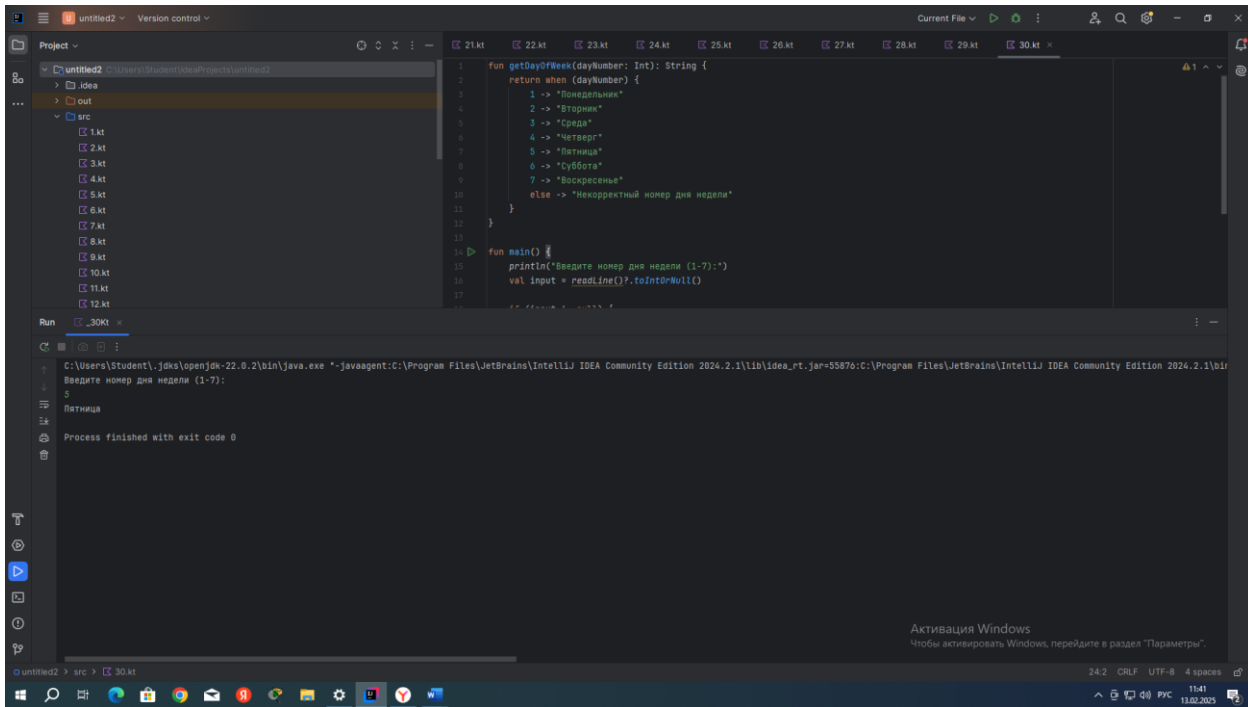


# Практическая работа 4

## Рыжевский

1)

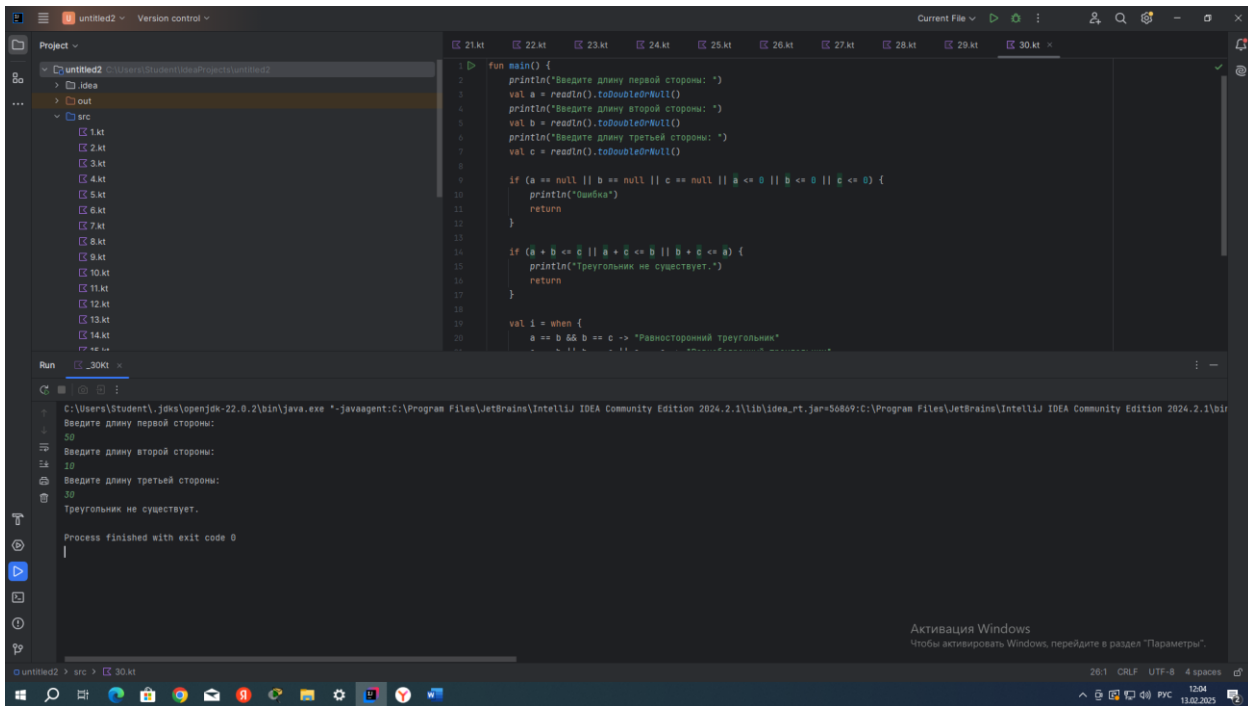


```
fun getDayOfWeek(dayNumber: Int): String {
    return when (dayNumber) {
        1 -> "Понедельник"
        2 -> "Вторник"
        3 -> "Среда"
        4 -> "Четверг"
        5 -> "Пятница"
        6 -> "Суббота"
        7 -> "Воскресенье"
        else -> "Некорректный номер дня недели"
    }
}

fun main() {
    println("Введите номер дня недели (1-7):")
    val input = readLine()?.toIntOrNull()

    if (input != null) {
        val dayOfWeek = getDayOfWeek(input)
        println(dayOfWeek)
    } else {
        println("Ошибка: введите число от 1 до 7.")
    }
}
```

