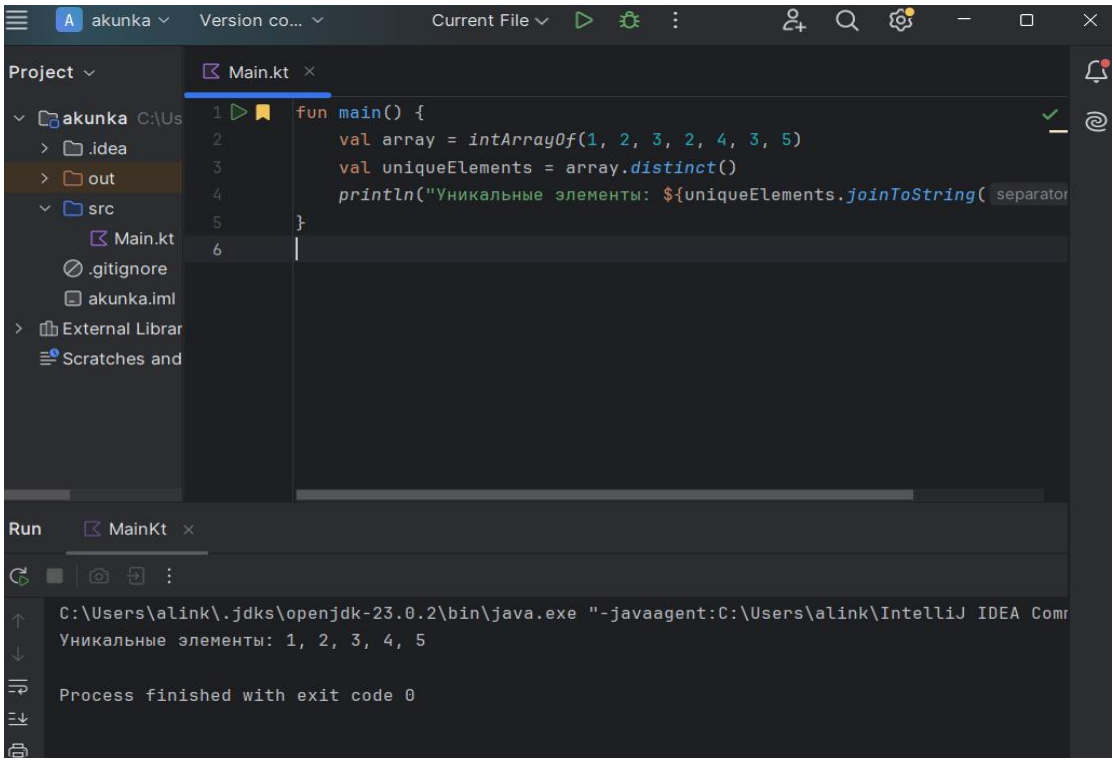
5.fun main() {

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

val uniqueElements = array.distinct()

println("Уникальные элементы: ${uniqueElements.joinToString(", ")}")

}



6.fun main() {

    val array = intArrayOf(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

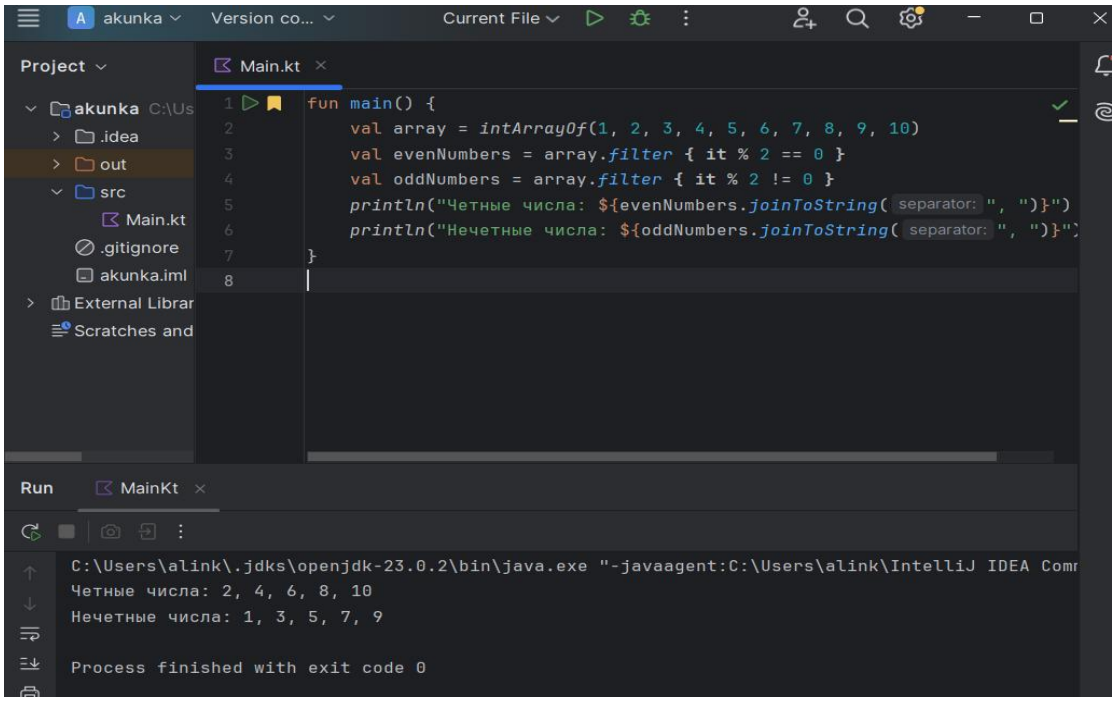
    val evenNumbers = array.filter { it % 2 == 0 }

    val oddNumbers = array.filter { it % 2 != 0 }

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

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

}



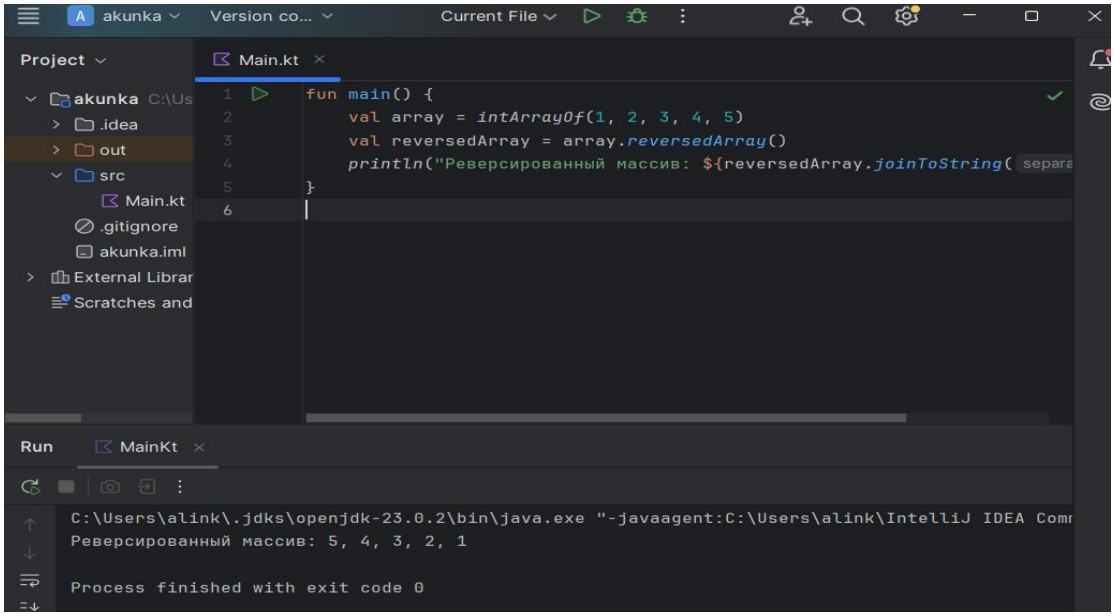
7.fun main() {

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

    val reversedArray = array.reversedArray()

