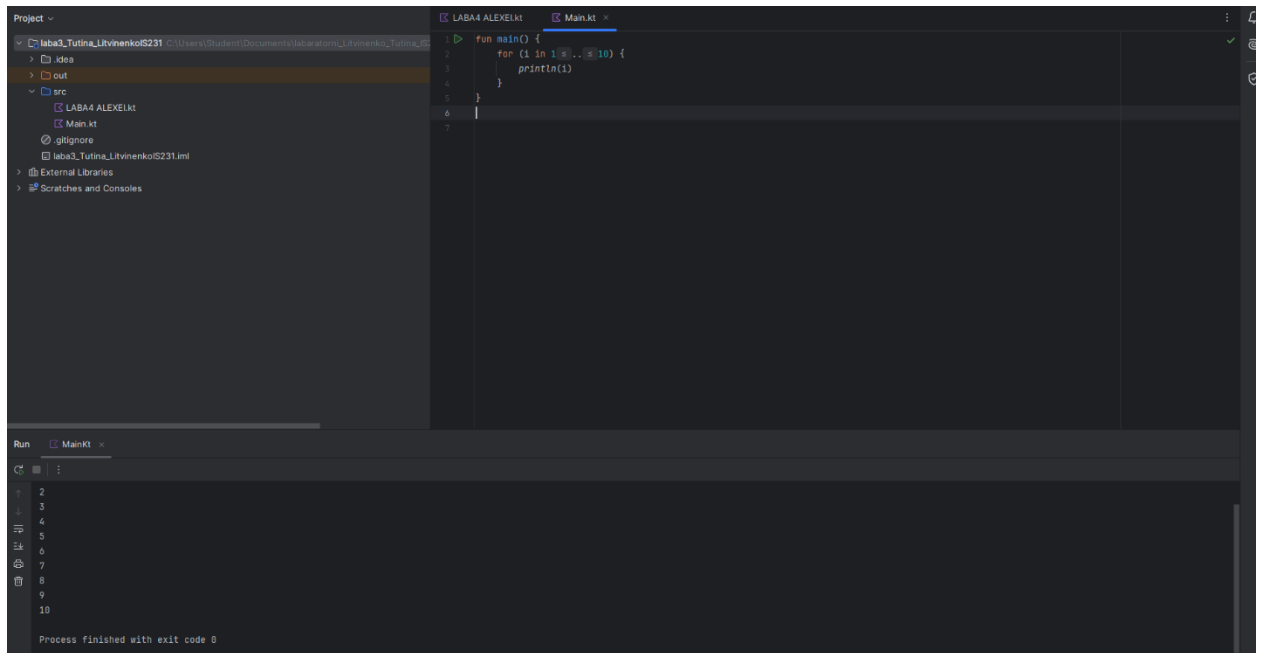


Практическая работа №5

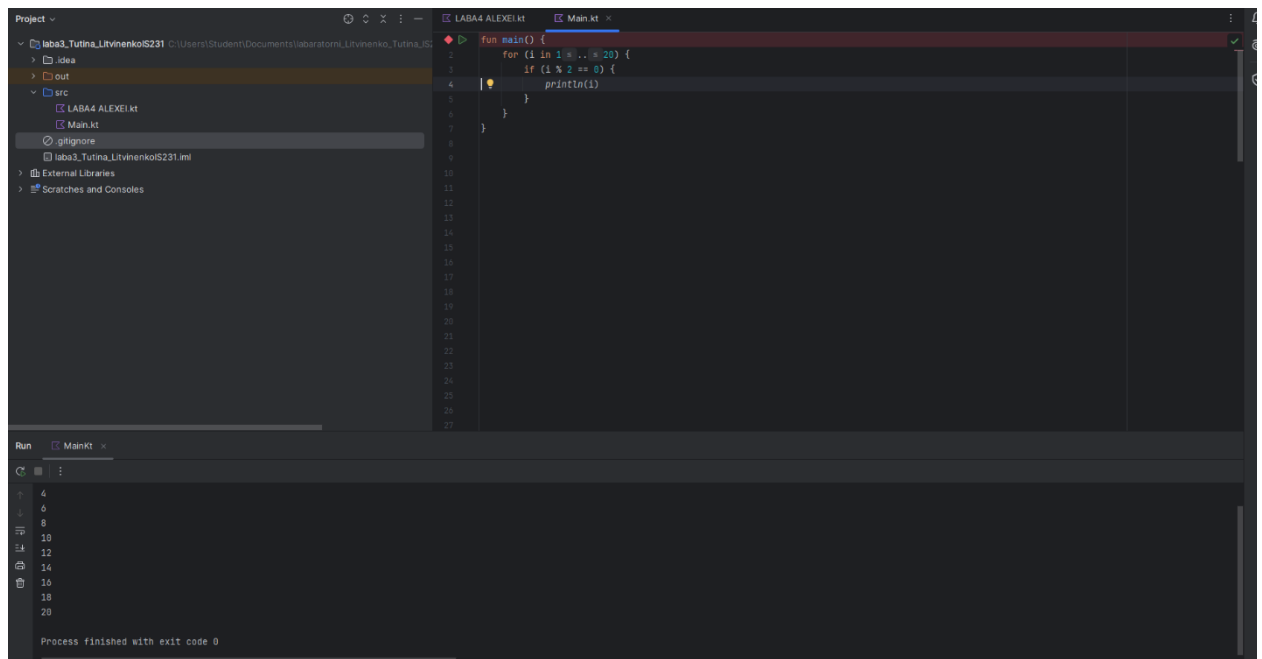
1.

```
fun main() {  
    for (i in 1..10) {  
        println(i)  
    }  
}
```



2.

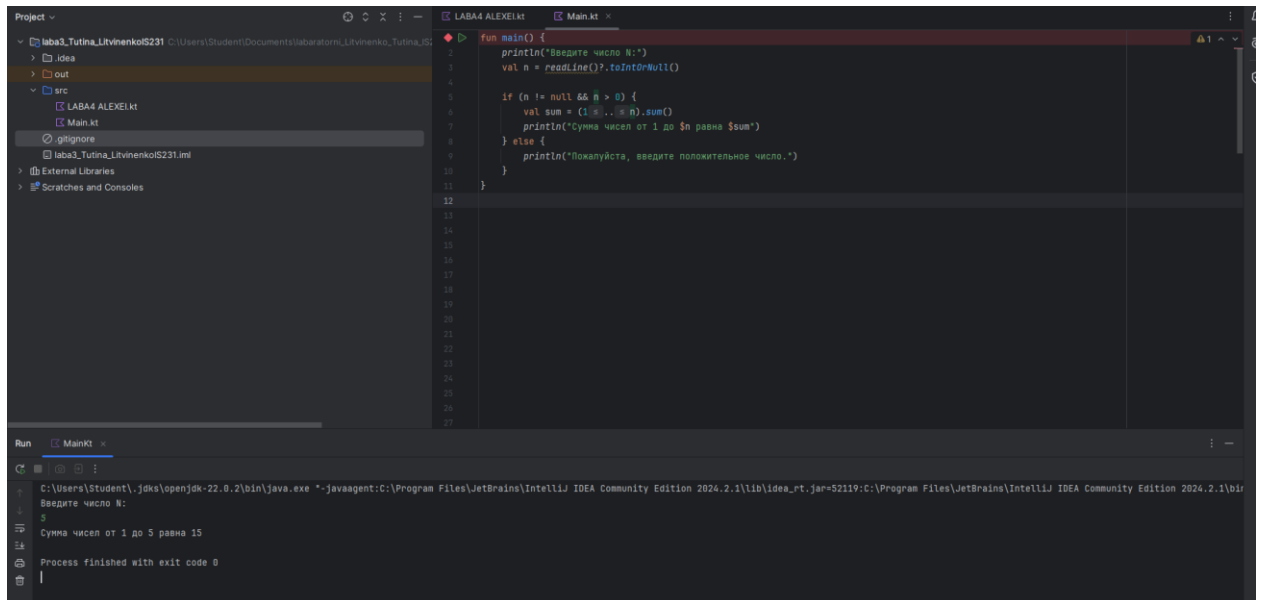
```
fun main() {  
    for (i in 1..20) {  
        if (i % 2 == 0) {  
            println(i)  
        }  
    }  
}
```



3.

```
fun main() {
    println("Введите число N:")
    val n = readLine()?.toIntOrNull()

    if (n != null && n > 0) {
        val sum = (1..n).sum()
        println("Сумма чисел от 1 до $n равна $sum")
    } else {
        println("Пожалуйста, введите положительное число.")
    }
}
```



4.

```
fun factorial(n: Int): Int {
```

```
    return if (n == 0 || n == 1) 1 else n * factorial(n - 1)
```

```
}
```

```
fun main() {
```

```
    println("Введите целое число:")
```

```
    val input = readLine()
```

```
    try {
```

```
        val num = input?.toInt() ?: throw NumberFormatException()
```

```
        if (num < 0) {
```

```
            println("Факториал отрицательного числа не существует.")
```

```
        } else {
```

```
            val result = factorial(num)
```

```
            println("Факториал числа $num равен $result.")
```

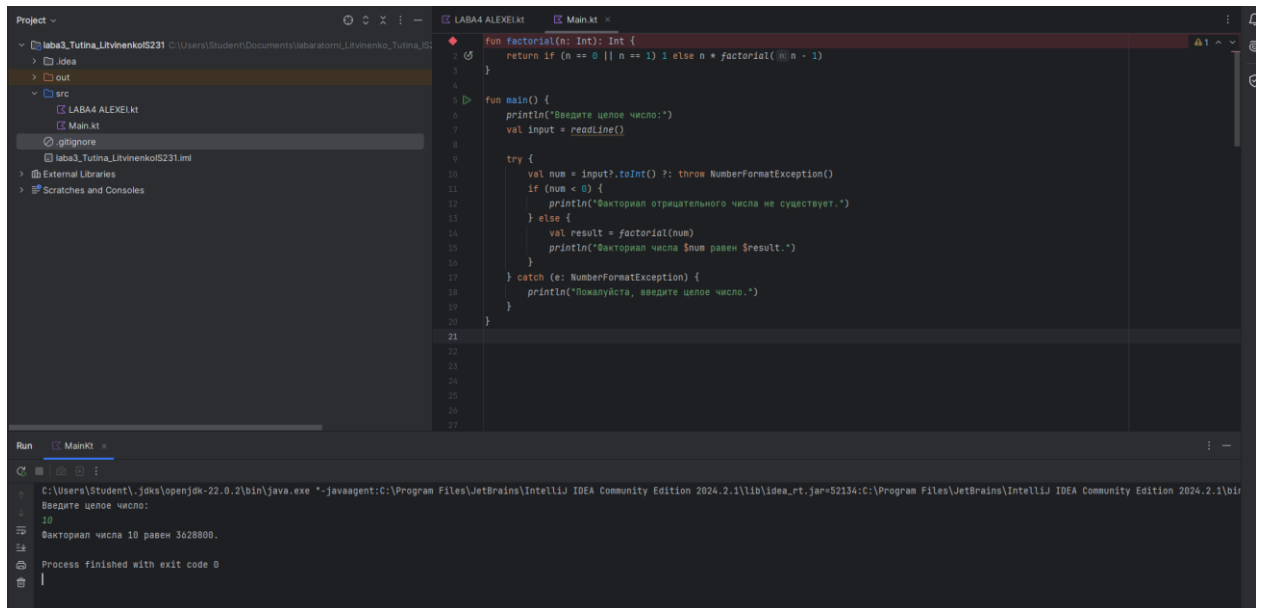
```
        }
```

```
    } catch (e: NumberFormatException) {
```

```
        println("Пожалуйста, введите целое число.")
```

```
    }
```

```
}
```