2)



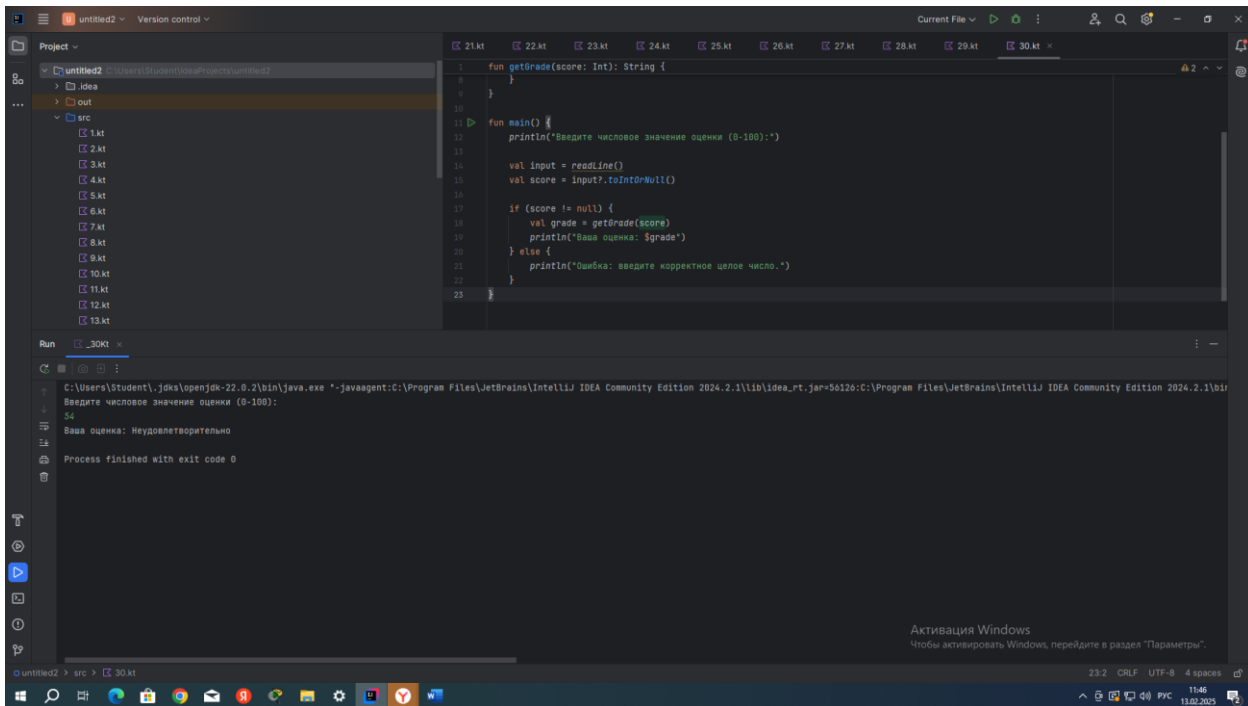
```
fun main() {
    println("Введите длину первой стороны: ")
    val a = readln().toDoubleOrNull()
    println("Введите длину второй стороны: ")
    val b = readln().toDoubleOrNull()
    println("Введите длину третьей стороны: ")
    val c = readln().toDoubleOrNull()

    if (a == null || b == null || c == null || a <= 0 || b <= 0 || c <= 0) {
        println("Ошибка")
        return
    }

    if (a + b <= c || a + c <= b || b + c <= a) {
        println("Треугольник не существует.")
        return
    }

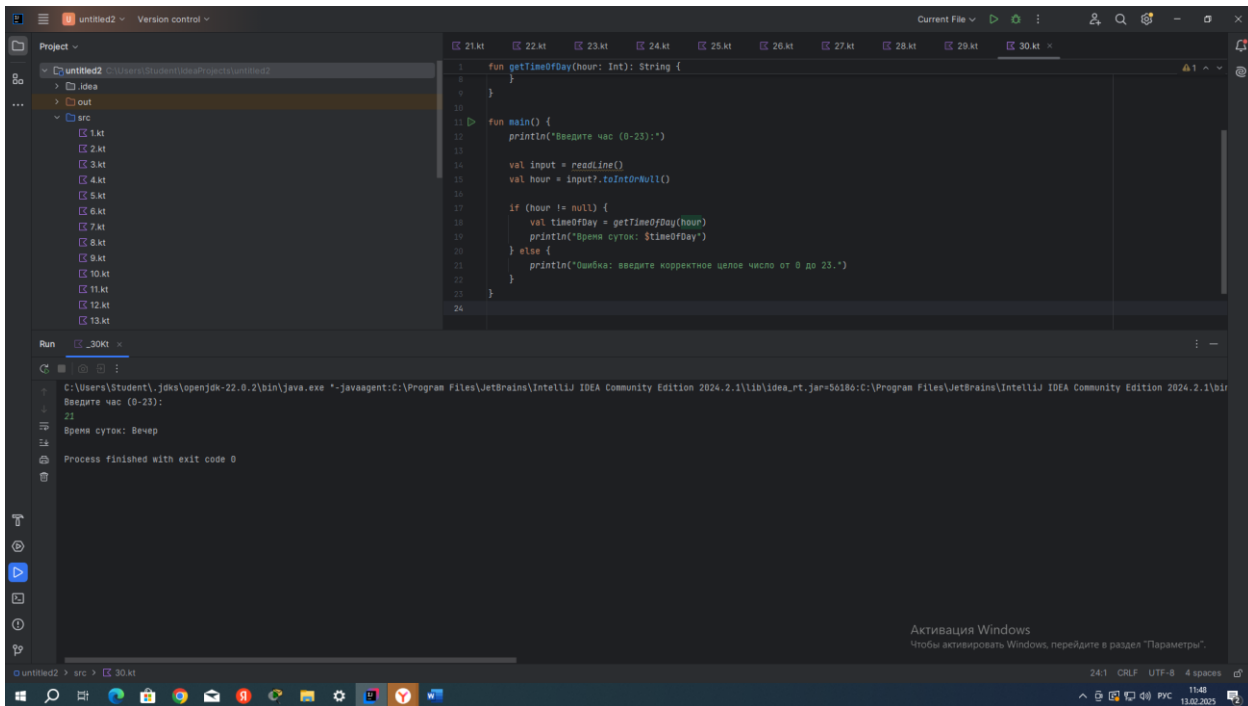
    val i = when {
        a == b && b == c -> "Равносторонний треугольник"
        a == b || b == c || a == c -> "Равнобедренный треугольник"
        else -> "Разносторонний треугольник"
    }
    println("Тип треугольника: $i")
}
```