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

}



8.fun main() {

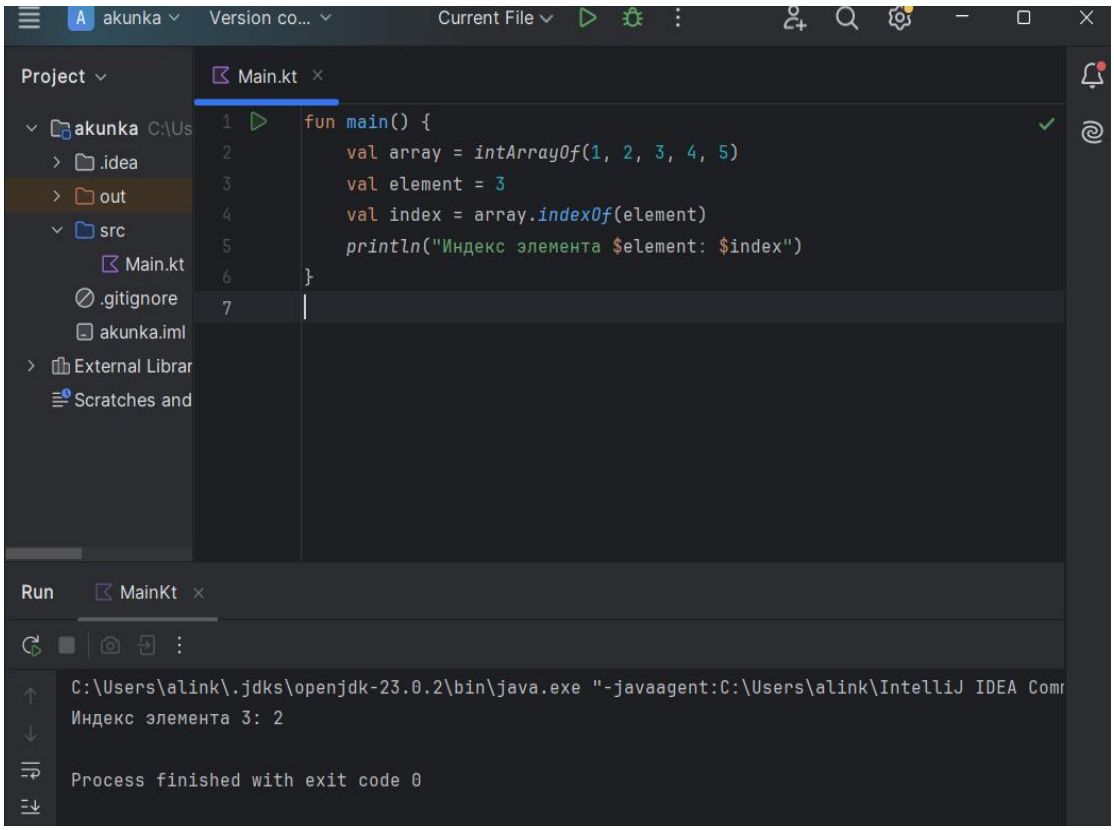
val array = intArrayOf(1, 2, 3, 4, 5)

val element = 3

val index = array.indexOf(element)

println("Индекс элемента $element: $index")

}



9.fun main() {

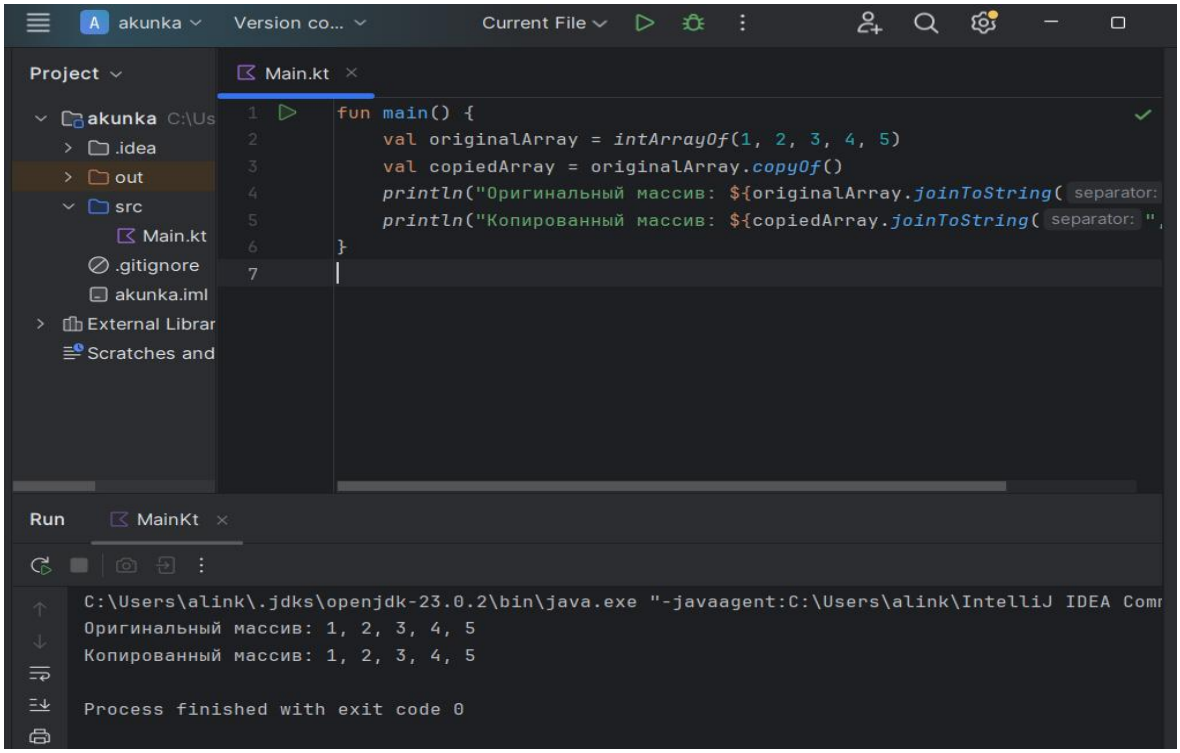
    val originalArray = intArrayOf(1, 2, 3, 4, 5)

    val copiedArray = originalArray.copyOf()

    println("Оригинальный массив: ${originalArray.joinToString(", ")}")

    println("Копированный массив: ${copiedArray.joinToString(", ")}")

}



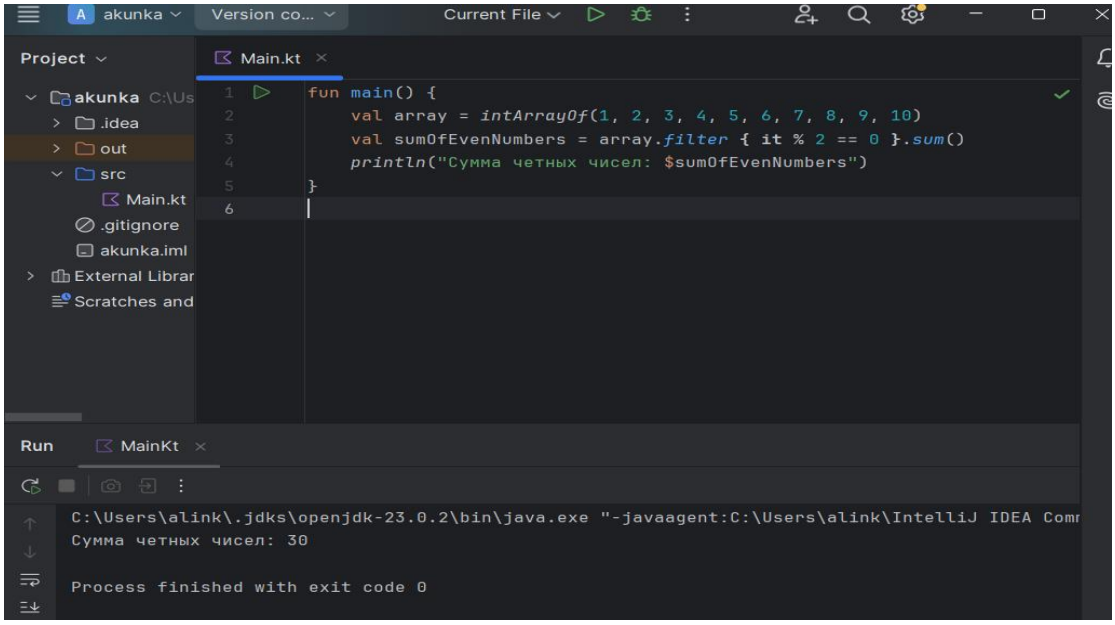
10.fun main() {

    val array = intArrayOf(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

    val sumOfEvenNumbers = array.filter { it % 2 == 0 }.sum()

    println("Сумма четных чисел: $sumOfEvenNumbers")