5.

```

fun isPrime(n: Int): Boolean {
    if (n <= 1) return false
    for (i in 2 until Math.sqrt(n.toDouble()).toInt() + 1) {
        if (n % i == 0) return false
    }
    return true
}

```

```

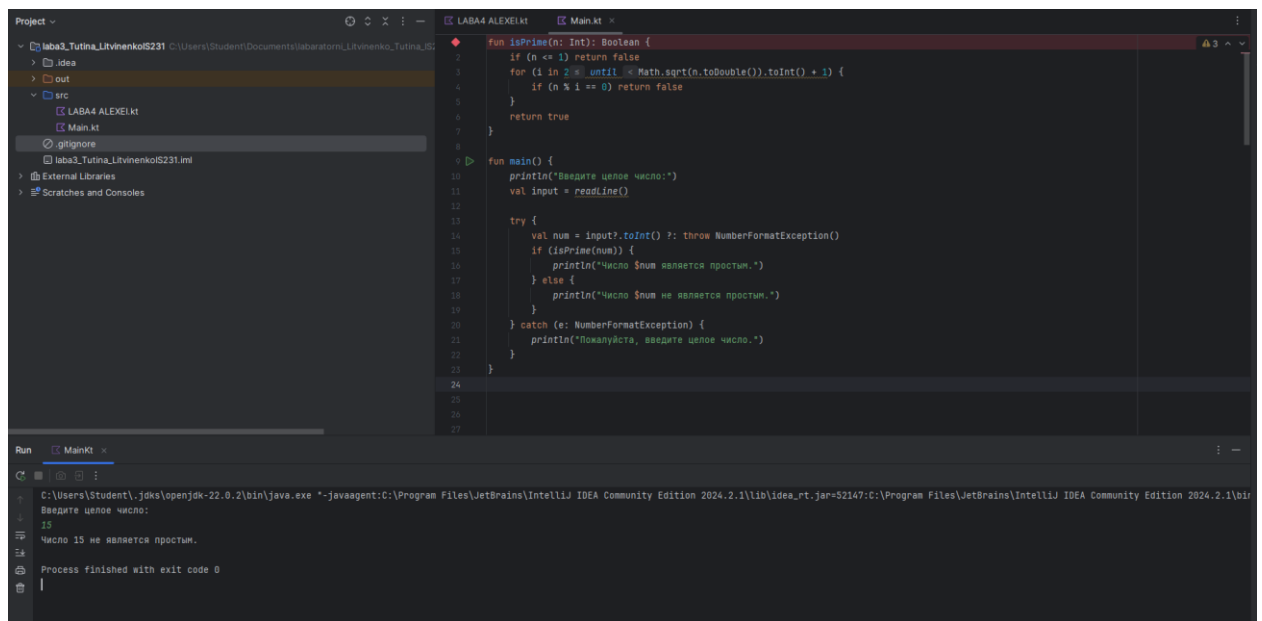
fun main() {
    println("Введите целое число:")
    val input = readLine()

    try {
        val num = input?.toInt() ?: throw NumberFormatException()
        if (isPrime(num)) {
            println("Число $num является простым.")
        } else {
            println("Число $num не является простым.")
        }
    } catch (e: NumberFormatException) {
        println("Пожалуйста, введите целое число.")
    }
}

```

```
}
```

```
}
```



6.

```
fun main() {
```

```
    println("Таблица умножения от 1 до 10:")
```

```
    for (i in 1..10) {
```

```
        for (j in 1..10) {
```

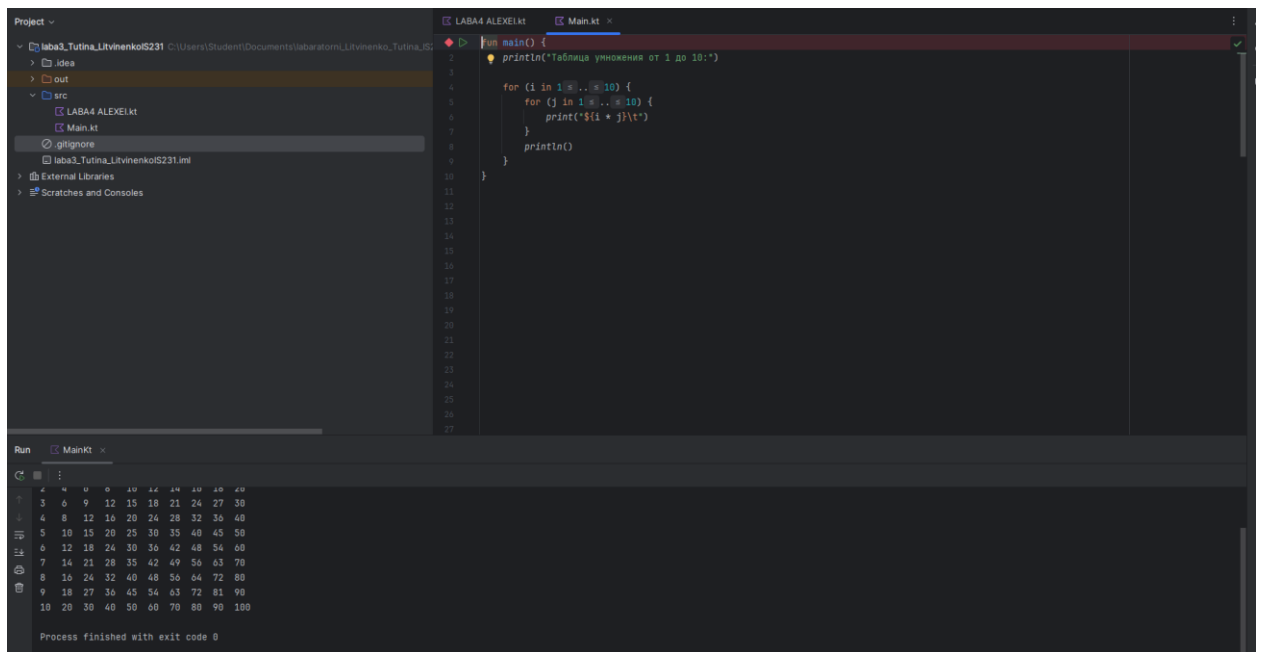
```
            print("${i * j}\t")
```

```
        }
```

```
        println()
```

```
    }
```

```
}
```



7.

```

fun main() {

    println("Введите количество чисел Фибоначчи для генерации:")

    val n = readLine()?.toIntOrNull() ?: return

    val fibonacci = mutableListOf<Int>()

    for (i in 0 until n) {

        if (i == 0) {

            fibonacci.add(0)

        } else if (i == 1) {

            fibonacci.add(1)

        } else {

            fibonacci.add(fibonacci[i - 1] + fibonacci[i - 2])

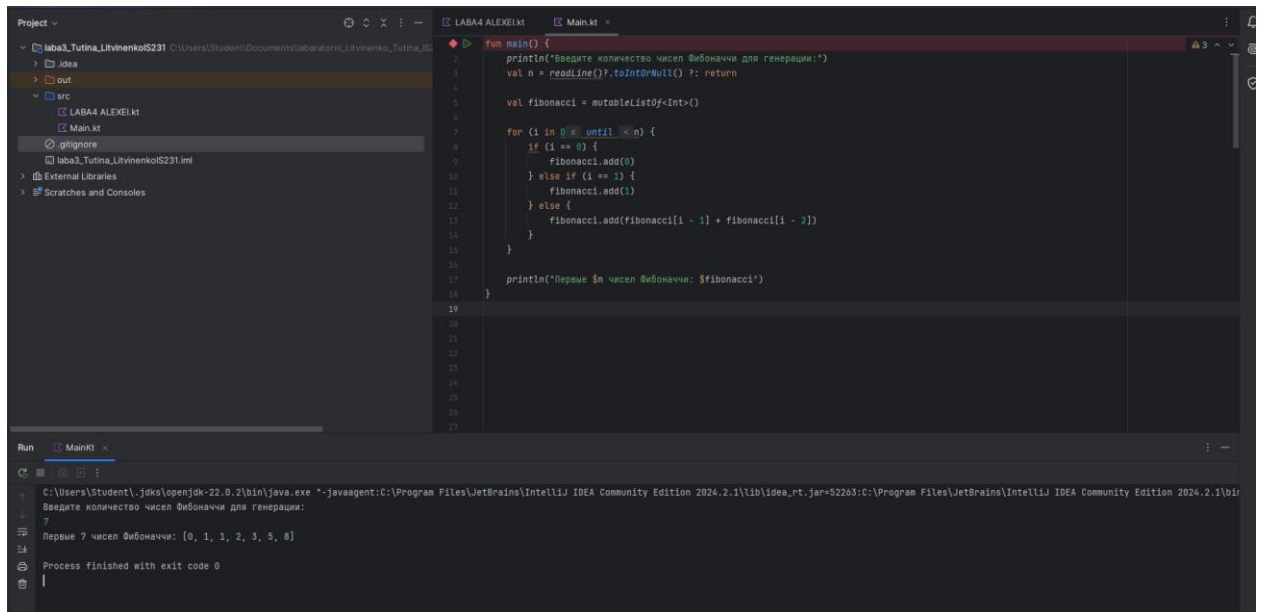
        }

    }

    println("Первые $n чисел Фибоначчи: $fibonacci")

}

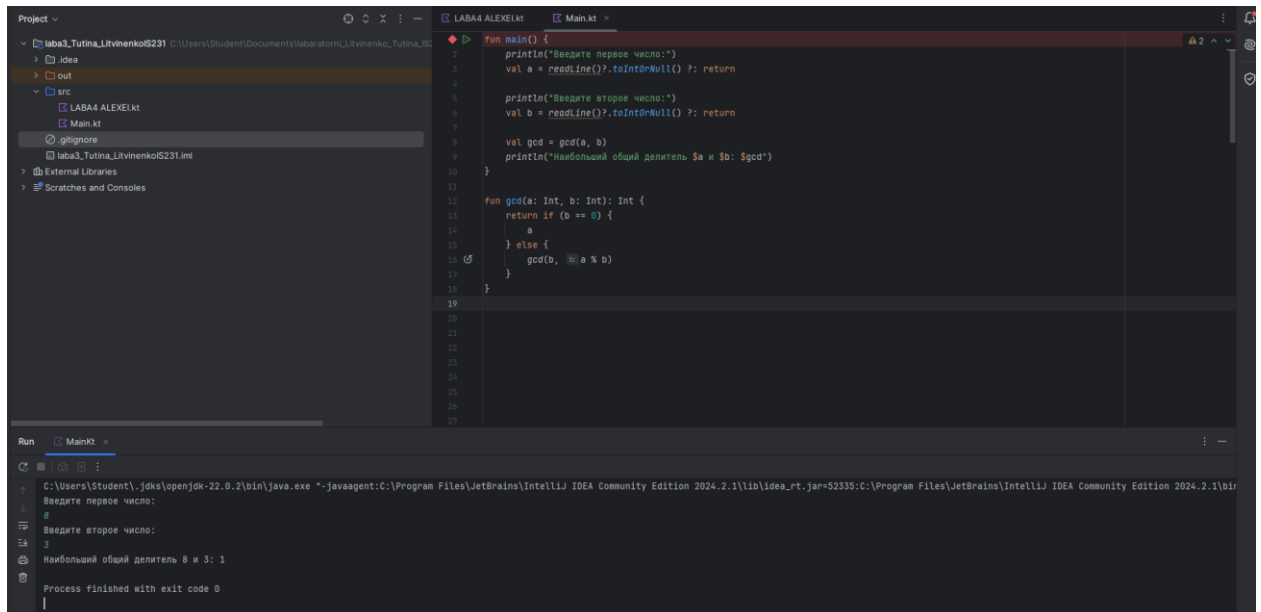
```