3)



```
fun getGrade(score: Int): String {  
    return when {  
        score in 90..100 -> "Отлично"  
        score in 75..89 -> "Хорошо"  
        score in 60..74 -> "Удовлетворительно"  
        score in 0..59 -> "Неудовлетворительно"  
        else -> "Некорректное значение"  
    }  
}  
  
fun main() {  
    println("Введите числовое значение оценки (0-100):")  
  
    val input = readLine()  
    val score = input?.toIntOrNull()  
  
    if (score != null) {  
        val grade = getGrade(score)  
        println("Ваша оценка: $grade")  
    } else {  
        println("Ошибка: введите корректное целое число.")  
    }  
}
```

4)



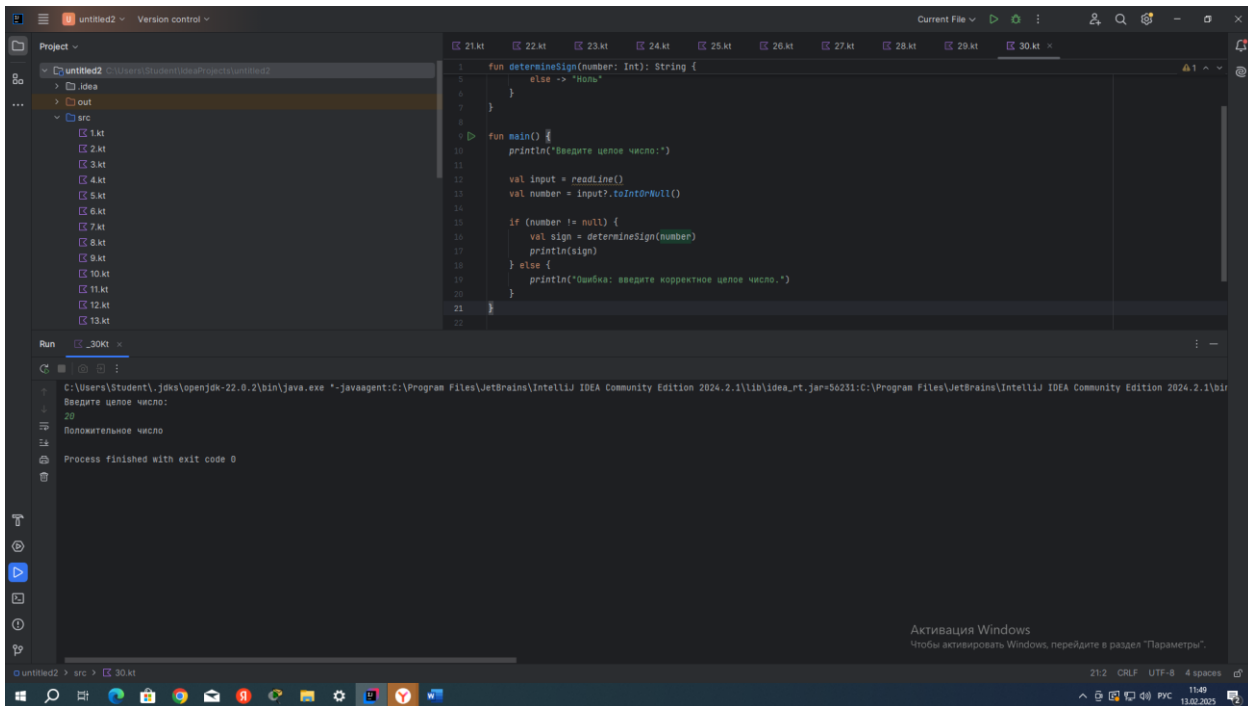
```
fun getTimeOfDay(hour: Int): String {
    return when (hour) {
        in 5..11 -> "Утро"
        in 12..17 -> "День"
        in 18..21 -> "Вечер"
        in 22..23, in 0..4 -> "Ночь"
        else -> "Некорректное значение"
    }
}

fun main() {
    println("Введите час (0-23):")

    val input = readLine()
    val hour = input?.toIntOrNull()

    if (hour != null) {
        val timeOfDay = getTimeOfDay(hour)
        println("Время суток: $timeOfDay")
    } else {
        println("Ошибка: введите корректное целое число от 0 до 23.")
    }
}
```

5)



