

**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ СВЯЗИ И МАССОВЫХ
КОММУНИКАЦИЙ**

Ордена Трудового Красного Знамени

**Федеральное государственное бюджетное образовательное учреждение
высшего образования**

«Московский технический университет связи и информатики»

Кафедра «Математическая кибернетика и информационные технологии»

**Лабораторная работа №2
по дисциплине
«Функциональное программирование»**

Выполнил: студент группы

БВТ2201

Нъяти Каелиле

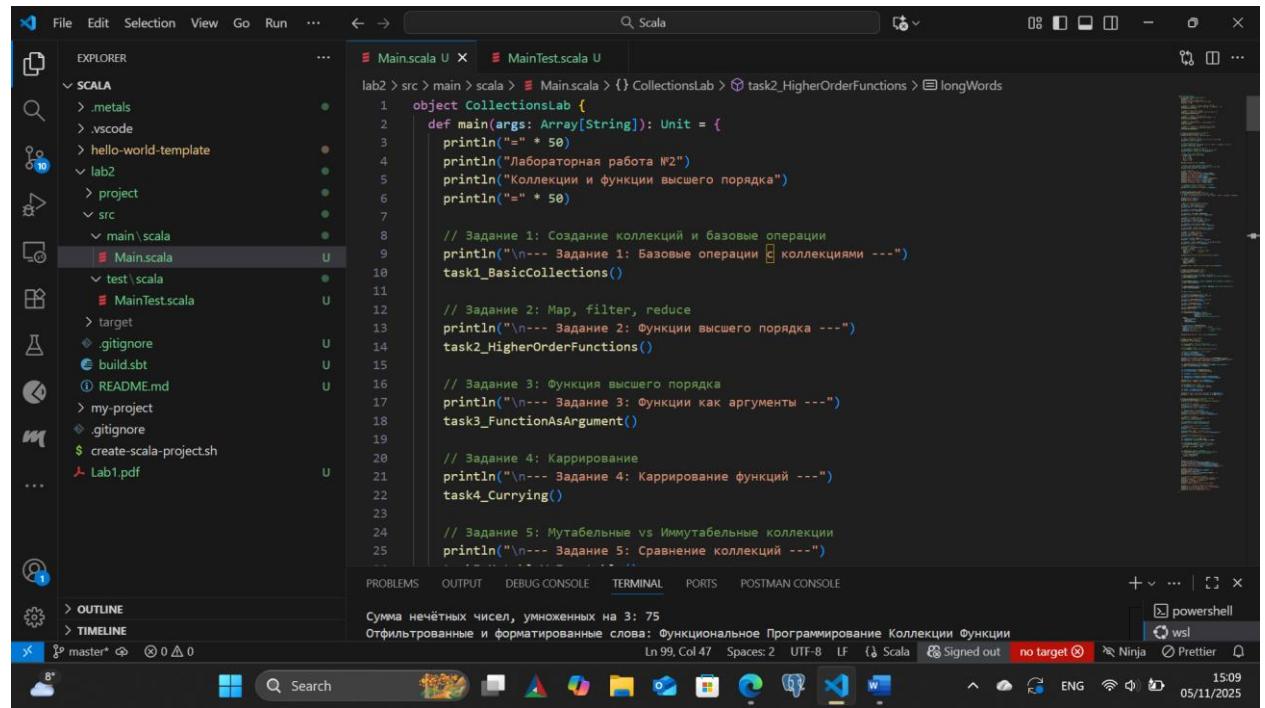
Москва

2025

Цель работы

Закрепить базовый синтаксис Scala. Научиться использовать функции высшего порядка (map, filter, reduce). Понять преимущества иммутабельных коллекций.

