

## Практическая работа № 4

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"

}

}

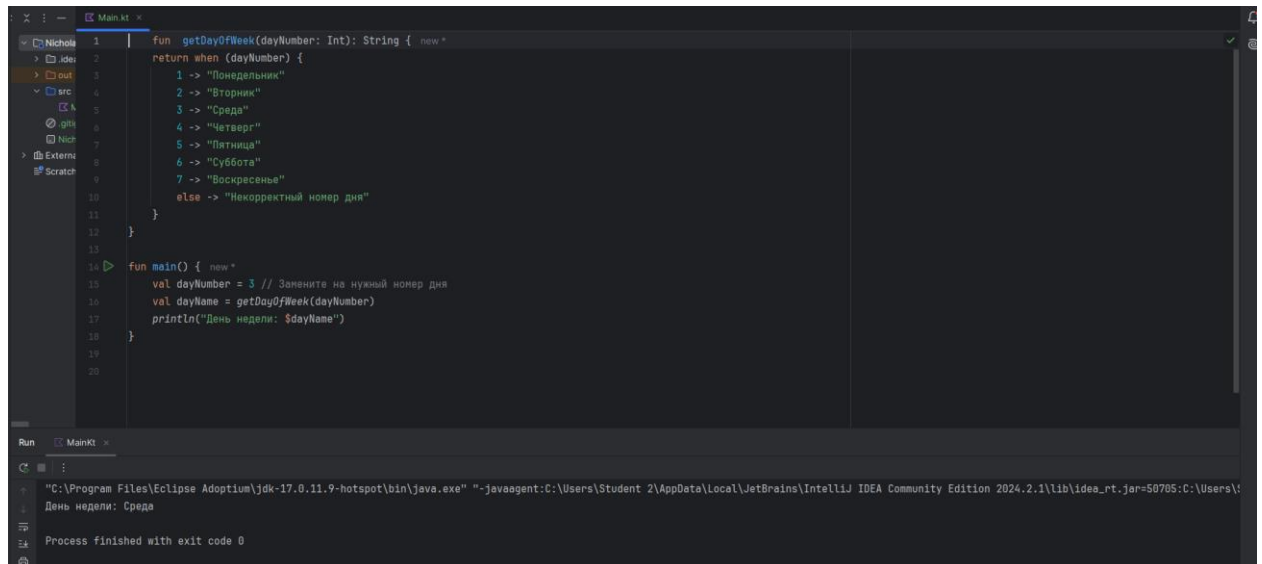
fun main() {

val dayNumber = 3 // Замените на нужный номер дня

val dayName = getDayOfWeek(dayNumber)

println("День недели: \$dayName")

}



2

fun determineTriangleType(a: Double, b: Double, c: Double): String {

// Проверка на существование треугольника

if (a <= 0 || b <= 0 || c <= 0 || a + b <= c || a + c <= b || b + c <= a) {

return "Треугольник не существует"

}

```
// Определение типа треугольника

return when {

    a == b && b == c -> "Равносторонний треугольник"

    a == b || b == c || a == c -> "Равнобедренный треугольник"

    else -> "Разносторонний треугольник"

}

}
```

```
fun main() {

    println("Введите длины сторон треугольника:")

    val a = readLine()!!.toDouble()

    val b = readLine()!!.toDouble()

    val c = readLine()!!.toDouble()

    val result = determineTriangleType(a, b, c)

    println(result)

}
```

The screenshot shows an IDE with a Kotlin file named 'Main.kt'. The code defines a function `determineTriangleType` that takes three doubles `a`, `b`, and `c` and returns a string representing the triangle type. It uses a `when` expression to handle different cases: equilateral, isosceles, and scalene. The `main` function prompts the user to enter the side lengths, reads them, and prints the result. The Run console shows the execution output: 'Введите длины сторон треугольника:' followed by user input '4 4 3' and the result 'Разносторонний треугольник'.

```
1 fun determineTriangleType(a: Double, b: Double, c: Double): String { new *
2     // Проверка на существование треугольника
3     if (a <= 0 || b <= 0 || c <= 0 || a + b <= c || a + c <= b || b + c <= a) {
4         return "Треугольник не существует"
5     }
6
7     // Определение типа треугольника
8     return when {
9         a == b && b == c -> "Равносторонний треугольник"
10        a == b || b == c || a == c -> "Равнобедренный треугольник"
11        else -> "Разносторонний треугольник"
12    }
13 }
14
15 fun main() { new *
16     println("Введите длины сторон треугольника:")
17     val a = readLine()!!.toDouble()
18     val b = readLine()!!.toDouble()
19     val c = readLine()!!.toDouble()
20
21     val result = determineTriangleType(a, b, c)
22     println(result)
23 }
```

Run Main.kt

```
"C:\Program Files\Eclipse Adoptium\jdk-17.0.11.9-hotspot\bin\java.exe" "-javaagent:C:\Users\Student 2\AppData\Local\JetBrains\IntelliJ IDEA Community Edition 2024.2.1\lib\idea_rt.jar=50813:C:\Users\Student 2\AppData\Local\JetBrains\IntelliJ IDEA Community Edition 2024.2.1\bin" -Didea.config.path=C:\Users\Student 2\AppData\Local\JetBrains\IntelliJ IDEA Community Edition 2024.2.1\config -Didea.system.path=C:\Users\Student 2\AppData\Local\JetBrains\IntelliJ IDEA Community Edition 2024.2.1\bin
Введите длины сторон треугольника:
4
4
3
Разносторонний треугольник
Process finished with exit code 0
```

```
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 -> "Недопустимое значение"

    }

}
```

```
}
```

```
fun main() {  
    println("Введите оценку (0-100):")  
    val input = readLine()  
  
    if (input != null) {  
        try {  
            val score = input.toInt()  
            val grade = getGrade(score)  
            println("Оценка: $grade")  
        } catch (e: NumberFormatException) {  
            println("Пожалуйста, введите целое число.")  
        }  
    } else {  
        println("Ошибка ввода.")  
    }  
}
```