8.

```
fun main() {  
    println("Введите первое число:")  
    val a = readLine()?.toIntOrNull() ?: return  
  
    println("Введите второе число:")  
    val b = readLine()?.toIntOrNull() ?: return  
  
    val gcd = gcd(a, b)  
    println("Наибольший общий делитель $a и $b: $gcd")  
}
```

```
fun gcd(a: Int, b: Int): Int {  
    return if (b == 0) {  
        a  
    } else {  
        gcd(b, a % b)  
    }  
}
```



9.

```

fun main() {

    println("Введите строку:")

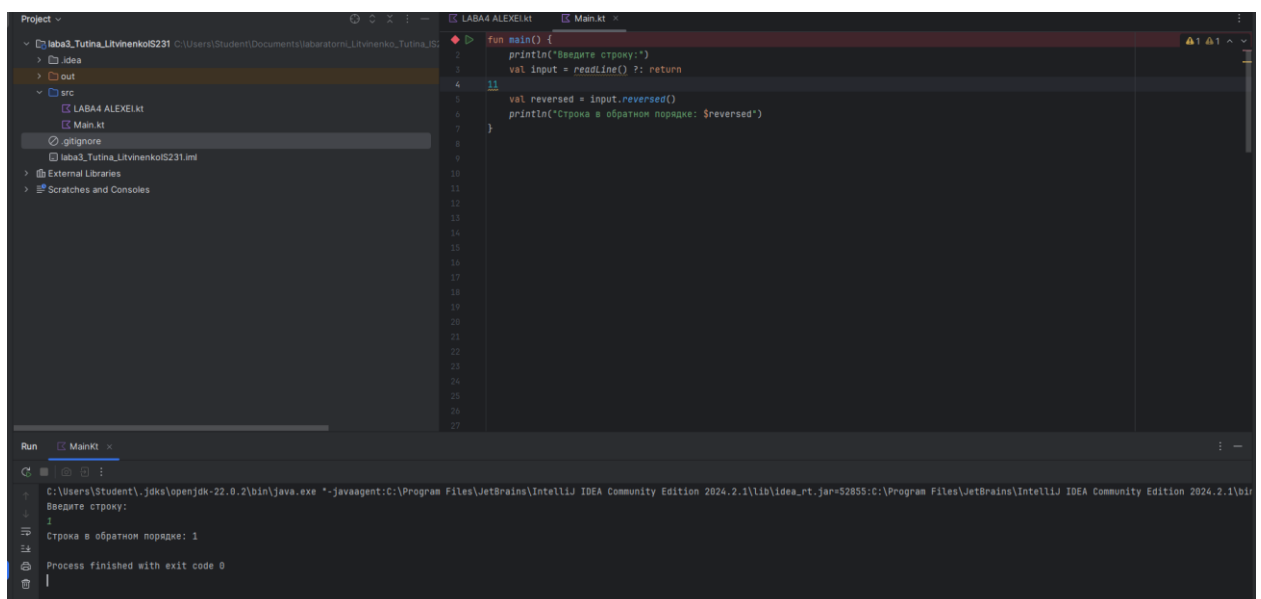
    val input = readLine() ?: return

    val reversed = input.reversed()

    println("Строка в обратном порядке: $reversed")

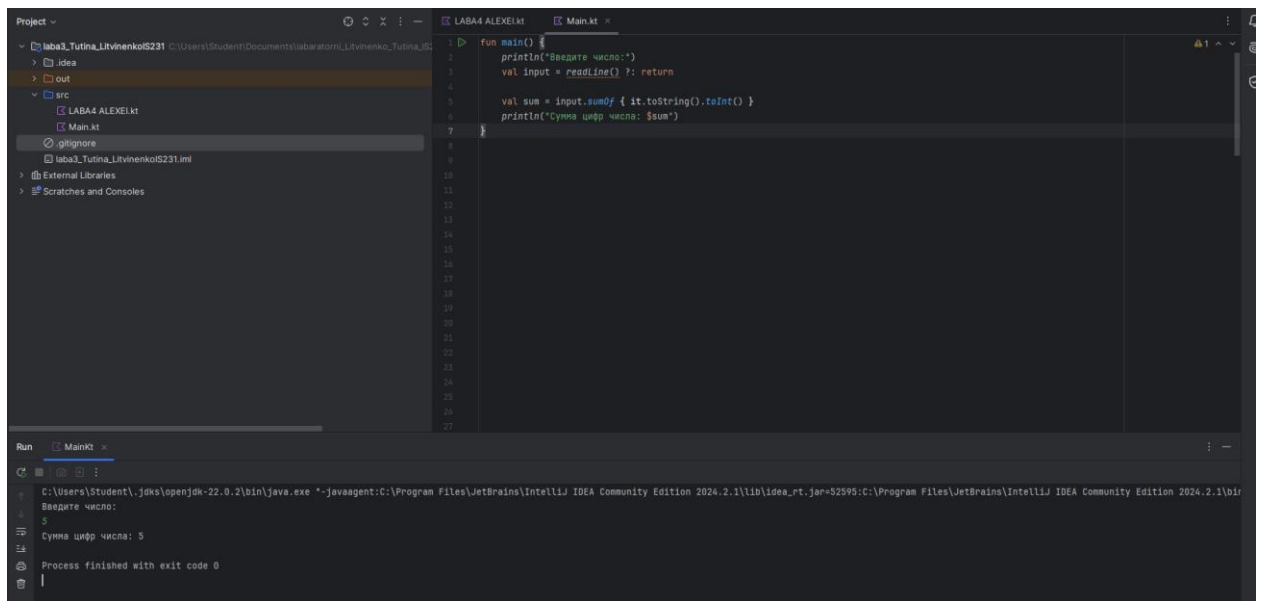
}

```



10.

```
fun main() {  
    println("Введите число:")  
  
    val input = readLine() ?: return  
  
    val sum = input.sumOf { it.toString().toInt() }  
  
    println("Сумма цифр числа: $sum")  
}
```



11.

```
fun main() {  
    println("Введите первую строку:")  
  
    val firstString = readLine()?.replace(" ", "").toLowerCase() ?: ""  
  
    println("Введите вторую строку:")  
  
    val secondString = readLine()?.replace(" ", "").toLowerCase() ?: ""  
  
    val areAnagrams = areAnagrams(firstString, secondString)  
  
    if (areAnagrams) {  
        println("Строки являются анаграммами.")  
    } else {  
        println("Строки не являются анаграммами.")  
    }  
}
```

```

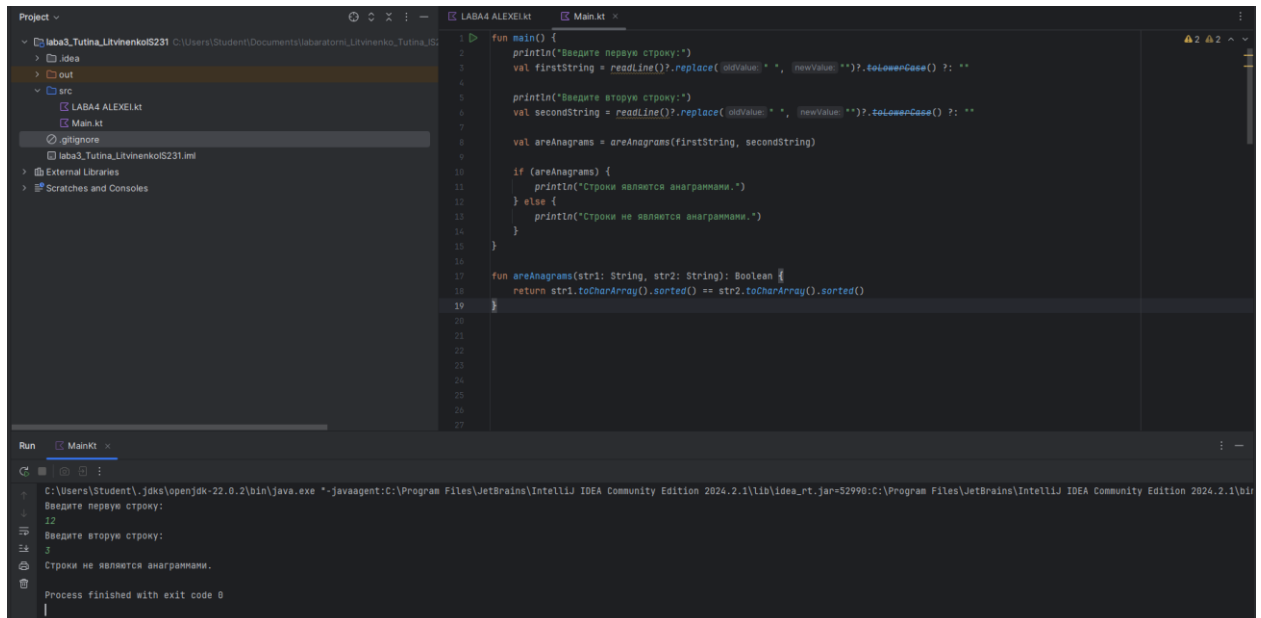
    }
}

```

```

fun areAnagrams(str1: String, str2: String): Boolean {
    return str1.toCharArray().sorted() == str2.toCharArray().sorted()
}

```



12.

```

fun main() {
    println("Введите начальное число:")
    val startInput = readLine()
    val startNumber = startInput?.toIntOrNull()

    println("Введите шаг:")
    val stepInput = readLine()
    val step = stepInput?.toIntOrNull()

    if (startNumber == null || step == null) {
        println("Пожалуйста, введите действительные числа.")
        return
    }

    println("Введите количество чисел в последовательности:")

```

```

val countInput = readLine()

val count = countInput?.toIntOrNull()

if (count == null || count <= 0) {

    println("Пожалуйста, введите положительное число для количества.")

    return

}

println("Сгенерированная числовая последовательность:")

for (i in 0 until count) {

    val number = startNumber + i * step

    print("$number ")

}

println() // для новой строки после последовательности
}