}



11.fun main() {

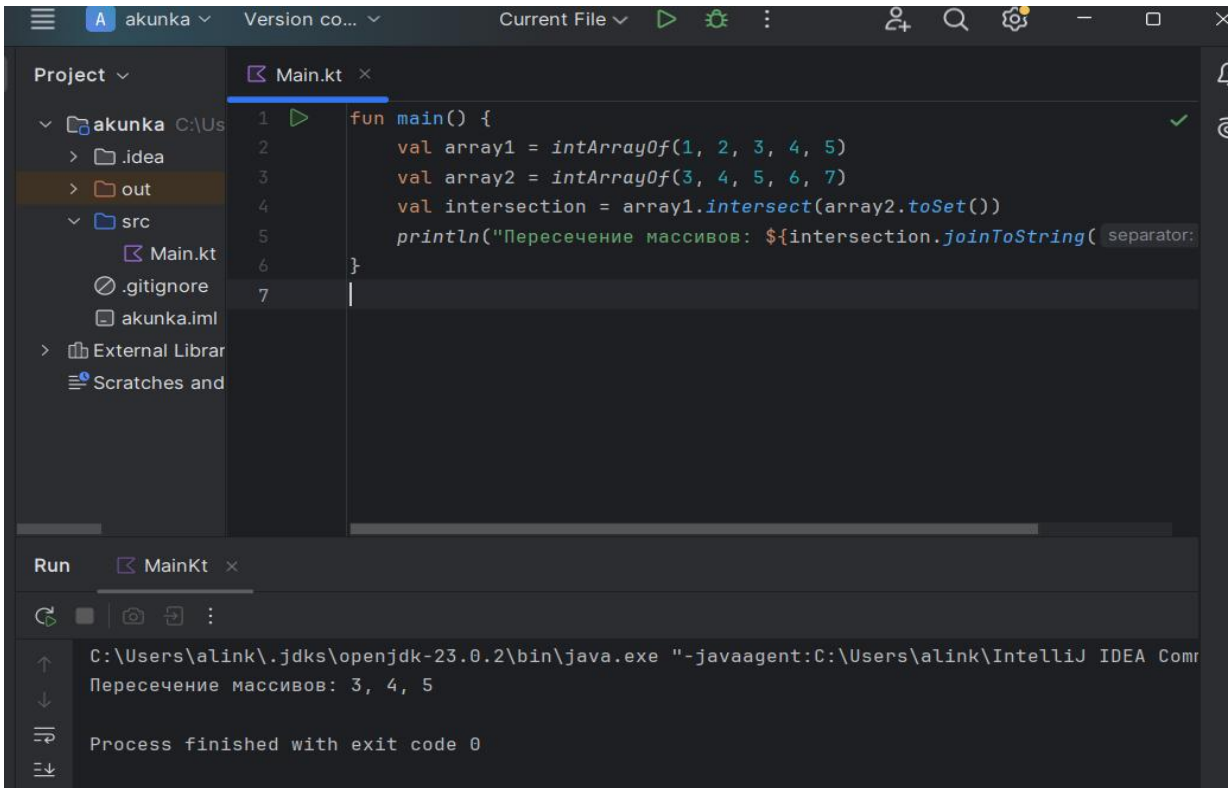
val array1 = intArrayOf(1, 2, 3, 4, 5)

val array2 = intArrayOf(3, 4, 5, 6, 7)

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

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

}



12.fun main() {

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

    val index1 = 0

    val index2 = 2

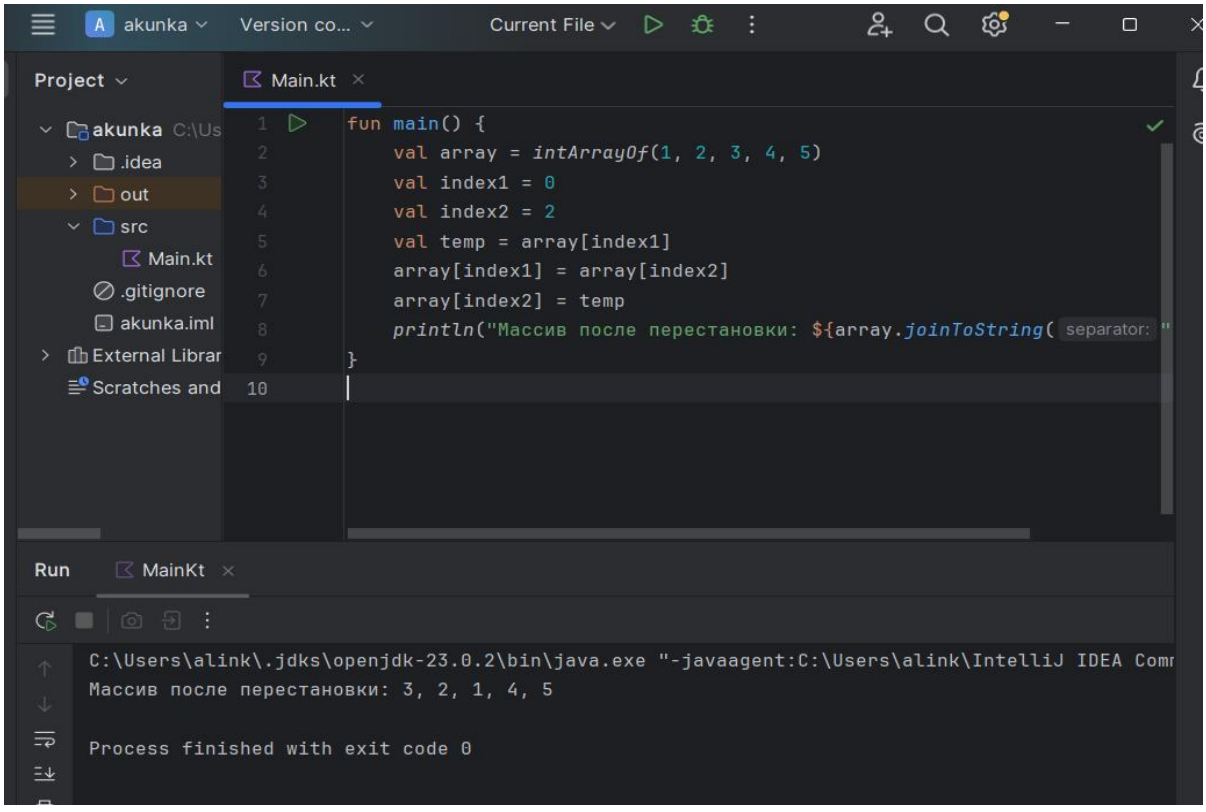
    val temp = array[index1]

    array[index1] = array[index2]

    array[index2] = temp

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

}

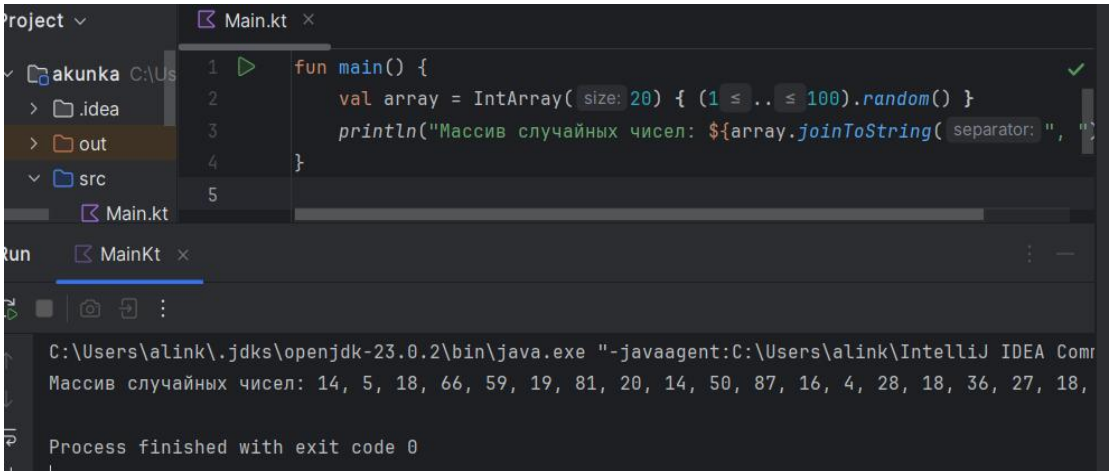


13.fun main() {

    val array = IntArray(20) { (1..100).random() }

    println("Массив случайных чисел: ${array.joinToString(", ")}")