4

```
fun getTimeOfDay(hour: Int): String {  
    return when (hour) {  
        in 0..5 -> "Ночь"  
        in 6..11 -> "Утро"  
        in 12..17 -> "День"  
        in 18..23 -> "Вечер"  
        else -> "Недопустимое время"  
    }  
}
```

```
fun main() {  
    println("Введите час (0-23):")  
    val input = readLine()  
  
    if (input != null) {  
        try {  
            val hour = input.toInt()  
            val timeOfDay = getTimeOfDay(hour)  
            println("Время суток: $timeOfDay")  
        }  
    }  
}
```

```

    } catch (e: NumberFormatException) {
        println("Пожалуйста, введите целое число.")
    }
} else {
    println("Ошибка ввода.")
}
}

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

```

```

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

    if (input != null) {
        try {
            val number = input.toInt()
            val sign = determineSign(number)
            println(sign)
        } catch (e: NumberFormatException) {
            println("Пожалуйста, введите целое число.")
        }
    } else {
        println("Ошибка ввода.")
    }
}

```

6 import kotlin.random.Random

```

fun main() {
    val numberToGuess = Random.nextInt(1, 101) // Случайное число от 1 до 100
    var attempts = 0
    var guessed = false

```

```
println("Угадайте число от 1 до 100:")
```

```
while (!guessed) {
```

```
    println("Введите ваше предположение:")
```

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

```
    if (input != null) {
```

```
        try {
```

```
            val guess = input.toInt()
```

```
            attempts++
```

```
            when {
```

```
                guess < numberToGuess -> println("Слишком низко, попробуйте снова.")
```

```
                guess > numberToGuess -> println("Слишком высоко, попробуйте снова.")
```

```
                else -> {
```

```
                    guessed = true
```

```
                    println("Поздравляю! Вы угадали число $numberToGuess за $attempts попыток.")
```

```
                }
```

```
            }
```

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

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

```
        }
```

```
    } else {
```

```
        println("Ошибка ввода.")
```

```
    }
```

```
}
```

```
}
```

7.

```
fun main() {
```

```
    println("Введите строку:")
```

```
    val inputString = readLine() // Считываем строку от пользователя
```

```
    if (inputString != null) {
```

```
        val length = inputString.length // Определяем длину строки
```

```
        println("Длина введенной строки: $length")
```

```
    } else {
```

```
        println("Ошибка: строка не была введена.")
```

```
    }
```

```
}
```

8.

```
fun getDayOfWeek(dayNumber: Int): String {  
    return when (dayNumber) {  
        1 -> "Понедельник"  
        2 -> "Вторник"  
        3 -> "Среда"  
        4 -> "Четверг"  
        5 -> "Пятница"  
        6 -> "Суббота"  
        7 -> "Воскресенье"  
        else -> "Неверный номер, введите число от 1 до 7"  
    }  
}
```

```
fun main() {  
    println("Введите номер дня недели (1-7):")  
    val input = readLine()?.toIntOrNull()  
    if (input != null) {  
        val day = getDayOfWeek(input)  
        println(day)  
    } else {  
        println("Введите корректное число")  
    }  
}
```

9.

```
fun main() {  
    val строка = "Привет, мир!"  
    val длина = строка.length  
  
    println("Длина строки: $длина")  
}
```

10.

```
fun main() {  
    println("Введите способ оплаты (наличные, кредитная карта, PayPal):")  
    val способОплаты = readLine()  
  
    when (способОплаты) {
```

```

        "наличные" -> println("Вы выбрали оплату наличными.")
        "кредитная карта" -> println("Вы выбрали оплату кредитной картой.")
        "PayPal" -> println("Вы выбрали оплату через PayPal.")

        else -> println("Неизвестный способ оплаты. Пожалуйста, выберите: наличные, кредитная карта или PayPal.")
    }
}

```

11.

```

fun main() {
    println("Введите группу крови (A, B, AB, O):")
    val группаКрови = readLine()?.toUpperCase()

    when (группаКрови) {
        "A" -> println("Можно переливать: A, O")
        "B" -> println("Можно переливать: B, O")
        "AB" -> println("Можно переливать: A, B, AB, O")
        "O" -> println("Можно переливать: O")
        else -> println("Некорректная группа крови. Пожалуйста, введите A, B, AB или O.")
    }
}

```

12

```

fun main() {
    println("Введите название страны (США, Россия, Япония):")
    val страна = readLine()?.toLowerCase()

    when (страна) {
        "сша" -> println("Национальность: американская")
        "россия" -> println("Национальность: русская")
        "япония" -> println("Национальность: японская")
        "германия" -> println("Национальность: немецкая")
        "франция" -> println("Национальность: французская")
        else -> println("Неизвестная страна. Пожалуйста, введите одну из предложенных стран.")
    }
}

```

13

```

fun main() {
    println("Введите код ошибки (100, 200, 300):")
    val кодОшибки = readLine()?.toIntOrNull()
}

```

```
when (кодОшибки) {  
    100 -> println("Ошибка сети")  
    200 -> println("Ошибка сервера")  
    300 -> println("Неизвестная ошибка")  
    else -> println("Некорректный код ошибки. Пожалуйста, введите 100, 200 или 300.")  
}  
}
```