```

The screenshot shows an IDE with a project named 'LABA4_ALEXELK1'. The code in 'Main.kt' is as follows:

```

1 fun main() {
2     println("Введите начальное число:")
3     val startInput = readLine()
4     val startNumber = startInput?.toIntOrNull()
5
6     println("Введите шаг:")
7     val stepInput = readLine()
8     val step = stepInput?.toIntOrNull()
9
10    if (startNumber == null || step == null) {
11        println("Пожалуйста, введите действительные числа.")
12        return
13    }
14
15    println("Введите количество чисел в последовательности:")
16    val countInput = readLine()
17    val count = countInput?.toIntOrNull()
18
19    if (count == null || count <= 0) {
20        println("Пожалуйста, введите положительное число для количества.")
21        return
22    }
23
24    println("Сгенерированная числовая последовательность:")
25    for (i in 0 until count) {
26        val number = startNumber + i * step
27        print("$number ")

```

The Run console shows the following output:

```

Введите начальное число:
0
Введите шаг:
7
Введите количество чисел в последовательности:
4
Сгенерированная числовая последовательность:
0 13 20 27
Process finished with exit code 0

```

13.

```

fun main() {

    println("Таблица квадратов чисел от 1 до 20")

    println("Число\tКвадрат")

    for (i in 1..20) {

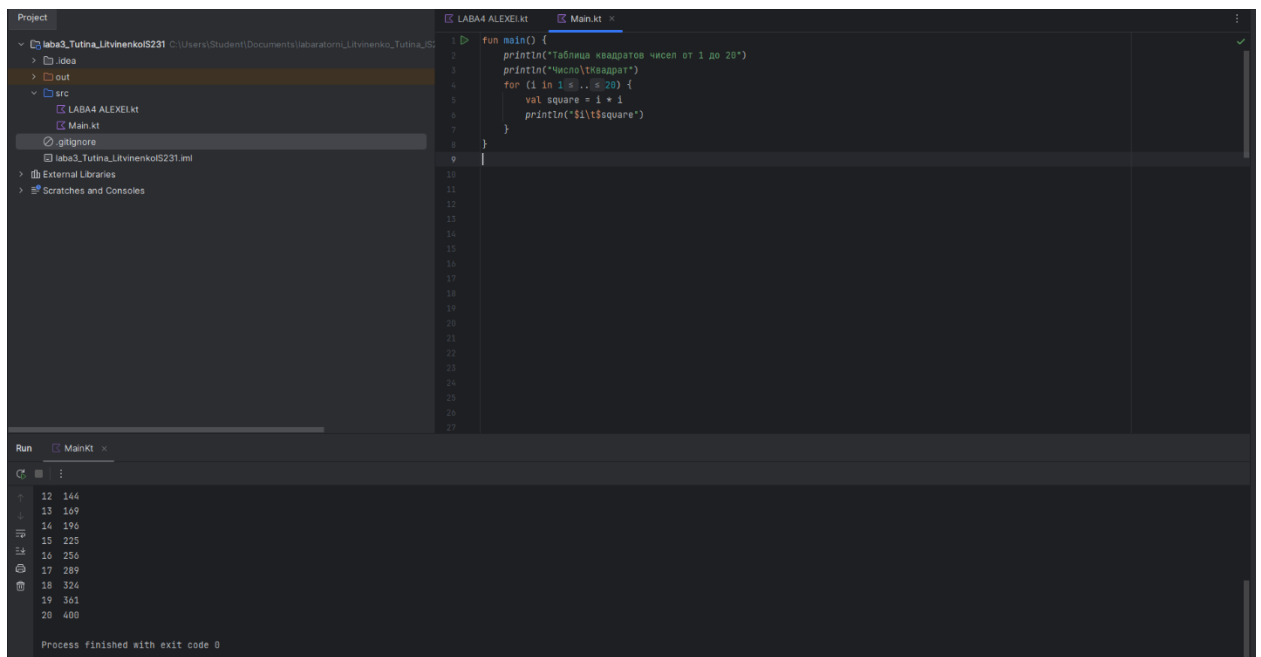
        val square = i * i

```

```

println("${i}\t${square}")
}
}

```



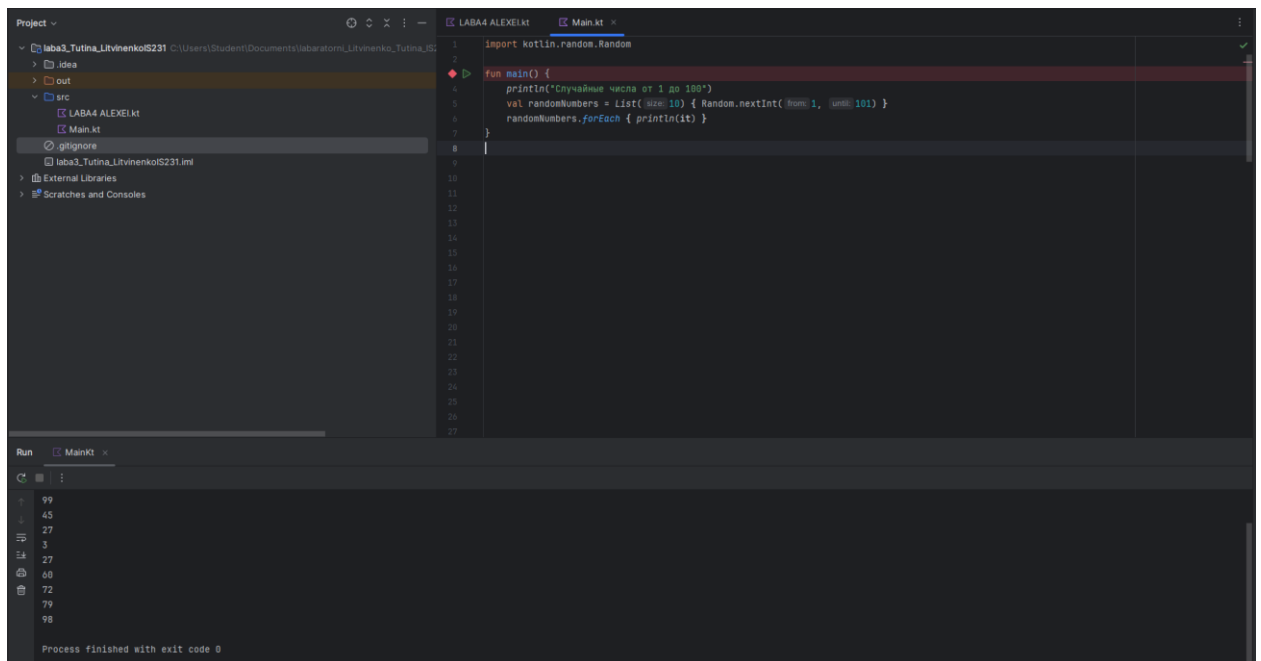
14.

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

```

fun main() {
    println("Случайные числа от 1 до 100")
    val randomNumbers = List(10) { Random.nextInt(1, 101) }
    randomNumbers.forEach { println(it) }
}

```



15.

```
fun main() {

    println("Введите строку для проверки на палиндром:")

    val input = readLine()?.trim()?.lowercase()

    if (input != null) {

        val isPalindrome = input == input.reversed()

        if (isPalindrome) {

            println("Строка является палиндромом.")

        } else {

            println("Строка не является палиндромом.")

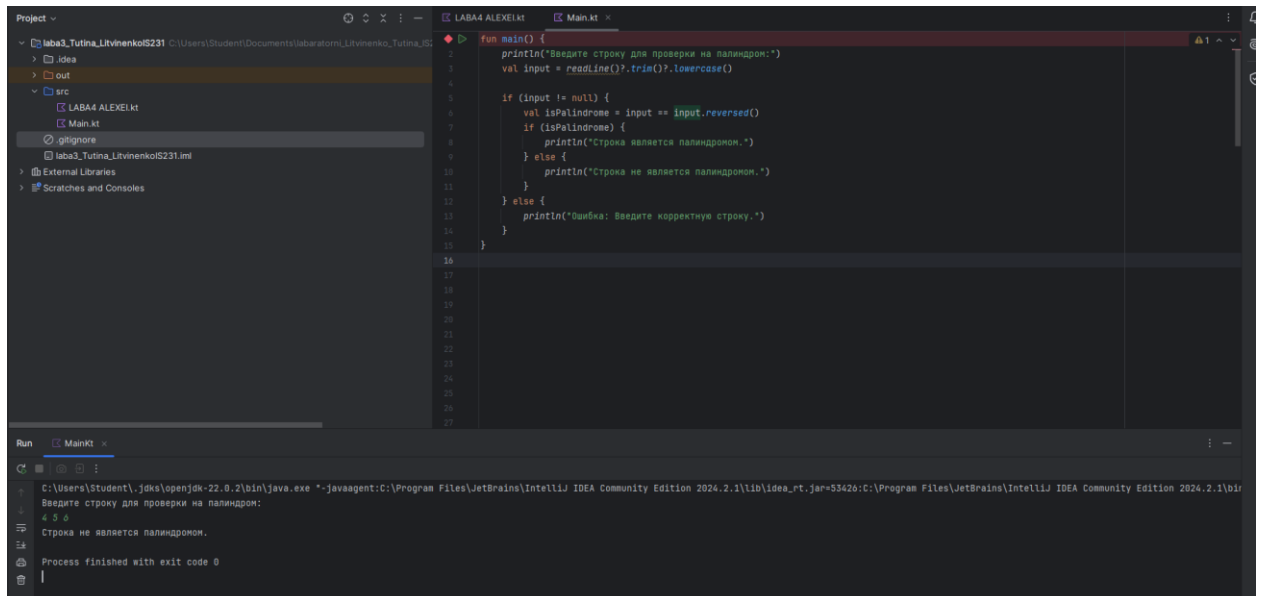
        }

    } else {

        println("Ошибка: Введите корректную строку.")

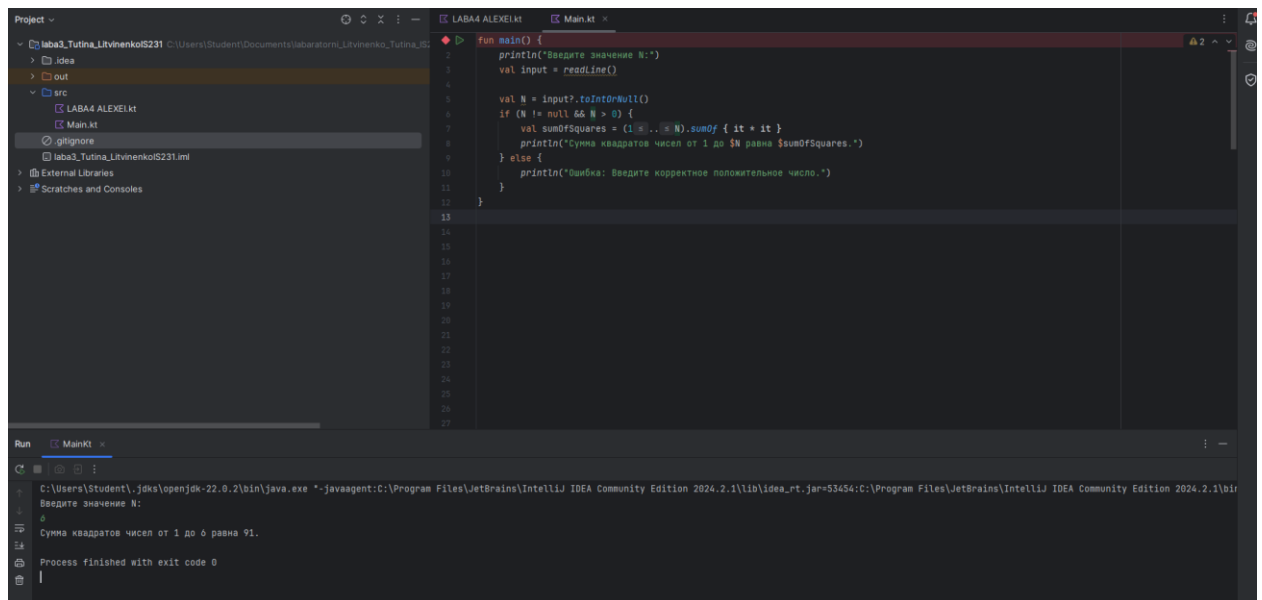
    }

}
```