}



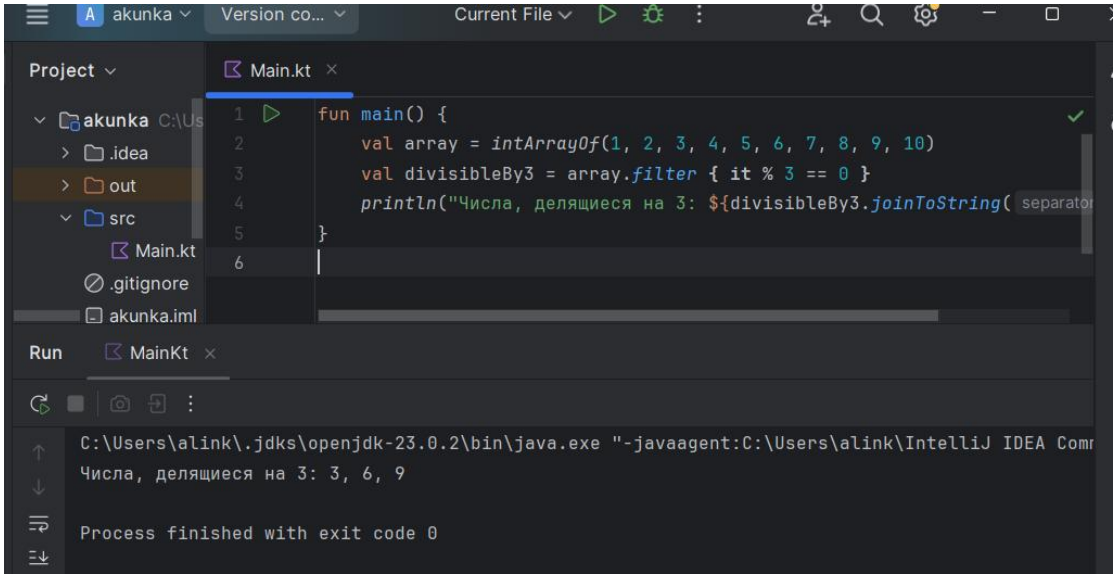
14.fun main() {

val array = intArrayOf(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

val divisibleBy3 = array.filter { it % 3 == 0 }

println("Числа, делящиеся на 3: ${divisibleBy3.joinToString(", ")}")

}



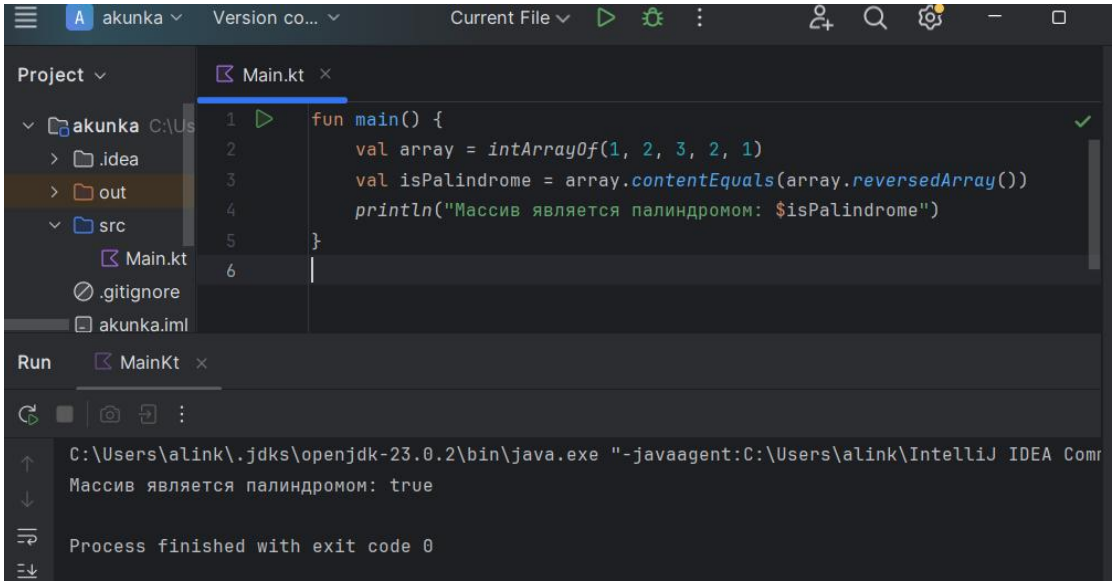
15.fun main() {

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

val isPalindrome = array.contentEquals(array.reversedArray())

println("Массив является палиндромом: $isPalindrome")

}



16.fun main() {

val array1 = intArrayOf(1, 2, 3)

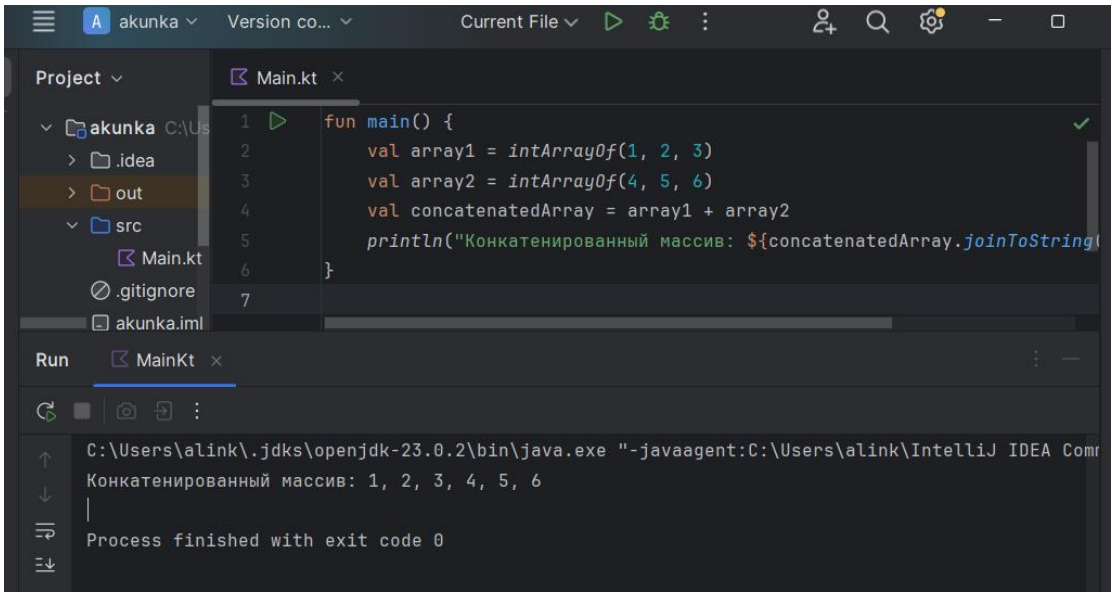
val array2 = intArrayOf(4, 5, 6)

val concatenatedArray = array1 + array2

println("Конкатенированный массив:

${concatenatedArray.joinToString(", ")}")