Ход работы



The screenshot shows the IntelliJ IDEA interface with the following details:

- File Structure (EXPLORER):** Shows the project structure with modules SCALA, lab2, and test\scala. Under test\scala, Main.scala and MainTest.scala are selected.
- Code Editor:** Displays the Main.scala file content:

```
object CollectionsLab {  
    def main(args: Array[String]): Unit = {  
        println("= * 50)  
        println("Лабораторная работа №2")  
        println("Коллекции и функции высшего порядка")  
        println("= * 50)  
  
        // Задание 1: Создание коллекций и базовые операции  
        println("\n--- Задание 1: Базовые операции с коллекциями ---")  
        task1_BasicCollections()  
  
        // Задание 2: Map, filter, reduce  
        println("\n--- Задание 2: Функции высшего порядка ---")  
        task2_HigherOrderFunctions()  
  
        // Задание 3: Функция высшего порядка  
        println("\n--- Задание 3: Функции как аргументы ---")  
        task3_FunctionAsArgument()  
  
        // Задание 4: Каррирование  
        println("\n--- Задание 4: Каррирование функций ---")  
        task4_Currying()  
  
        // Задание 5: Мутабельные vs Иммутабельные коллекции  
        println("\n--- Задание 5: Сравнение коллекций ---")  
    }  
}
```
- Terminal:** Shows the output of the Scala code execution:

```
Сумма нечётных чисел, умноженных на 3: 75  
Отфильтрованные и форматированные слова: Функциональное Программирование Коллекции Функции
```
- Bottom Bar:** Includes tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS, and POSTMAN CONSOLE. It also shows system status like battery level, network, and date/time (15:09, 05/11/2025).

Рис 1. Создание функцию main.

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows a project structure under the SCALA folder, including files like Main.scala, MainTest.scala, build.sbt, and README.md.
- Code Editor:** Displays Scala code for basic collection operations. The code includes:
 - Creating various collections (List, Set, Map).
 - Performing operations like summing odd numbers.
 - Printing collection contents.
- Terminal:** Shows the output of the Scala interpreter with messages like "Сумма нечётных чисел, умноженных на 3: 75".
- Bottom Bar:** Includes icons for file operations, search, and system status.

Рис 2. Создать коллекции различных типов и выполнить над ними базовые операции.

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows a project structure under the SCALA folder, including files like Main.scala, MainTest.scala, build.sbt, and README.md.
- Code Editor:** Displays Scala code for higher-order functions. The code includes:
 - Defining a CollectionsLab object.
 - Task 2: Higher Order Functions. It uses List, map, filter, and reduce.
 - Printing initial data and results.
- Terminal:** Shows the output of the Scala interpreter with messages like "Сумма нечётных чисел, умноженных на 3: 75".
- Bottom Bar:** Includes icons for file operations, search, and system status.

Рис 3. Использовать map, filter и reduce для обработки числовых и строковых коллекций.

The screenshot shows the VS Code interface with the Scala extension installed. The Explorer sidebar shows a project structure with files like `.metals`, `.vscode`, `hello-world-template`, `lab2`, `project`, `src`, `main\scala`, `Main.scala`, and `MainTest.scala`. The main editor tab displays Scala code for a higher-order function:

```
object CollectionsLab {  
    // задание 3. функция высшего порядка  
    def task3_FunctionAsArgument(): Unit = {  
        println("Функции высшего порядка:")  
  
        // Функция, принимающая другую функцию как аргумент  
        def processNumbers(numbers: List[Int], processor: Int => Int): List[Int] = {  
            numbers.map(processor)  
        }  
  
        def filterNumbers(numbers: List[Int], predicate: Int => Boolean): List[Int] = {  
            numbers.filter(predicate)  
        }  
  
        def aggregateNumbers(numbers: List[Int], aggregator: (Int, Int) => Int): Int = {  
            numbers.reduce(aggregator)  
        }  
  
        val data = List(1, 2, 3, 4, 5)  
  
        // Использование функций высшего порядка  
        val doubled = processNumbers(data, x => x * 2)  
        println(s"Удвоенные числа: $doubled")  
  
        val squares = processNumbers(data, x => x * x)  
        println(s"Квадраты чисел: $squares")  
    }  
}
```

The terminal below shows the output of the code execution:

```
Сумма нечетных чисел, умноженных на 3: 75  
Отфильтрованные и форматированные слова: Функциональное Программирование Коллекции Функции
```

Рис 4. Написать функцию высшего порядка, принимающую другую функцию как аргумент.