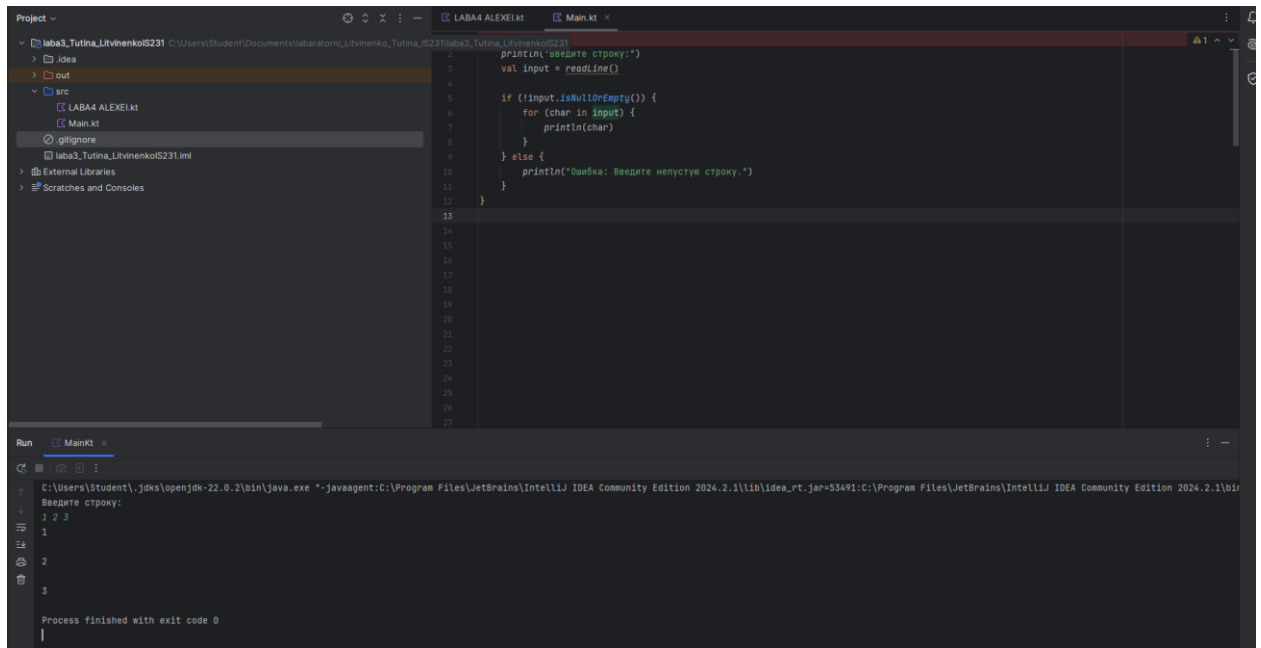
16.

```
fun main() {  
    println("Введите значение N:")  
    val input = readLine()  
  
    val N = input?.toIntOrNull()  
    if (N != null && N > 0) {  
        val sumOfSquares = (1..N).sumOf { it * it }  
        println("Сумма квадратов чисел от 1 до $N равна $sumOfSquares.")  
    } else {  
        println("Ошибка: Введите корректное положительное число.")  
    }  
}
```



17.

```
fun main() {  
    println("Введите строку:")  
    val input = readLine()  
  
    if (!input.isNullOrEmpty()) {  
        for (char in input) {  
            println(char)  
        }  
    } else {  
        println("Ошибка: Введите непустую строку.")  
    }  
}
```



18.

```
fun main() {
```

```
    println("Введите высоту лестницы:")
```

```
    val input = readLine()
```

```
    val height = input?.toIntOrNull()
```

```
    if (height != null && height > 0) {
```

```
        for (i in 1..height) {
```

```
            println("#".repeat(i))
```

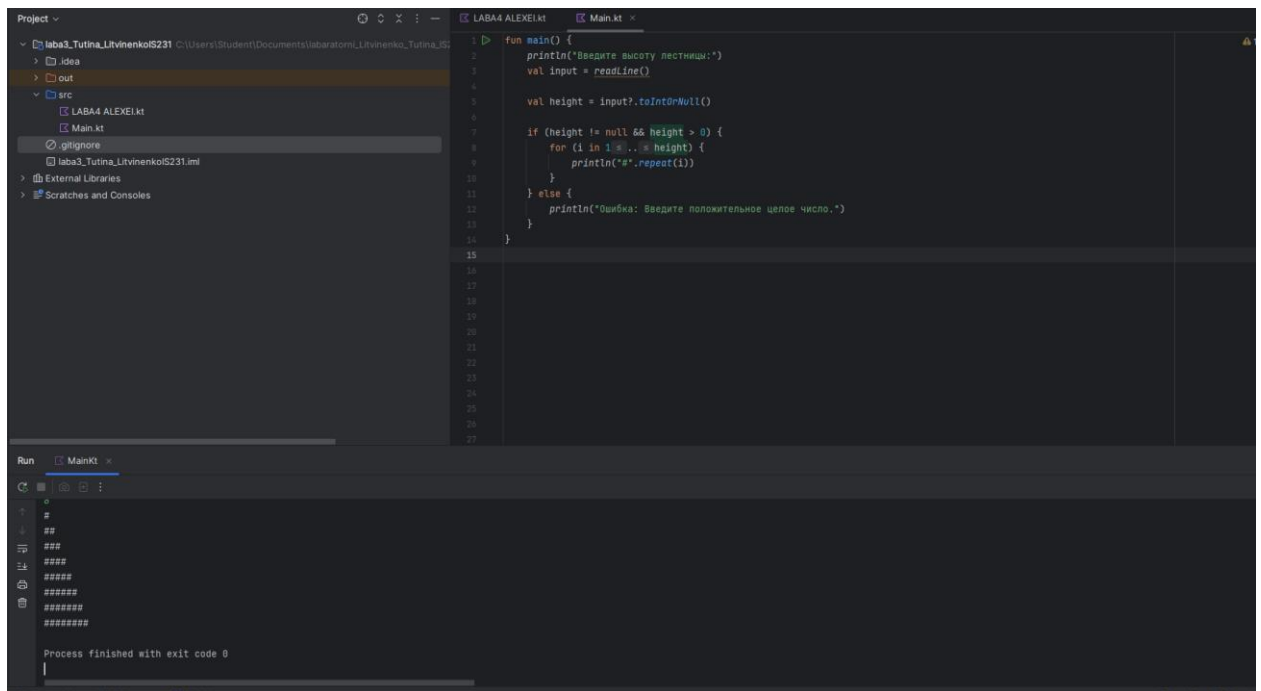
```
        }
```

```
    } else {
```

```
        println("Ошибка: Введите положительное целое число.")
```

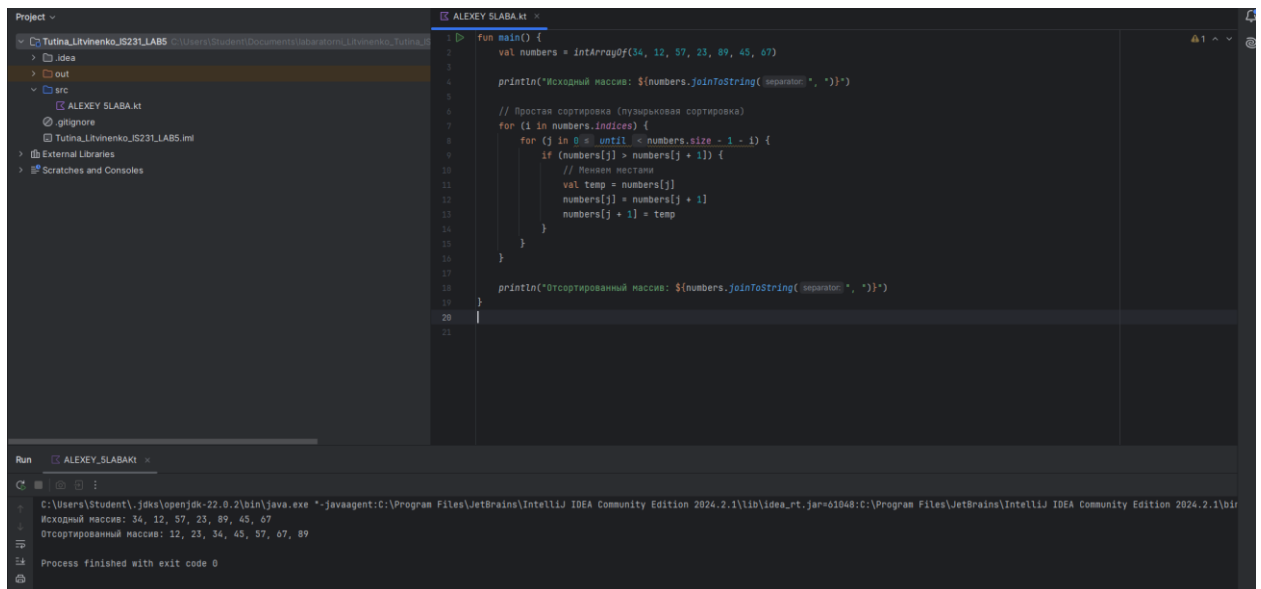
```
    }
```

```
}
```

19.

```
fun main() {  
    val numbers = intArrayOf(34, 12, 57, 23, 89, 45, 67)  
  
    println("Исходный массив: ${numbers.joinToString(", ")}")  
  
    // Простая сортировка (пузырьковая сортировка)  
    for (i in numbers.indices) {  
        for (j in 0 until numbers.size - 1 - i) {  
            if (numbers[j] > numbers[j + 1]) {  
                // Меняем местами  
                val temp = numbers[j]  
                numbers[j] = numbers[j + 1]  
                numbers[j + 1] = temp  
            }  
        }  
    }  
  
    println("Отсортированный массив: ${numbers.joinToString(", ")}")  
}
```