}

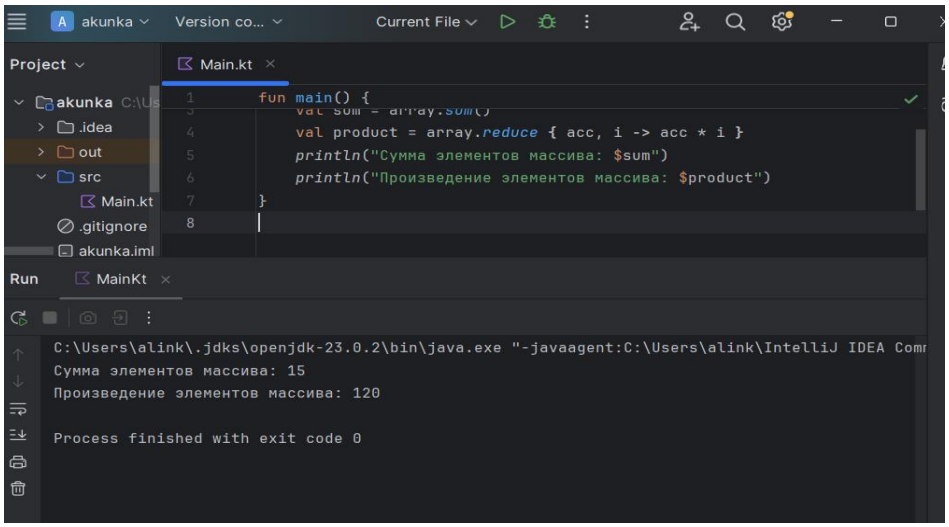


17.fun main() {

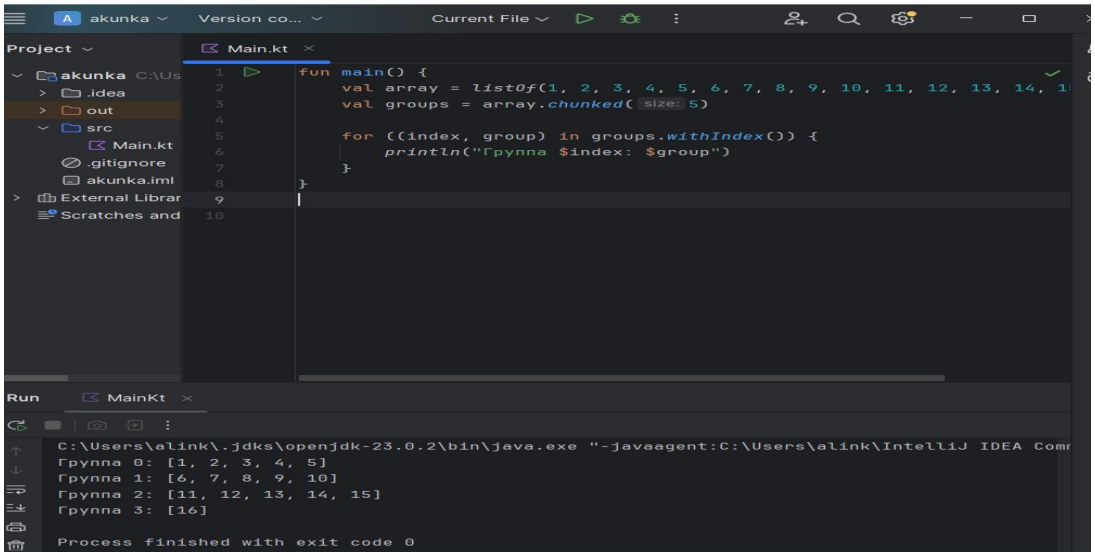
val array = intArrayOf(1, 2, 3, 4, 5)

val sum = array.sum()

```
    val product = array.reduce { acc, i -> acc * i }
    println("Сумма элементов массива: $sum")
    println("Произведение элементов массива: $product")
}
```

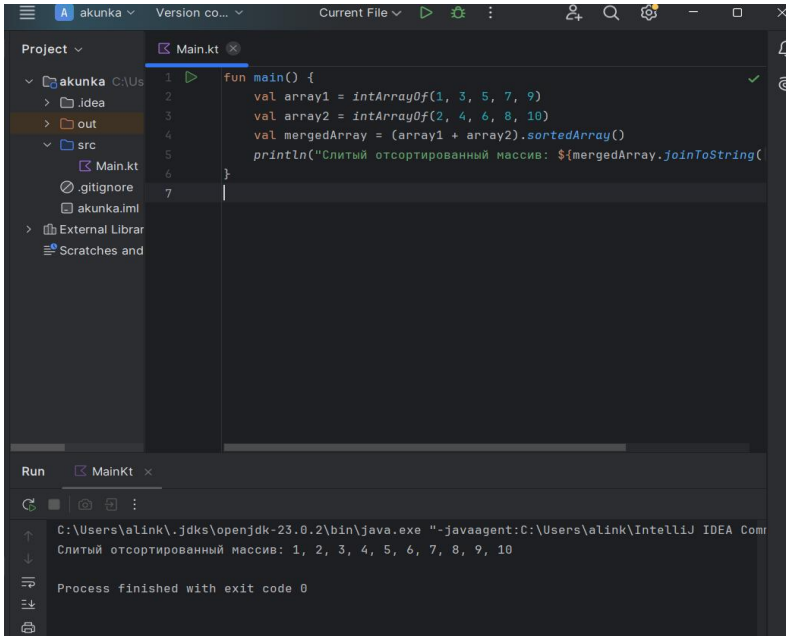



```
18.fun main() {
    val array = listOf(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16)
    val groups = array.chunked(5)
    for ((index, group) in groups.withIndex()) {
        println("Группа $index: $group")
    }
}
```

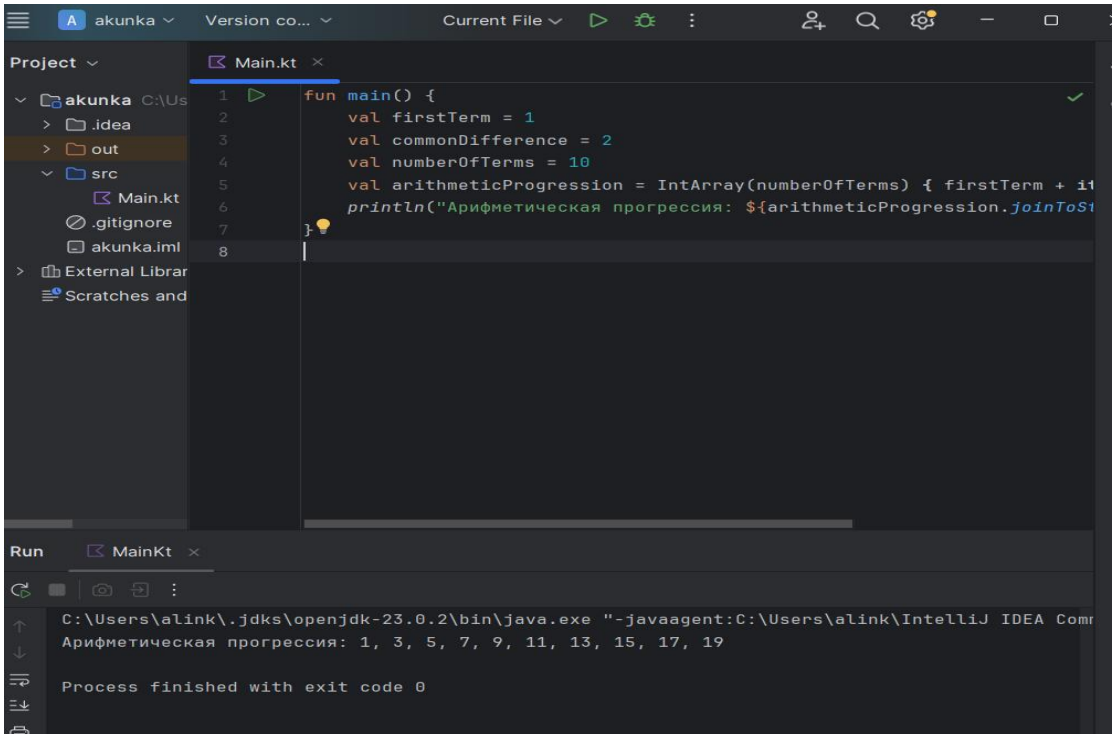



```
19.fun main() {
```

```
    val array1 = intArrayOf(1, 3, 5, 7, 9)
    val array2 = intArrayOf(2, 4, 6, 8, 10)
    val mergedArray = (array1 + array2).sortedArray()
    println("Слитый отсортированный массив:
${mergedArray.joinToString(", ")}")
}
```




```
20.fun main() {
    val firstTerm = 1
    val commonDifference = 2
    val numberOfTerms = 10
    val arithmeticProgression = IntArray(numberOfTerms) { firstTerm + it
* commonDifference }
    println("Арифметическая прогрессия:
${arithmeticProgression.joinToString(", ")}")
}
```



21.fun main() {

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

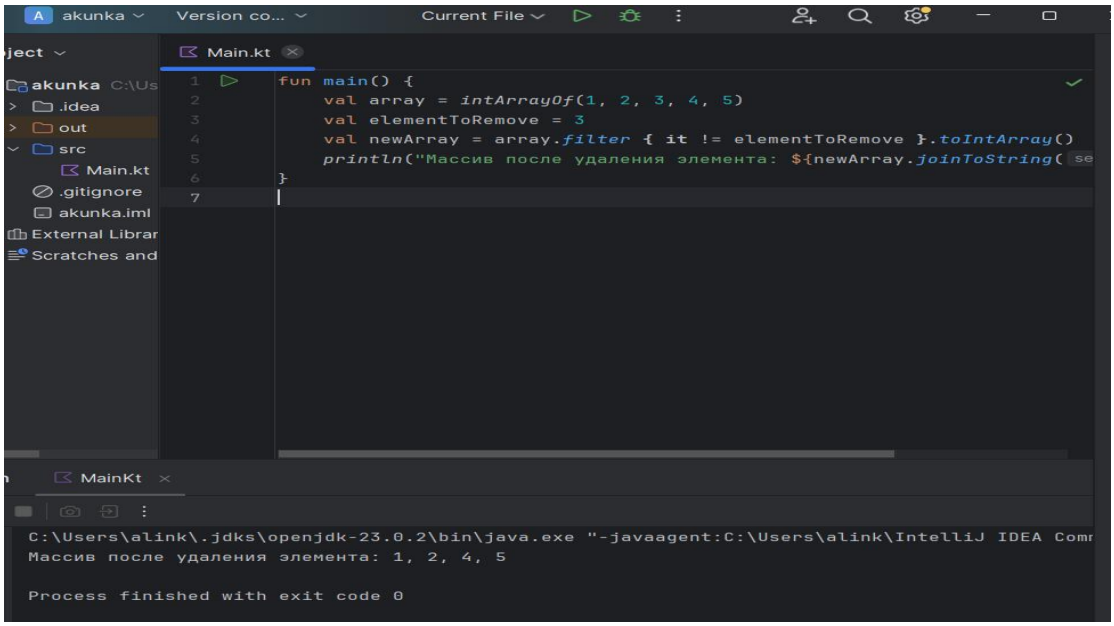
val elementToRemove = 3

val newArray = array.filter { it != elementToRemove }.toIntArray()