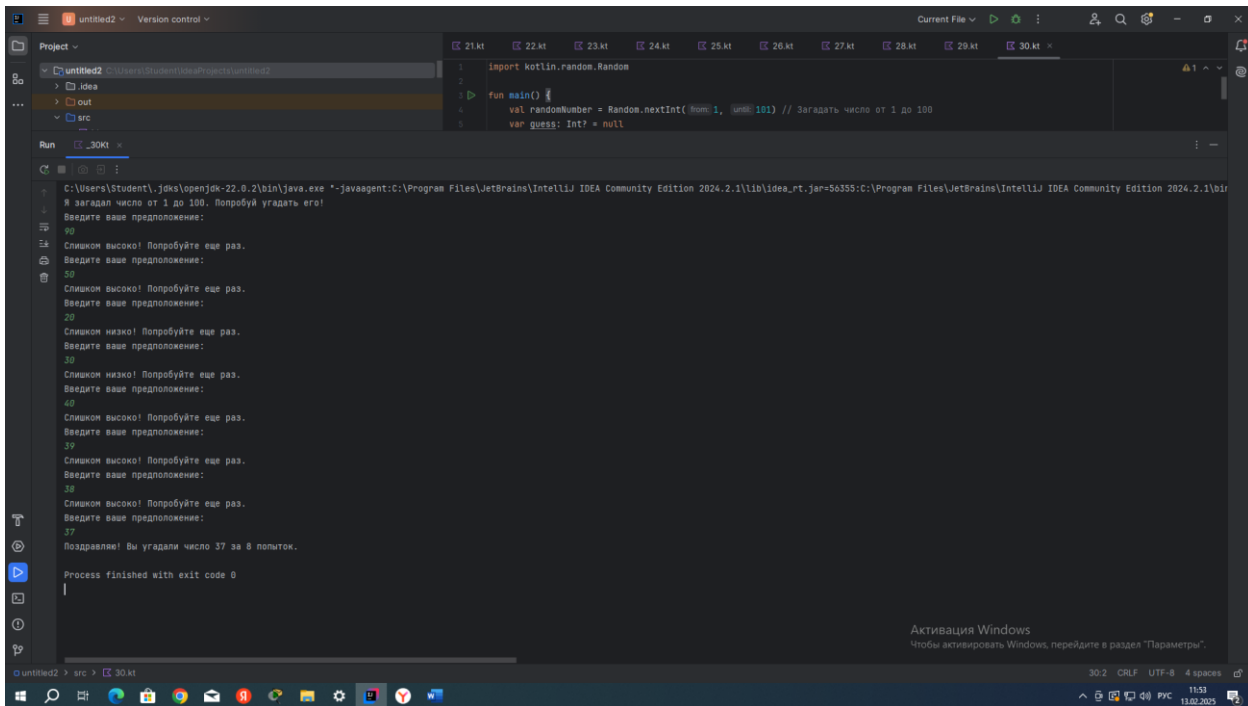
```
fun determineSign(number: Int): String {
    return when {
        number > 0 -> "Положительное число"
        number < 0 -> "Отрицательное число"
        else -> "Ноль"
    }
}

fun main() {
    println("Введите целое число:")

    val input = readLine()
    val number = input?.toIntOrNull()

    if (number != null) {
        val sign = determineSign(number)
        println(sign)
    } else {
        println("Ошибка: введите корректное целое число.")
    }
}
```