The screenshot shows the VS Code interface with the Scala extension installed. The Explorer sidebar shows a project structure with files like `.metals`, `.vscode`, `hello-world-template`, `lab2`, `project`, `src`, `main\scala`, `Main.scala`, and `MainTest.scala`. The main editor tab displays Scala code for currying:

```
object CollectionsLab {  
    // Задание 4: Каррирование функций  
    def task4_Currying(): Unit = {  
        println("Каррирование функций")  
  
        // Обычная функция с несколькими параметрами  
        def normalAdd(a: Int, b: Int, c: Int): Int = a + b + c  
  
        // Каррированная версия  
        def curriedAdd(a: Int)(b: Int)(c: Int): Int = a + b + c  
  
        // Частичное применение каррированной функции  
        val addFive = curriedAdd(5) _  
        val addFiveAndThree = addFive(3)  
        val finalResult = addFiveAndThree(2)  
  
        println(s"Обычная функция: normalAdd(5, 3, 2) = ${normalAdd(5, 3, 2)}")  
        println(s"Каррированная функция: curriedAdd(5)(3)(2) = ${curriedAdd(5)(3)(2)}")  
        println(s"Частичное применение: addFive(3)(2) = $finalResult")  
  
        // Практический пример: конфигурируемый фильтр  
        def createFilter(minValue: Int)(maxValue: Int)(number: Int): Boolean = {  
            number >= minValue && number <= maxValue  
        }  
    }  
}
```

The terminal below shows the output of the code execution:

```
Сумма нечетных чисел, умноженных на 3: 75  
Отфильтрованные и форматированные слова: Функциональное Программирование Коллекции Функции
```

Рис 5. Реализовать пример каррирования функции.

The screenshot shows the VS Code interface with the Scala extension installed. The Explorer sidebar shows a project structure under the 'SCALA' folder, including '.metals', '.vscode', 'hello-world-template', 'lab2', 'project', 'src', 'main\ scala', 'test\ scala', 'Main.scala', 'MainTest.scala', 'target', '.gitignore', 'build.sbt', 'README.md', 'my-project', '.gitignore', and '\$ create-scala-project.sh'. The terminal tab is active, displaying the output of the command 'sbt run'. The output shows the Scala REPL running the 'CollectionsLab' example, which demonstrates various collection operations like creation, filtering, mapping, and reduction.

```

nyathi@Moxamed-Idi:/mnt/c/Users/khaye/Desktop/Scala/lab2$ sbt run
[info] welcome to sbt 1.11.7 (Ubuntu Java 11.0.24)
[info] loading project definition from /mnt/c/Users/khaye/Desktop/Scala/lab2/project
[info] loading settings for project root from build.sbt...
[info] set current project to lab2 (in build file:/mnt/c/Users/khaye/Desktop/Scala/lab2/)
[info] running CollectionsLab
=====
Лабораторная работа №2
Коллекции и функции высшего порядка
=====

--- Задание 1: Базовые операции с коллекциями ---
Создание коллекций различных типов:
List[Int]: List(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
Seq[String]: List(Анна, Борис, Виктор, Галина, Дмитрий)
Set[Int] (уникальные): HashSet(5, 1, 2, 3, 4)
Map[String, Int]: Map(Анна -> 25, Борис -> 30, Виктор -> 35, Галина -> 28)
Vector[Double]: Vector(1.5, 2.7, 3.1, 4.9)

Базовые операции:
Первый элемент списка: 1
Последний элемент списка: 10
Хвост списка: List(2, 3, 4, 5, 6, 7, 8, 9, 10)
Длина списка: 10
Содержит ли список 5? true
Сумма всех элементов: 55
Минимальный элемент: 1
Максимальный элемент: 10
Новый список после добавления: List(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)

```

Рис 6. Сравнить работу с мутабельными и иммутабельными коллекциями.