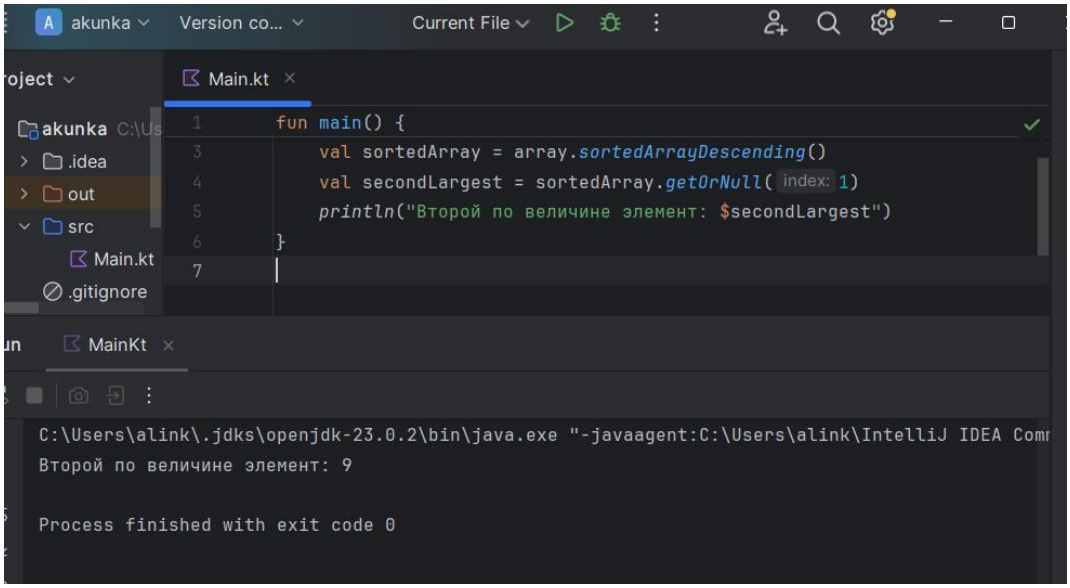
println("Массив после удаления элемента:

${newArray.joinToString(", ")}")

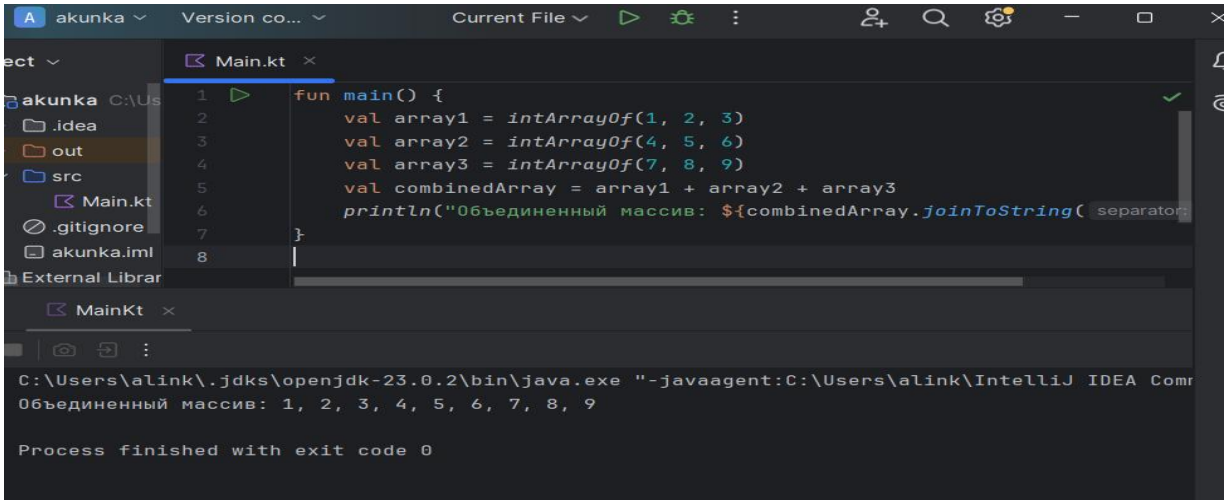
}



22.fun main() {
    val array = intArrayOf(10, 5, 8, 3, 9, 7, 2, 1, 6, 4)
    val sortedArray = array.sortedArrayDescending()
    val secondLargest = sortedArray.getOrNull(1)
    println("Второй по величине элемент: $secondLargest")
}

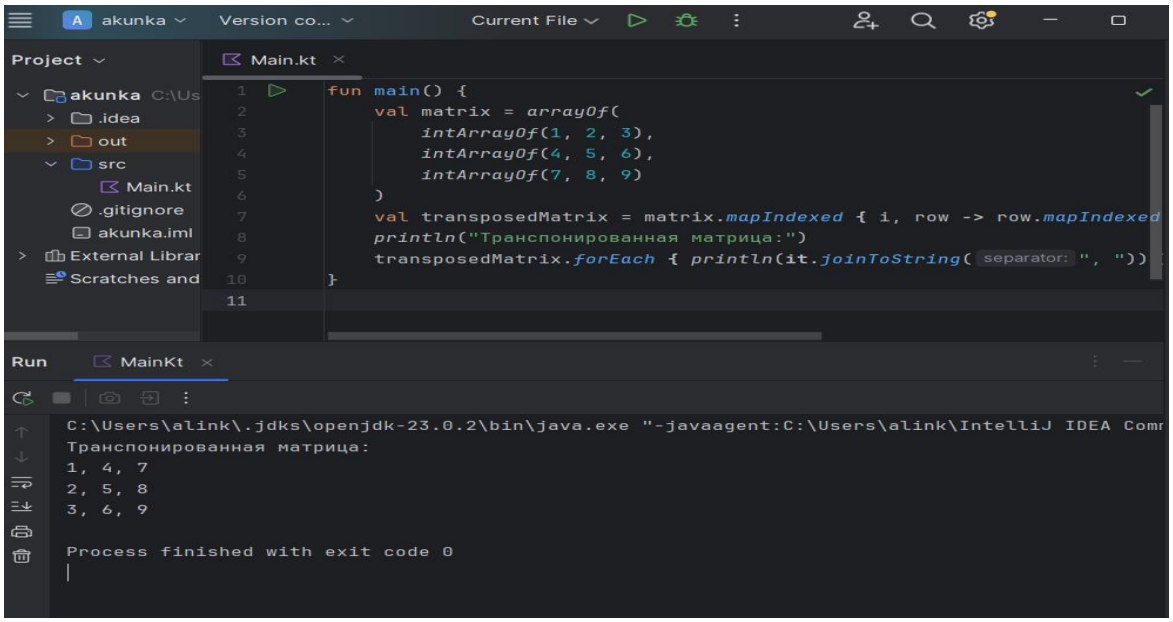


23.fun main() {
    val array1 = intArrayOf(1, 2, 3)
    val array2 = intArrayOf(4, 5, 6)
    val array3 = intArrayOf(7, 8, 9)
    val combinedArray = array1 + array2 + array3
    println("Объединенный массив: ${combinedArray.joinToString(", ")}")
}