20.

```
fun main() {
```

```
    println("Введите начало диапазона:")
```

```
    val start = readLine()?.toIntOrNull() ?: return
```

```
    println("Введите конец диапазона:")
```

```
    val end = readLine()?.toIntOrNull() ?: return
```

```
    println("Простые числа в диапазоне от $start до $end:")
```

```
    for (num in start..end) {
```

```
        if (isPrime(num)) {
```

```
            println(num)
```

```
        }
```

```
    }
```

```
}
```

```
fun isPrime(number: Int): Boolean {
```

```
    if (number < 2) return false
```

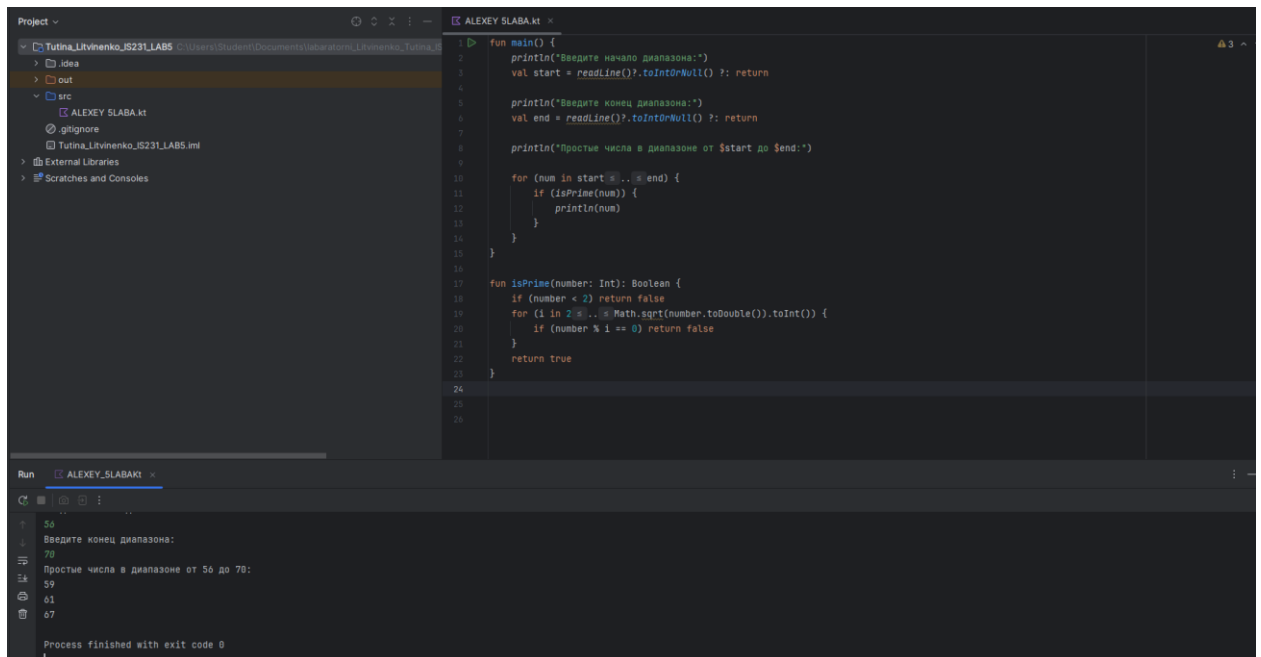
```
    for (i in 2..Math.sqrt(number.toDouble()).toInt()) {
```

```
        if (number % i == 0) return false
```

```
    }
```

```
return true
```

```
}
```



The screenshot shows an IDE with a project named 'Tutina_Litvinenko_JS231_LAB5'. The source file 'ALEXEY_SLABA.kt' contains the following Kotlin code:

```
1 fun main() {
2     println("Введите начало диапазона:")
3     val start = readLine()?.toIntOrNull() ?: return
4
5     println("Введите конец диапазона:")
6     val end = readLine()?.toIntOrNull() ?: return
7
8     println("Простые числа в диапазоне от $start до $end:")
9
10    for (num in start..end) {
11        if (isPrime(num)) {
12            println(num)
13        }
14    }
15 }
16
17 fun isPrime(number: Int): Boolean {
18     if (number < 2) return false
19     for (i in 2..Math.sqrt(number.toDouble()).toInt()) {
20         if (number % i == 0) return false
21     }
22     return true
23 }
24
25
26
```

The Run console shows the following output:

```
50
Введите конец диапазона:
70
Простые числа в диапазоне от 50 до 70:
59
61
67
Process finished with exit code 0
```

21.

```
import java.time.LocalDate
```

```
import java.time.YearMonth
```

```
fun main() {
```

```
    println("Введите год:")
```

```
    val year = readLine()?.toIntOrNull() ?: return
```

```
    println("Введите месяц:")
```

```
    val month = readLine()?.toIntOrNull() ?: return
```

```
    val yearMonth = YearMonth.of(month, year)
```

```
    val daysInMonth = yearMonth.lengthOfMonth()
```

```
    println("Даты в месяце $month $year:")
```

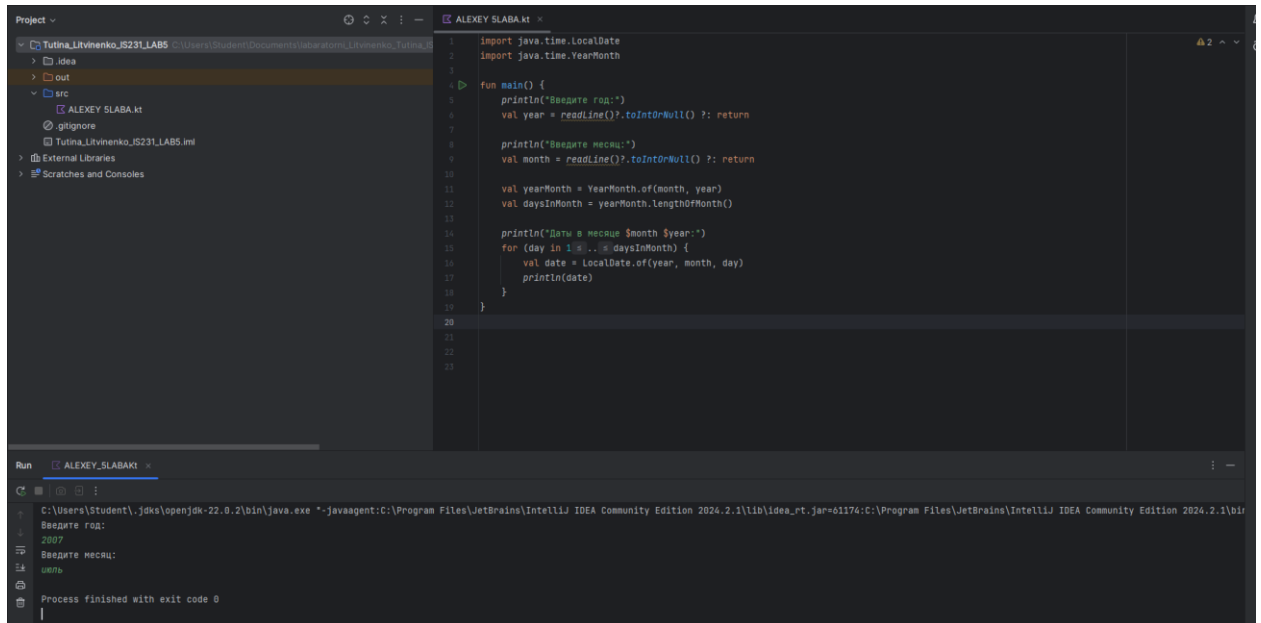
```
    for (day in 1..daysInMonth) {
```

```
        val date = LocalDate.of(year, month, day)
```

```
        println(date)
```

```
}
```

}



22.

import kotlin.random.Random

fun main() {

 val randomNumber = Random.nextInt(1, 101)

 var guess: Int? = null

 var attempts = 0

 println("Угадайте число от 1 до 100!")

 while (guess != randomNumber) {

 println("Введите ваше предположение:")

 guess = readLine()?.toIntOrNull()

 if (guess == null) {

 println("Пожалуйста, введите целое число.")

 continue

 }

 attempts++

```

when {

    guess < randomNumber -> println("Загаданное число больше.")

    guess > randomNumber -> println("Загаданное число меньше.")

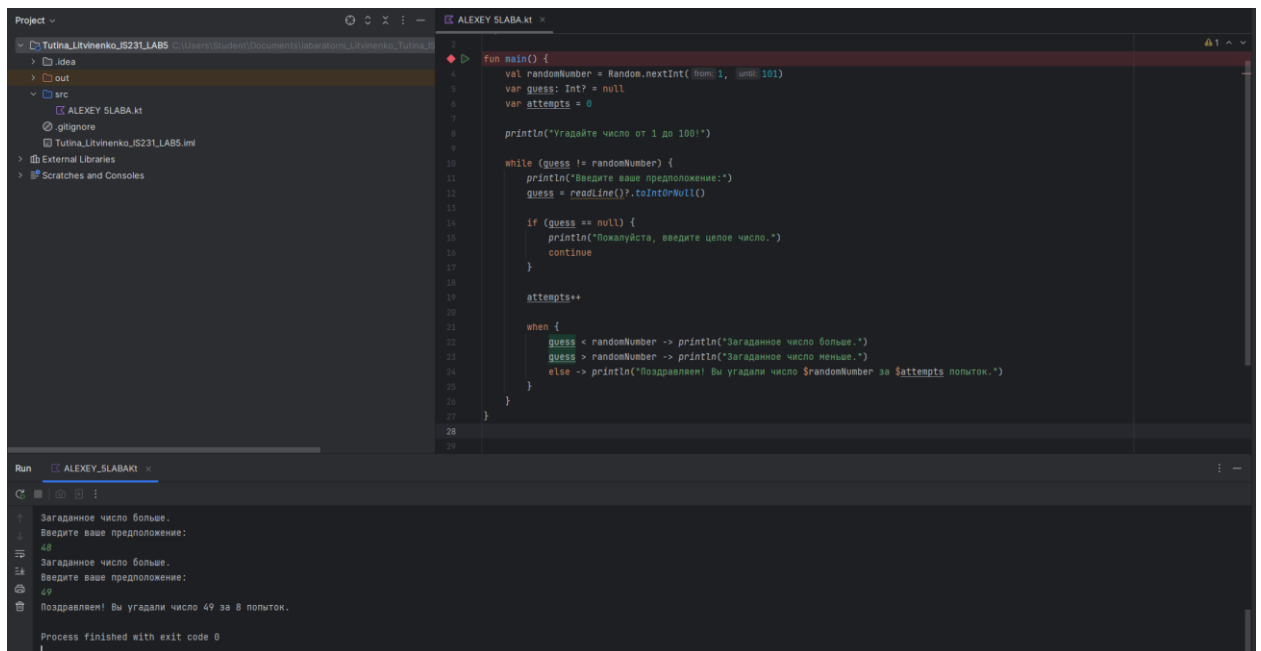
    else -> println("Поздравляем! Вы угадали число $randomNumber за $attempts попыток.")

}

}

}