6)



```
import kotlin.random.Random

fun main() {
    val randomNumber = Random.nextInt(1, 101) // Загадать число от 1 до 100
    var guess: Int? = null
    var attempts = 0

    println("Я загадал число от 1 до 100. Попробуй угадать его!")

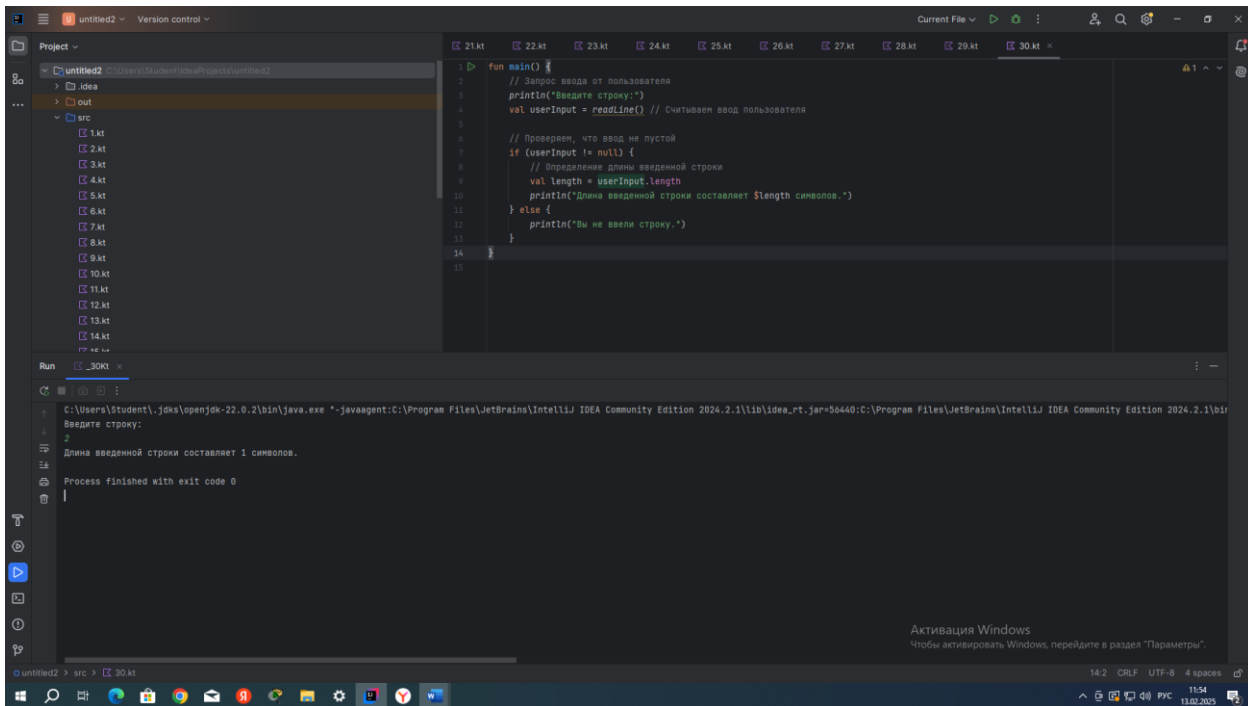
    while (guess != randomNumber) {
        println("Введите ваше предположение:")
        val input = readLine()

        // Проверка на корректность ввода
        guess = input?.toIntOrNull()

        if (guess == null) {
            println("Ошибка: введите корректное целое число.")
            continue
        }

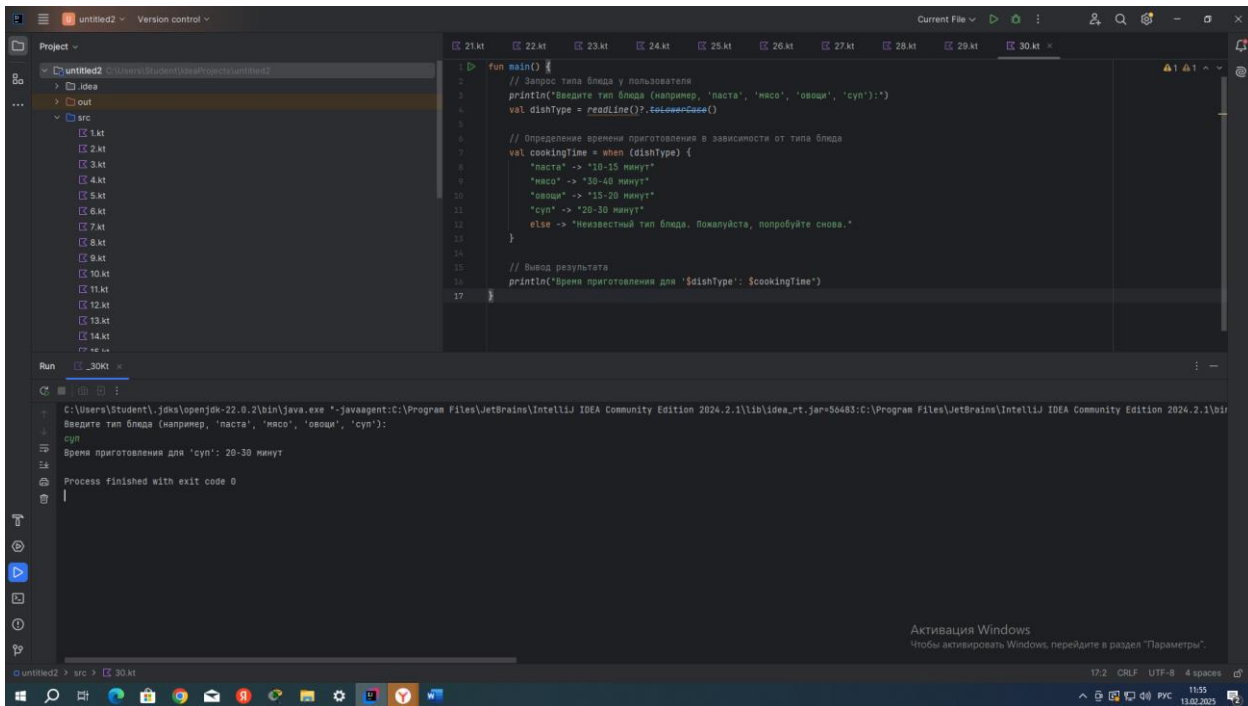
        attempts++

        when {
            guess < randomNumber -> println("Слишком низко! Попробуйте еще раз.")
            guess > randomNumber -> println("Слишком высоко! Попробуйте еще раз.")
            else -> println("Поздравляю! Вы угадали число $randomNumber за $attempts попыток.")
        }
    }
}
```



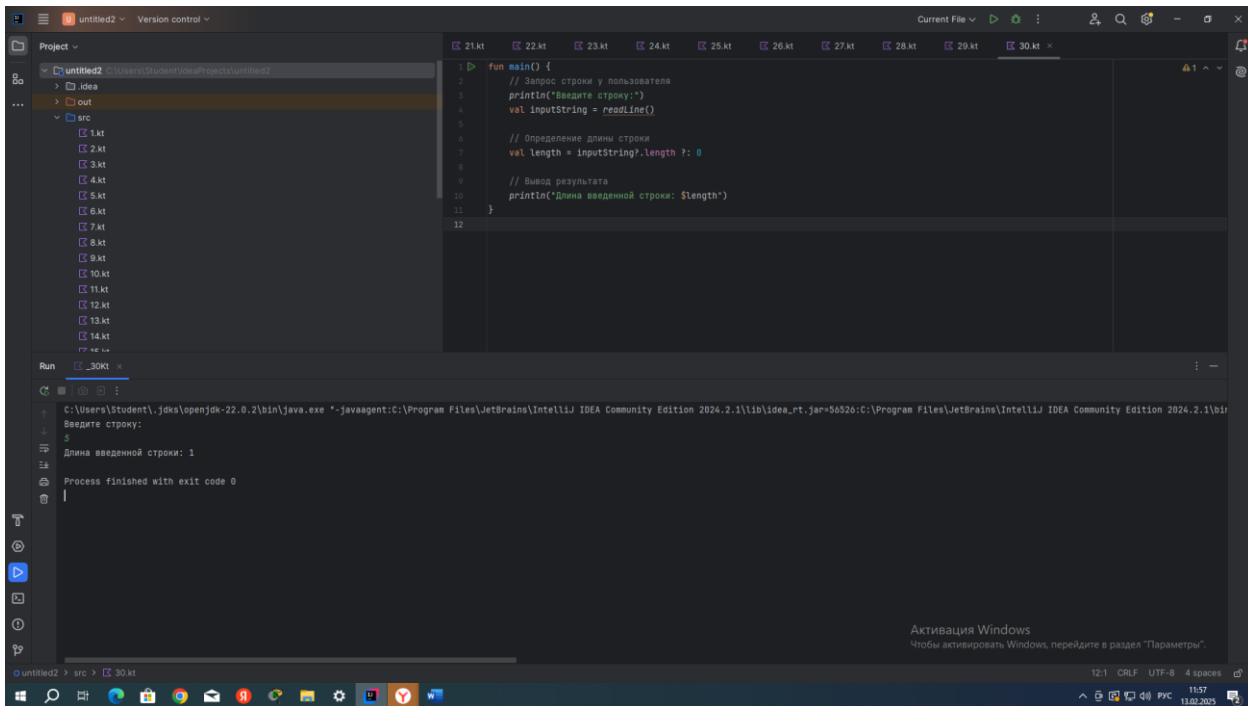
```
fun main() {  
    // Запрос ввода от пользователя  
    println("Введите строку:")  
    val userInput = readLine() // Считываем ввод пользователя  
  
    // Проверим, что ввод не пустой  
    if (userInput != null) {  
        // Определение длины введенной строки  
        val length = userInput.length  
        println("Длина введенной строки составляет $length символов.")  
    } else {  
        println("Вы не ввели строку.")  
    }  
}
```