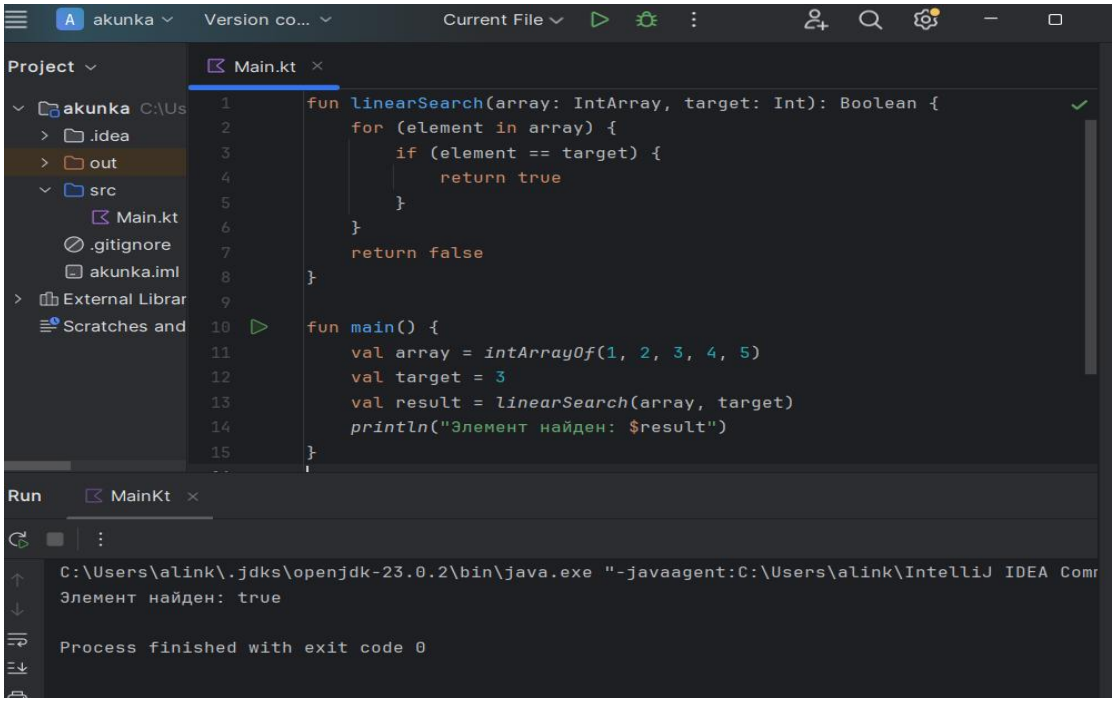


24.fun main() {

```
    val matrix = arrayOf(
        intArrayOf(1, 2, 3),
        intArrayOf(4, 5, 6),
        intArrayOf(7, 8, 9)
    )
    val transposedMatrix = matrix.mapIndexed { i, row ->
row.mapIndexed { j, _ -> matrix[j][i] }.toIntArray() }
    println("Транспонированная матрица:")
    transposedMatrix.forEach { println(it.joinToString(", ")) }
}
```

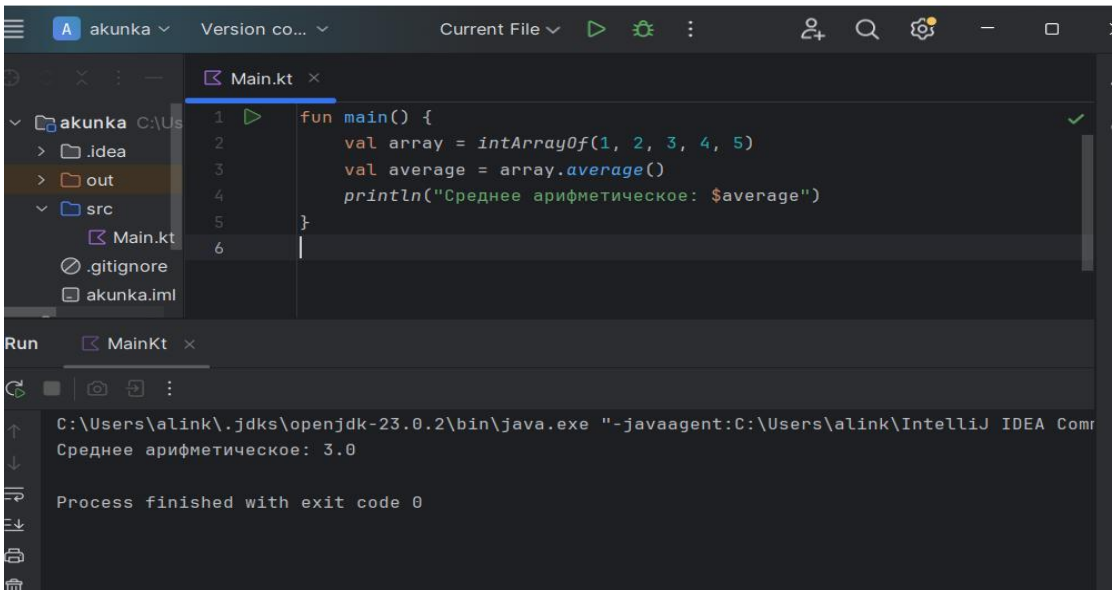


25.fun linearSearch(array: IntArray, target: Int): Boolean {
    for (element in array) {
        if (element == target) {
            return true
        }
    }
    return false
}
fun main() {
    val array = intArrayOf(1, 2, 3, 4, 5)
    val target = 3
    val result = linearSearch(array, target)
    println("Элемент найден: $result")
}



26.fun main() {
    val array = intArrayOf(1, 2, 3, 4, 5)
    val average = array.average()
    println("Среднее арифметическое: $average")

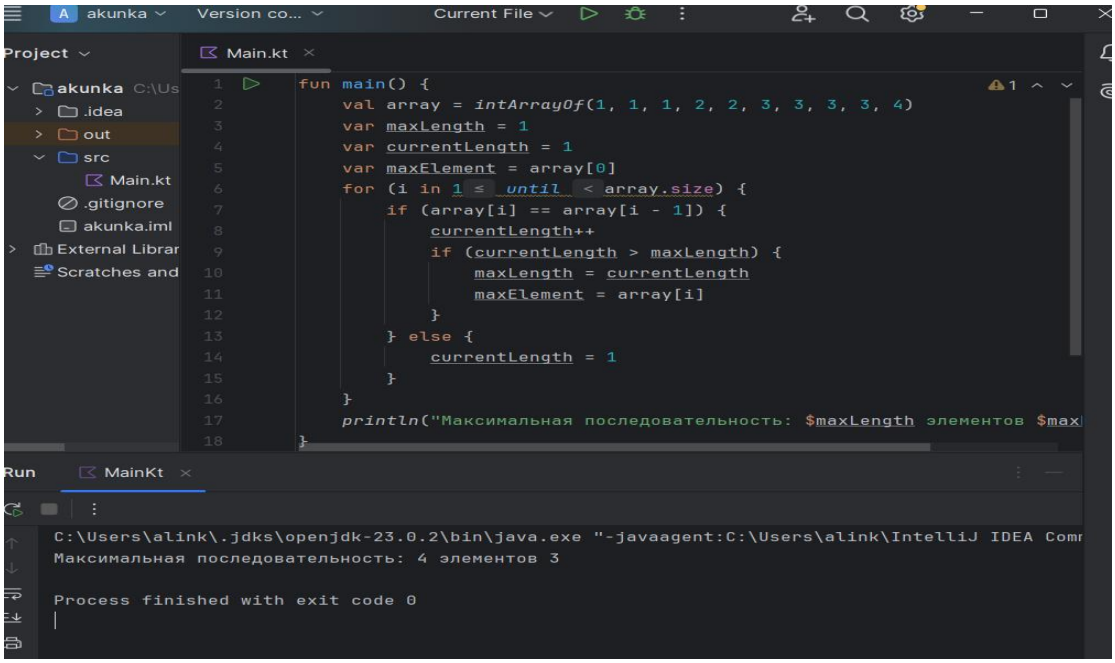
}



27.fun main() {

```
    val array = intArrayOf(1, 1, 1, 2, 2, 3, 3, 3, 3, 4)
    var maxLength = 1
    var currentLength = 1
    var maxElement = array[0]
    for (i in 1 until array.size) {
        if (array[i] == array[i - 1]) {
            currentLength++
            if (currentLength > maxLength) {
                maxLength = currentLength
                maxElement = array[i]
            }
        } else {
            currentLength = 1
        }
    }
    println("Максимальная последовательность: $maxLength
элементов $maxElement")
```