```



23.

```

fun main() {

    println("Введите две цифры для выполнения операций. Напишите 'стоп' для завершения.")

    while (true) {

        println("Введите первую цифру (или 'стоп'):")

        val input1 = readLine()

        if (input1.equals("стоп", ignoreCase = true)) break

        println("Введите вторую цифру (или 'стоп'):")

        val input2 = readLine()
    }
}

```

```
if (input2.equals("стоп", ignoreCase = true)) break

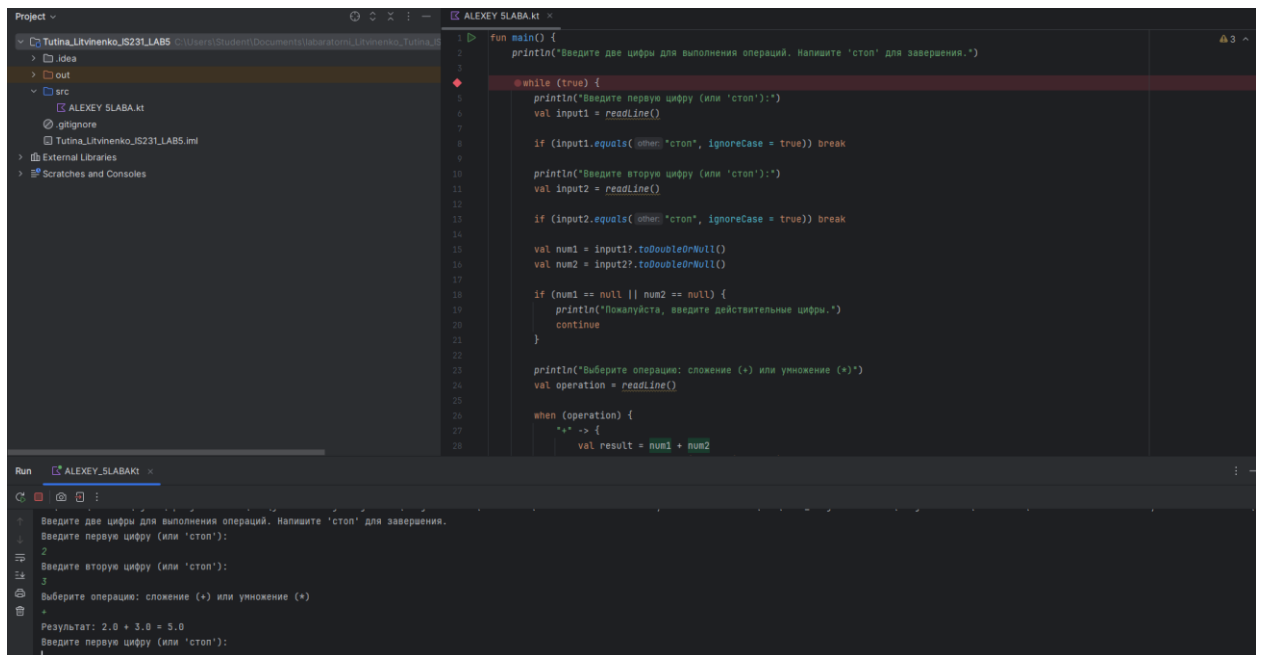
val num1 = input1?.toDoubleOrNull()
val num2 = input2?.toDoubleOrNull()

if (num1 == null || num2 == null) {
    println("Пожалуйста, введите действительные цифры.")
    continue
}

println("Выберите операцию: сложение (+) или умножение (*")
val operation = readLine()

when (operation) {
    "+" -> {
        val result = num1 + num2
        println("Результат: $num1 + $num2 = $result")
    }
    "*" -> {
        val result = num1 * num2
        println("Результат: $num1 * $num2 = $result")
    }
    else -> println("Неверная операция. Пожалуйста, выберите '+' или '*'.")
}

println("Программа завершена.")
}
```



24.

```
fun main() {
```

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

```
    val transposedMatrix = transpose(matrix)
```

```
    println("Исходная матрица:")
```

```
    printMatrix(matrix)
```

```
    println("Транспонированная матрица:")
```

```
    printMatrix(transposedMatrix)
```

```
}
```

```
fun transpose(matrix: Array<Array<Int>>): Array<Array<Int>> {
```

```
    val rowCount = matrix.size
```

```
    val colCount = matrix[0].size
```

```
    val transposed = Array(colCount) { Array(rowCount) { 0 } }
```

```

    for (i in 0 until rowCount) {
        for (j in 0 until colCount) {
            transposed[j][i] = matrix[i][j]
        }
    }

    return transposed
}

fun printMatrix(matrix: Array<Array<Int>>) {
    for (row in matrix) {
        println(row.joinToString(" "))
    }
}

```

The screenshot shows an IDE with a project named 'ALEXEY_SLABA.kt'. The code defines a `transpose` function and a `main` function. The `main` function creates a 3x3 matrix, transposes it, and prints both. The output in the Run console shows the original matrix, the transposed matrix, and the process finished with exit code 0.

```

1 fun main() {
2     val matrix = arrayOf(
3         arrayOf(1, 2, 3),
4         arrayOf(4, 5, 6),
5         arrayOf(7, 8, 9)
6     )
7
8     val transposedMatrix = transpose(matrix)
9
10    println("Исходная матрица:")
11    printMatrix(matrix)
12
13    println("Транспонированная матрица:")
14    printMatrix(transposedMatrix)
15 }
16
17 fun transpose(matrix: Array<Array<Int>>): Array<Array<Int>> {
18     val rowCount = matrix.size
19     val colCount = matrix[0].size
20     val transposed = Array(colCount) { Array(rowCount) { 0 } }
21
22     for (i in 0 until rowCount) {
23         for (j in 0 until colCount) {
24             transposed[j][i] = matrix[i][j]
25         }
26     }
27
28     return transposed
29 }

```

```

Run ALEXEY_SLABA.kt
1 2 3
4 5 6
7 8 9
Транспонированная матрица:
1 4 7
2 5 8
3 6 9
Process finished with exit code 0

```

25.

```

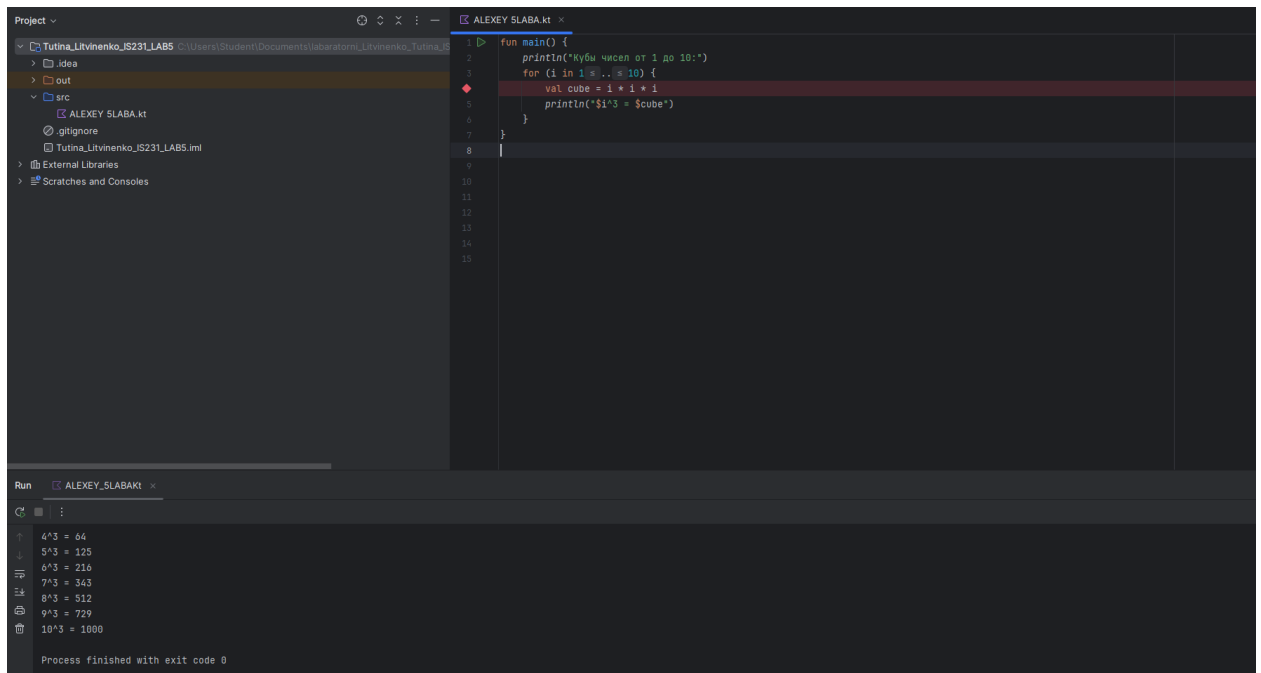
fun main() {
    println("Кубы чисел от 1 до 10:")

    for (i in 1..10) {
        val cube = i * i * i
        println("$i^3 = $cube")
    }
}

```

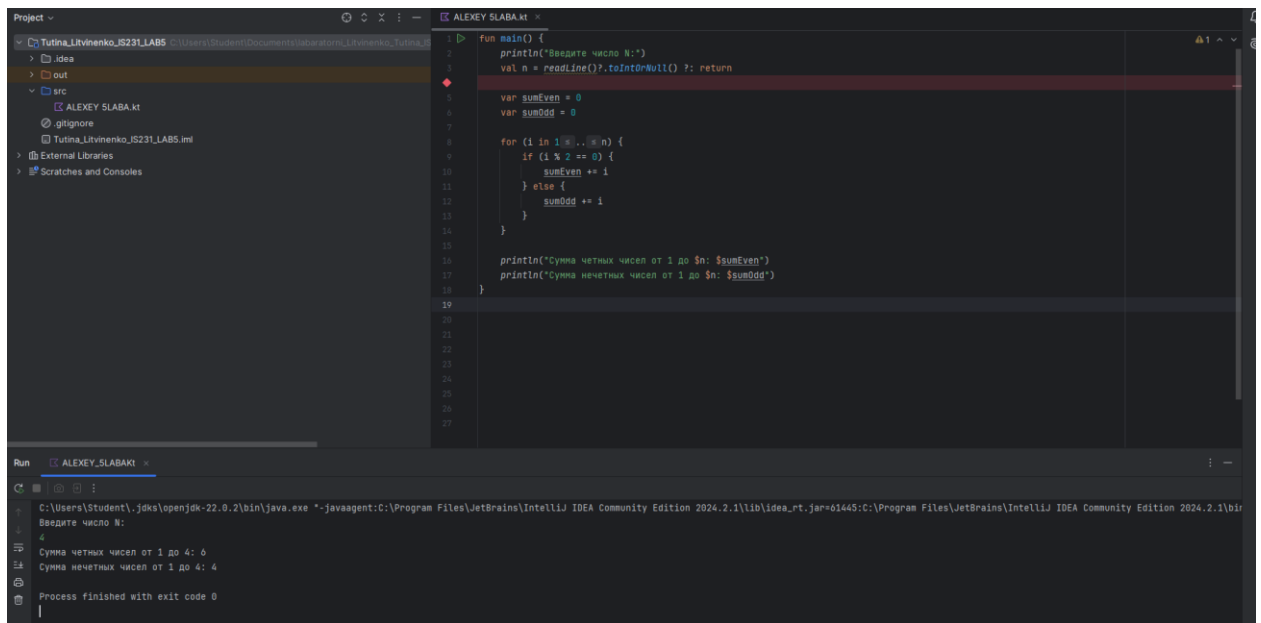

}

}



26.

```
fun main() {  
    println("Введите число N:")  
    val n = readLine()?.toIntOrNull() ?: return  
  
    var sumEven = 0  
    var sumOdd = 0  
  
    for (i in 1..n) {  
        if (i % 2 == 0) {  
            sumEven += i  
        } else {  
            sumOdd += i  
        }  
    }  
  
    println("Сумма четных чисел от 1 до $n: $sumEven")  
    println("Сумма нечетных чисел от 1 до $n: $sumOdd")  
}
```