8)



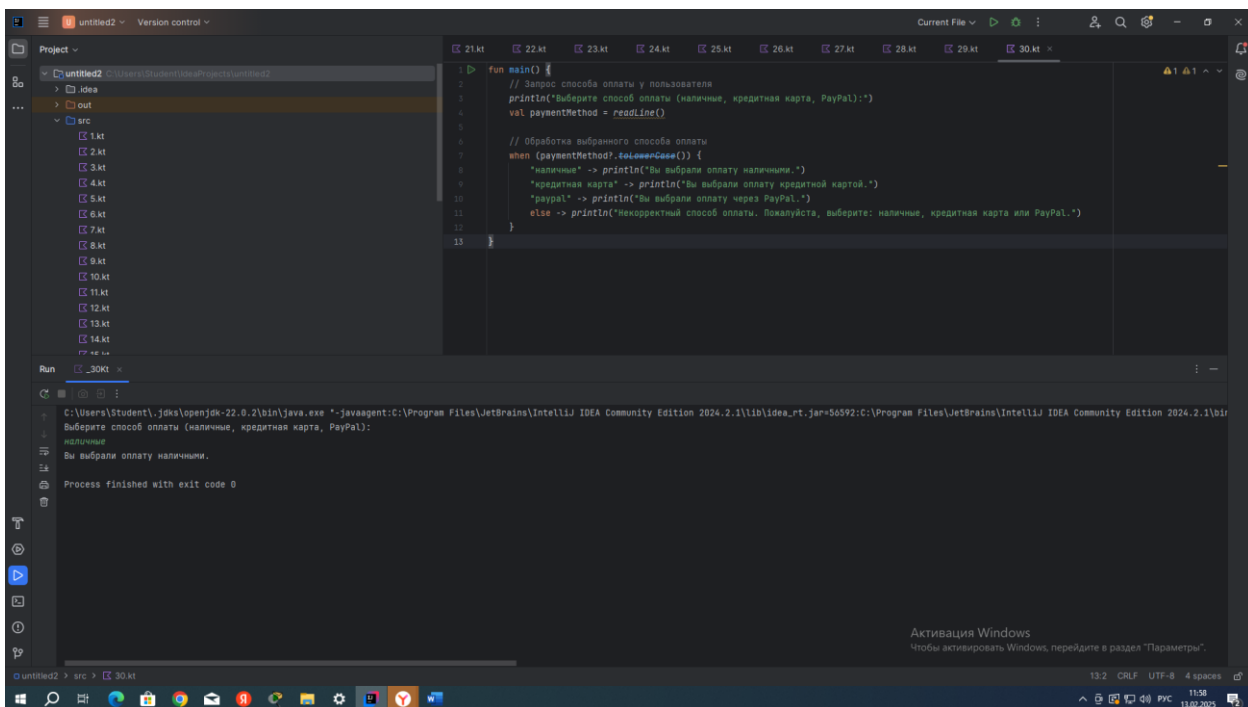
```
fun main() {  
    // Запрос типа блюда у пользователя  
    println("Введите тип блюда (например, 'паста', 'мясо', 'овощи', 'суп'):")  
    val dishType = readLine()?.toLowerCase()  
  
    // Определение времени приготовления в зависимости от типа блюда  
    val cookingTime = when (dishType) {  
        "паста" -> "10-15 минут"  
        "мясо" -> "30-40 минут"  
        "овощи" -> "15-20 минут"  
        "суп" -> "20-30 минут"  
        else -> "Неизвестный тип блюда. Пожалуйста, попробуйте снова."  
    }  
  
    // Вывод результата  
    println("Время приготовления для '$dishType': $cookingTime")  
}
```





```
fun main() {  
    // Запрос строки у пользователя  
    println("Введите строку:")  
    val inputString = readLine()  
  
    // Определение длины строки  
    val length = inputString?.length ?: 0  
  
    // Вывод результата  
    println("Длина введенной строки: $length")  
}
```

10)