}



28.fun main() {

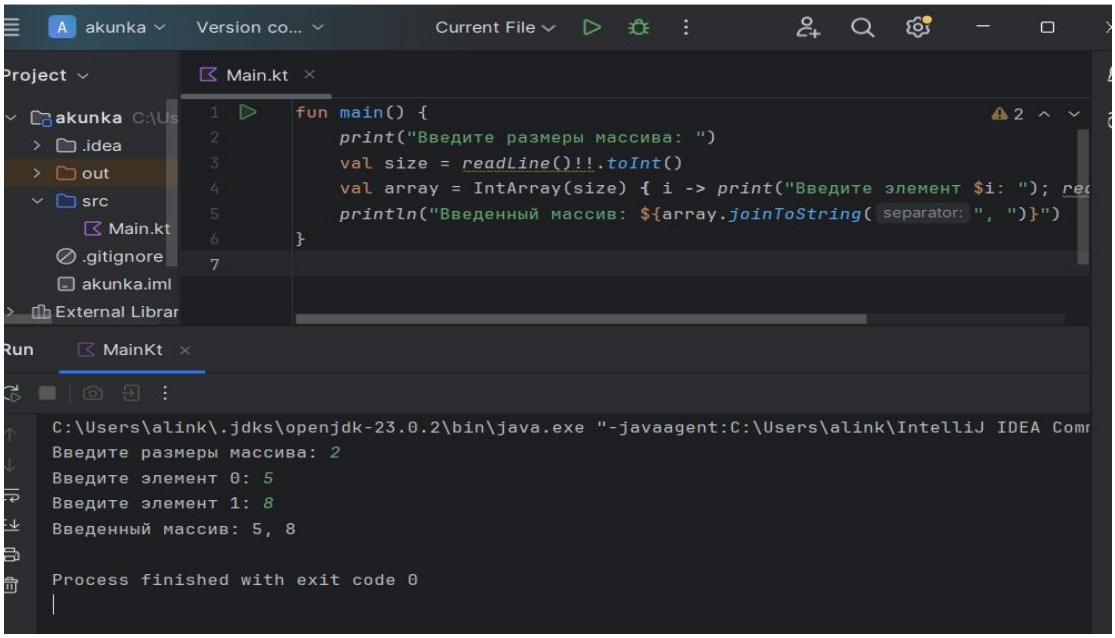
print("Введите размеры массива: ")

val size = readLine()!!.toInt()

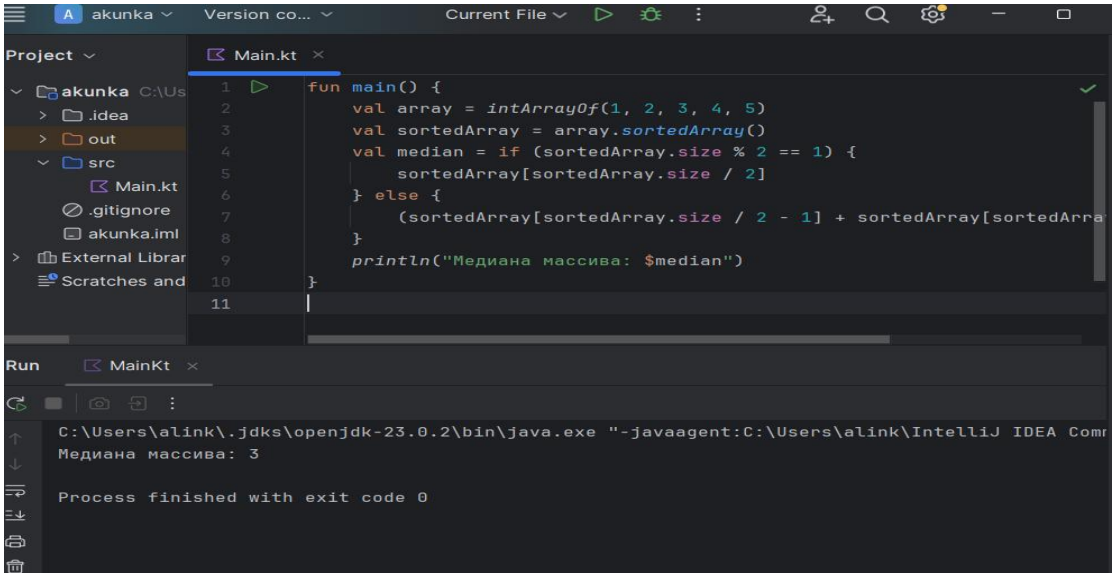
val array = IntArray(size) { i -> print("Введите элемент $i: "); readLine()!!.toInt() }

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

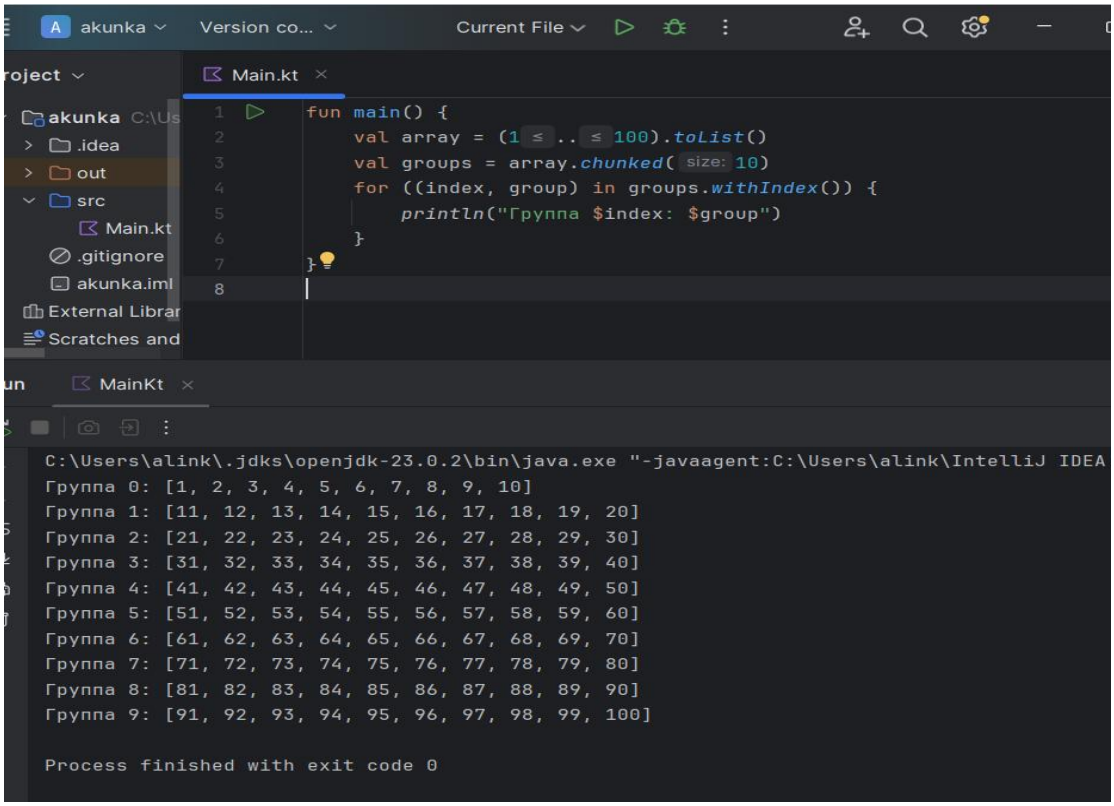
}



29.fun main() {
    val array = intArrayOf(1, 2, 3, 4, 5)
    val sortedArray = array.sortedArray()
    val median = if (sortedArray.size % 2 == 1) {
        sortedArray[sortedArray.size / 2]
    } else {
        (sortedArray[sortedArray.size / 2 - 1] +
sortedArray[sortedArray.size / 2]) / 2.0
    }
    println("Медиана массива: $median")
}



30. fun main() {
    val array = (1..100).toList()
    val groups = array.chunked(10)
    for ((index, group) in groups.withIndex()) {
      println("Группа $index: $group")
  }
}


akunka
Version co...
Current File
Main.kt
fun main() {
val array = (1 ≤ .. ≤ 100).toList()
val groups = array.chunked(size: 10)
for ((index, group) in groups.withIndex()) {
println("Группа $index: $group")
}
}
akunka C:\Us
.idea
out
src
Main.kt
.gitignore
akunka.iml
External Librar
Scratches and
MainKt
C:\Users\alink\.jdks\openjdk-23.0.2\bin\java.exe "-javaagent:C:\Users\alink\IntelliJ IDEA
Группа 0: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Группа 1: [11, 12, 13, 14, 15, 16, 17, 18, 19, 20]
Группа 2: [21, 22, 23, 24, 25, 26, 27, 28, 29, 30]
Группа 3: [31, 32, 33, 34, 35, 36, 37, 38, 39, 40]
Группа 4: [41, 42, 43, 44, 45, 46, 47, 48, 49, 50]
Группа 5: [51, 52, 53, 54, 55, 56, 57, 58, 59, 60]
Группа 6: [61, 62, 63, 64, 65, 66, 67, 68, 69, 70]
Группа 7: [71, 72, 73, 74, 75, 76, 77, 78, 79, 80]
Группа 8: [81, 82, 83, 84, 85, 86, 87, 88, 89, 90]
Группа 9: [91, 92, 93, 94, 95, 96, 97, 98, 99, 100]
Process finished with exit code 0