27.

```
fun main() {
```

```
    println("Введите число N:")
```

```
    val n = readLine()?.toIntOrNull() ?: return
```

```
    for (i in 1..n) {
```

```
        // Печатаем пробелы для выравнивания
```

```
        repeat(n - i) {
```

```
            print(" ")
```

```
        }
```

```
        // Печатаем числа от 1 до i
```

```
        for (j in 1..i) {
```

```
            print("$j ")
```

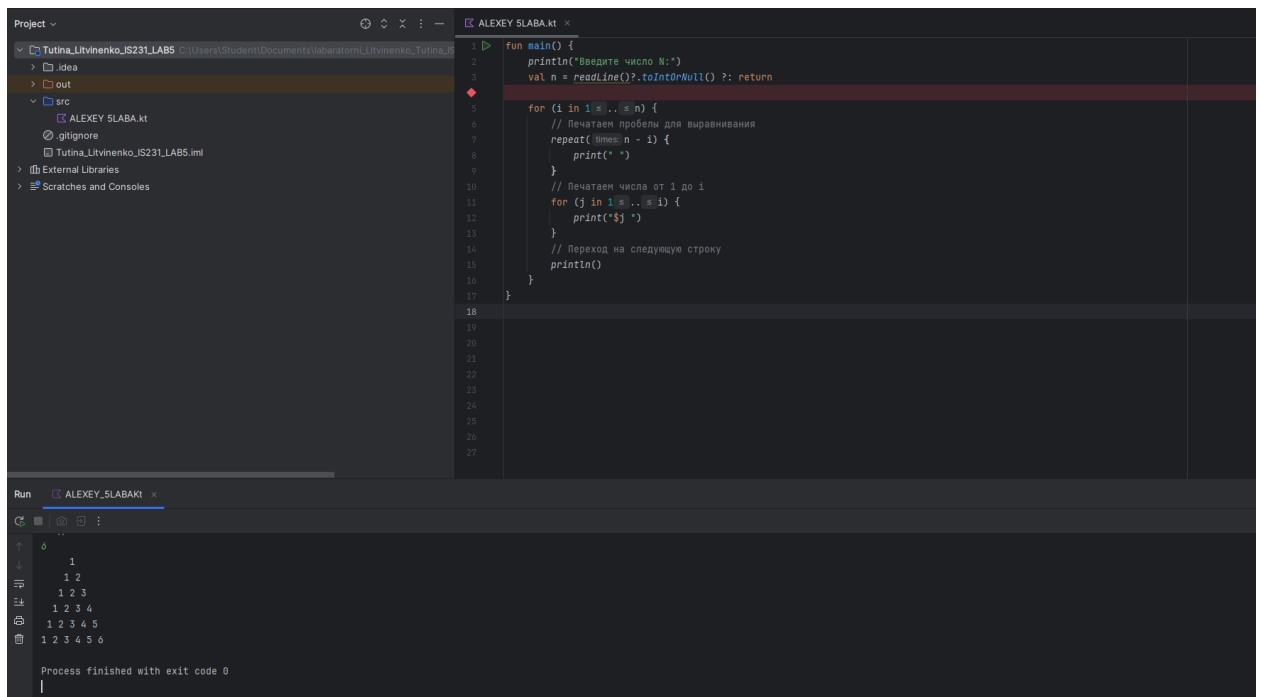
```
        }
```

```
        // Переход на следующую строку
```

```
        println()
```

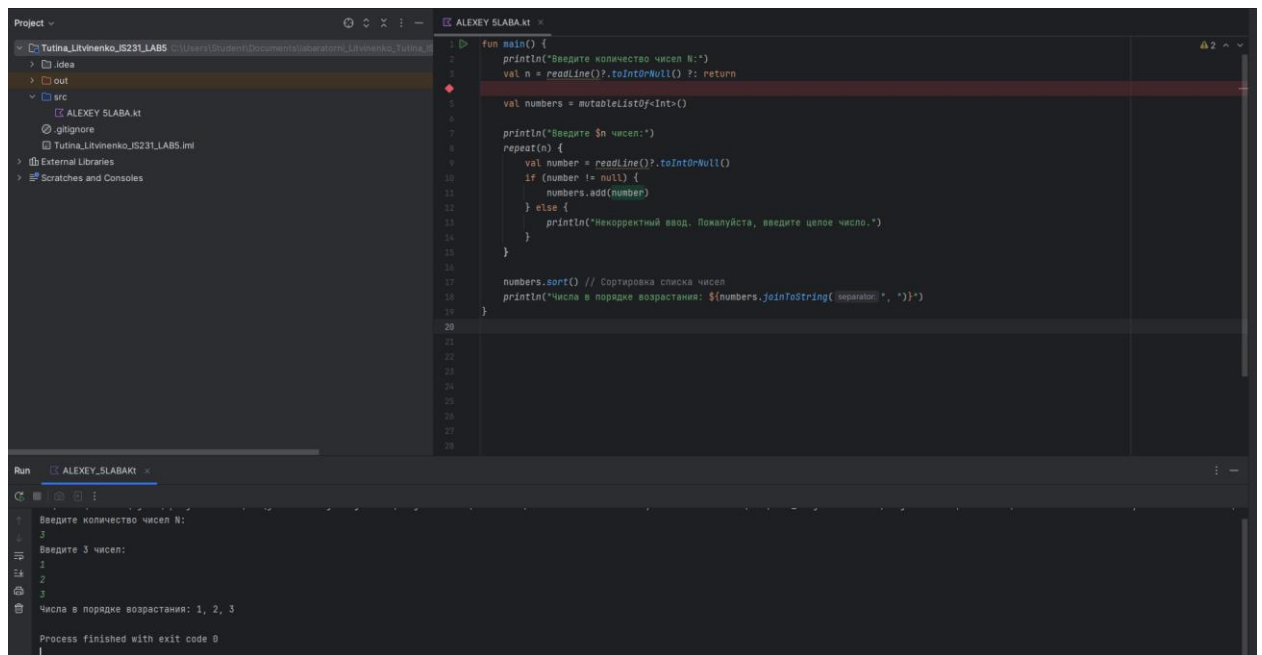
```
    }
```

```
}
```



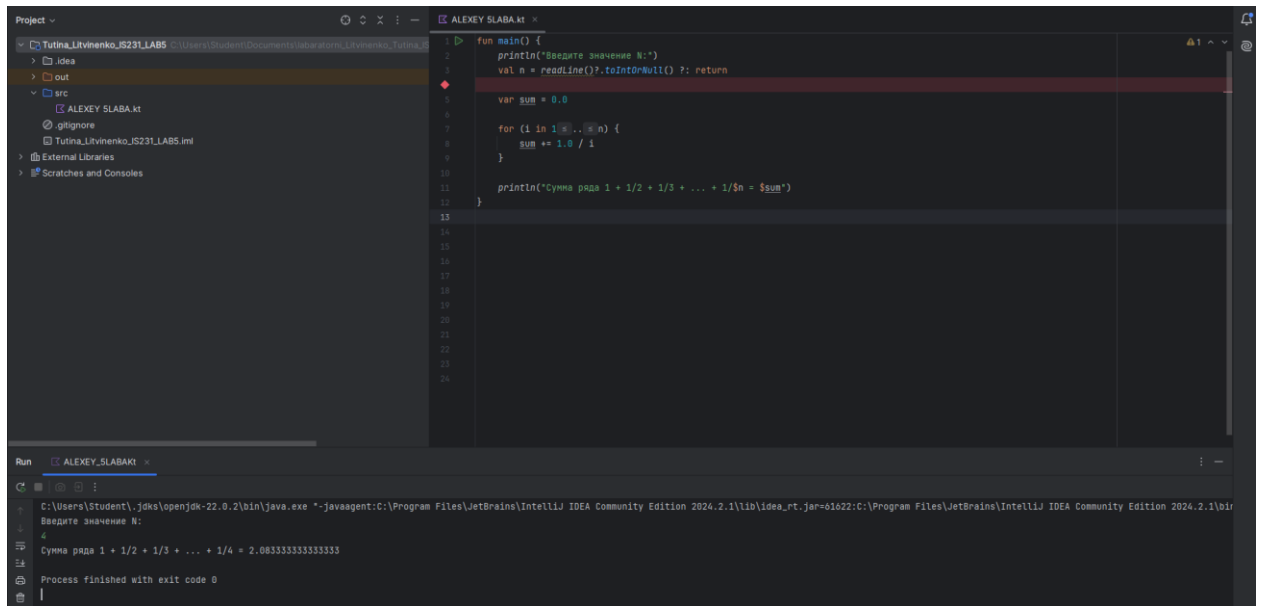
28.

```
fun main() {  
  
    println("Введите количество чисел N:")  
  
    val n = readLine()?.toIntOrNull() ?: return  
  
    val numbers = mutableListOf<Int>()  
  
    println("Введите $n чисел:")  
    repeat(n) {  
        val number = readLine()?.toIntOrNull()  
        if (number != null) {  
            numbers.add(number)  
        } else {  
            println("Некорректный ввод. Пожалуйста, введите целое число.")  
        }  
    }  
  
    numbers.sort() // Сортировка списка чисел  
    println("Числа в порядке возрастания: ${numbers.joinToString(" ", ")}")  
}
```



29.

```
fun main() {  
    println("Введите значение N:")  
    val n = readLine()?.toIntOrNull() ?: return  
  
    var sum = 0.0  
  
    for (i in 1..n) {  
        sum += 1.0 / i  
    }  
  
    println("Сумма ряда 1 + 1/2 + 1/3 + ... + 1/$n = $sum")  
}
```



30.

```
fun main() {  
    println("Введите целое число:")  
  
    val number = readLine()?.toIntOrNull() ?: return  
  
    val binary = number.toString(2)  
  
    println("Двоичное представление числа $number: $binary")  
}
```