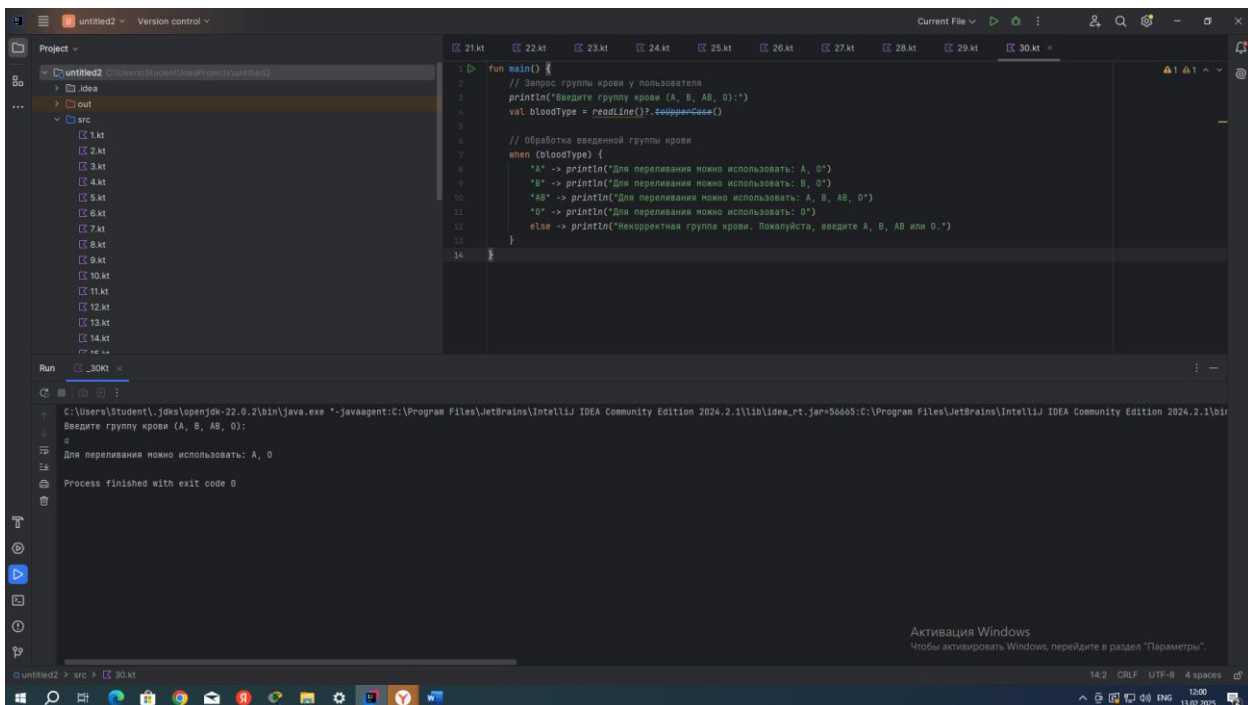
```

fun main() {
    // Запрос способа оплаты у пользователя
    println("Выберите способ оплаты (наличные, кредитная карта, PayPal):")
    val paymentMethod = readLine()

    // Обработка выбранного способа оплаты
    when (paymentMethod?.toLowerCase()) {
        "наличные" -> println("Вы выбрали оплату наличными.")
        "кредитная карта" -> println("Вы выбрали оплату кредитной картой.")
        "paypal" -> println("Вы выбрали оплату через PayPal.")
        else -> println("Некорректный способ оплаты. Пожалуйста, выберите: наличные, кредитная карта или PayPal.")
    }
}

```

11)



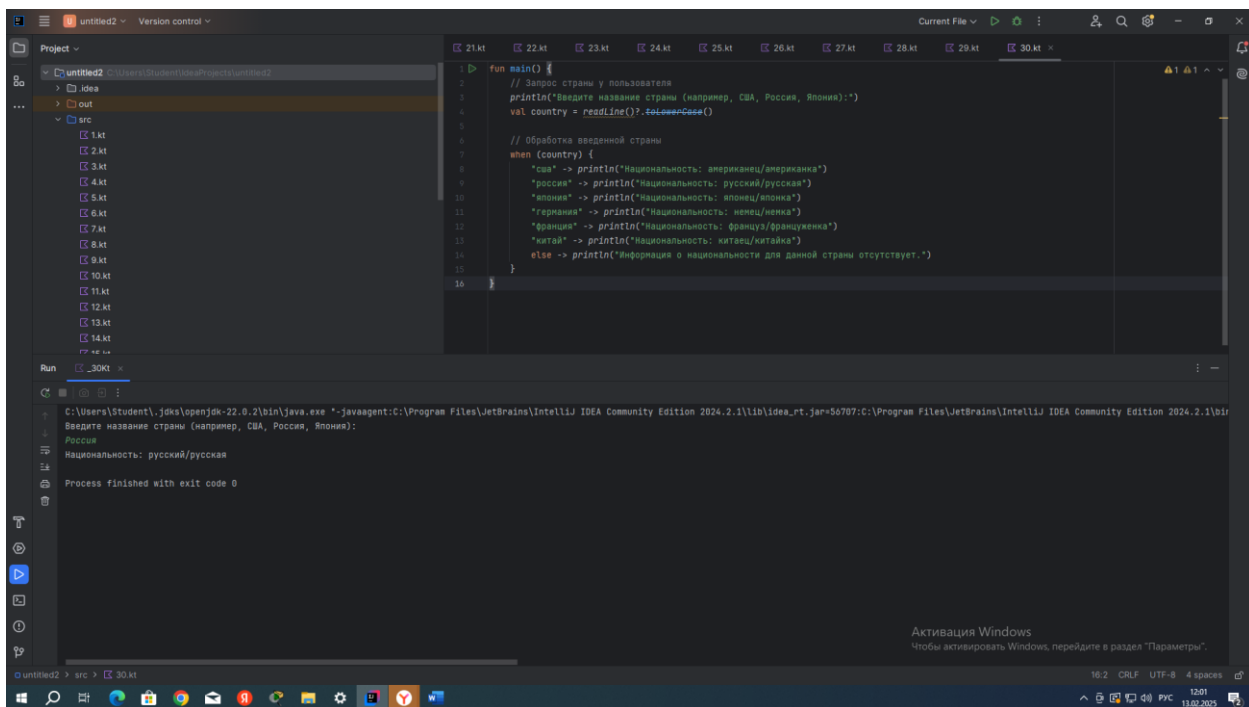
```

fun main() {
    // Запрос группы крови у пользователя
    println("Введите группу крови (A, B, AB, O):")
    val bloodType = readLine()?.toUpperCase()

    // Обработка введенной группы крови
    when (bloodType) {
        "A" -> println("Для переливания можно использовать: A, O")
        "B" -> println("Для переливания можно использовать: B, O")
        "AB" -> println("Для переливания можно использовать: A, B, AB, O")
        "O" -> println("Для переливания можно использовать: O")
        else -> println("Некорректная группа крови. Пожалуйста, введите A, B, AB или O.")
    }
}

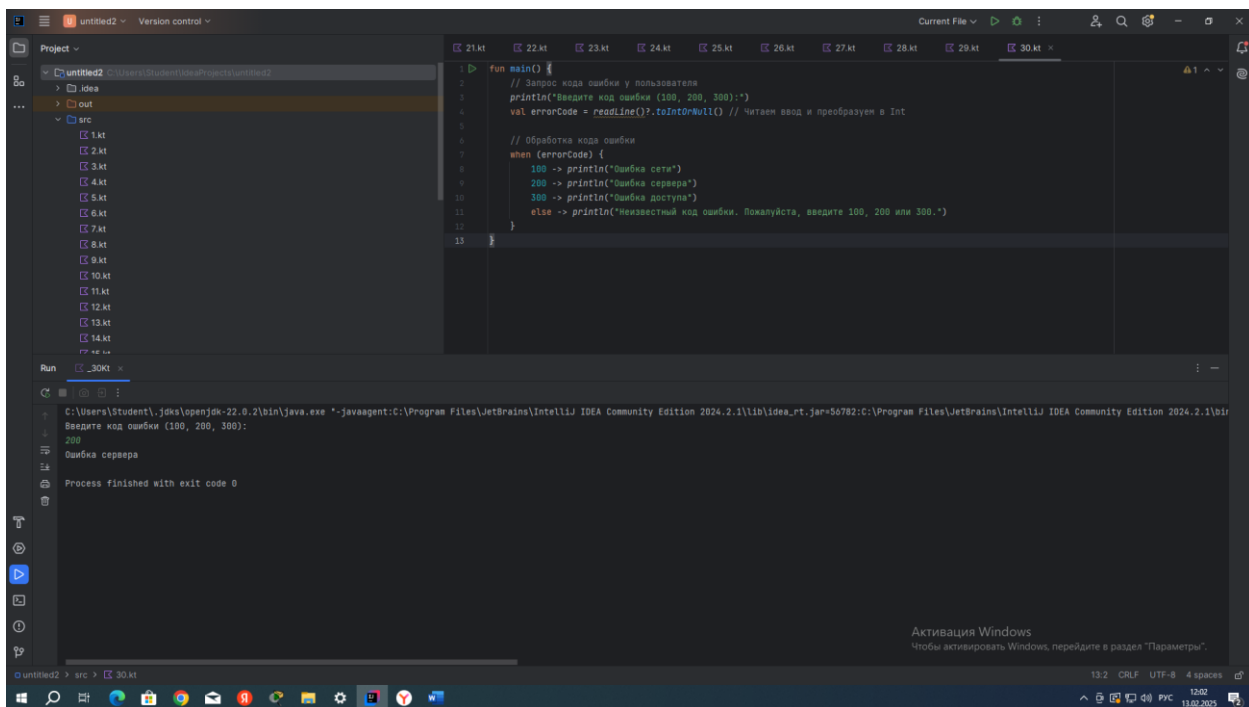
```

12)



```
fun main() {  
    // Запрос страны у пользователя  
    println("Введите название страны (например, США, Россия, Япония):")  
    val country = readLine()?.toLowerCase()  
  
    // Обработка введенной страны  
    when (country) {  
        "сша" -> println("Национальность: американец/американка")  
        "россия" -> println("Национальность: русский/русская")  
        "япония" -> println("Национальность: японец/японка")  
        "германия" -> println("Национальность: немец/немка")  
        "франция" -> println("Национальность: француз/француженка")  
        "китай" -> println("Национальность: китаец/китайка")  
        else -> println("Информация о национальности для данной страны  
отсутствует.")  
    }  
}
```

13)



The screenshot shows the IntelliJ IDEA IDE with a Kotlin file named `untitled2.kt`. The code defines a `main` function that prompts the user for an error code and handles it using a `when` expression. The IDE's Run window shows the execution output, indicating that the user entered `200` and the program printed `Ошибка сервера` (Server error). The status bar at the bottom indicates the file is encoded in UTF-8.

```
1 fun main() {  
2     // Запрос кода ошибки у пользователя  
3     println("Введите код ошибки (100, 200, 300):")  
4     val errorCode = readLine()?.toIntOrNull() // Читаем ввод и преобразуем в Int  
5  
6     // Обработка кода ошибки  
7     when (errorCode) {  
8         100 -> println("Ошибка сети")  
9         200 -> println("Ошибка сервера")  
10        300 -> println("Ошибка доступа")  
11        else -> println("Неизвестный код ошибки. Пожалуйста, введите 100, 200 или 300.")  
12    }  
13 }
```

Run: C:\Users\Student\jdk\openjdk-22.0.2\bin\java.exe -javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.2.1\lib\idea\_rt.jar=56782:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.2.1\bin  
Введите код ошибки (100, 200, 300):  
200  
Ошибка сервера  
Process finished with exit code 0

```
fun main() {  
    // Запрос кода ошибки у пользователя  
    println("Введите код ошибки (100, 200, 300):")  
    val errorCode = readLine()?.toIntOrNull() // Читаем ввод и преобразуем в  
    Int  
  
    // Обработка кода ошибки  
    when (errorCode) {  
        100 -> println("Ошибка сети")  
        200 -> println("Ошибка сервера")  
        300 -> println("Ошибка доступа")  
        else -> println("Неизвестный код ошибки. Пожалуйста, введите 100, 200  
или 300.")  
    }  
}
```