This screenshot shows the same VS Code environment as above, but the terminal output has been modified to focus on functional programming concepts. It includes examples of map, filter, reduce, and higher-order functions, illustrating how they can be used to process collections in Scala.

```

Минимальный элемент: 1
Максимальный элемент: 10
Новый список после добавления: List(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)

--- Задание 2: Функции высшего порядка ---
Исходные данные:
Числа: List(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
Слова: List(функциональное, программирование, scala, коллекции, функции)

--- MAP (преобразование) ---
Квадраты чисел: List(1, 4, 9, 16, 25, 36, 49, 64, 81, 100)
Длины слов: List(4, 6, 5, 9, 7)
Слова в верхнем регистре: List(ФУНКЦИОНАЛЬНОЕ, ПРОГРАММИРОВАНИЕ, SCALA, КОЛЛЕКЦИИ, ФУНКЦИИ)

--- FILTER (фильтрация) ---
Чётные числа: List(2, 4, 6, 8, 10)
Длинные слова (>8 символов): List(функциональное, программирование, коллекции)
Числа, делящиеся на 3: List(3, 6, 9)

--- REDUCE (агрегация) ---
Сумма всех чисел: 55
Произведение всех чисел: 3628800
Максимальное число: 10

--- Комбинации функций ---
Сумма нечётных чисел, умноженных на 3: 75
Отфильтрованные и форматированные слова: Функциональное Программирование Коллекции Функции

--- Задание 3: Функции как аргументы ---

```

Рис 7. Результаты

The screenshot shows the VS Code interface with the Scala extension installed. The Explorer sidebar shows a project structure with files like `Main.scala`, `MainTest.scala`, and `Lab1.pdf`. The terminal tab displays the output of Scala code execution, specifically demonstrating functional programming concepts such as higher-order functions, currying, and immutability.

```
-- Задание 3: Функции как аргументы --
Функции высшего порядка:
Удвоенные числа: List(2, 4, 6, 8, 10)
Квадраты чисел: List(1, 4, 9, 16, 25)
Чётные числа: List(2, 4)
Сумма: 15
Сумма квадратов чётных чисел: 220

--- Задание 4: Каррирование функций ---
Каррирование функций:
Обычная функция: normalAdd(5, 3, 2) = 10
Каррированная функция: curriedAdd(5)(3)(2) = 10
Частичное применение: addFive(3)(2) = 10
Числа в диапазоне 1-5: List(1, 3)
Числа в диапазоне 10-20: List(15)
Удвоение чисел [1,2,3]: List(2, 4, 6)
Утроение чисел [1,2,3]: List(3, 6, 9)

--- Задание 5: Сравнение коллекций ---
Сравнение мутабельных и иммутабельных коллекций:

--- ИММУТАБЕЛЬНЫЕ коллекции ---
Исходный список: List(1, 2, 3)
После добавления 4: List(1, 2, 3, 4)
Исходный список не изменился: List(1, 2, 3)

--- МУТАБЕЛЬНЫЕ коллекции ---

Signed out no target Ninja 15:12 05/11/2025
```

Рис 8. Результаты

The screenshot shows the VS Code interface with the Scala extension installed. The Explorer sidebar shows a project structure with files like `Main.scala`, `MainTest.scala`, and `Lab1.pdf`. The terminal tab displays the output of Scala code execution, specifically demonstrating mutable vs immutable collections.

```
-- МУТАБЕЛЬНЫЕ коллекции --
Исходный ArrayBuffer: ArrayBuffer(1, 2, 3)
После добавления 4: ArrayBuffer(1, 2, 3, 4)
После изменения первого элемента: ArrayBuffer(100, 2, 3, 4)

--- Сравнение производительности ---
Иммутабельные:
Исходные: List(1, 2, 3, 4, 5)...
После обработки: List(102, 104, 106, 108, 110)...
Исходные не изменились: List(1, 2, 3, 4, 5)...

Исходные не изменились: List(1, 2, 3, 4, 5)...
Исходные не изменились: List(1, 2, 3, 4, 5)...

Мутабельные:
Исходные не изменились: List(1, 2, 3, 4, 5)...
Исходные не изменились: List(1, 2, 3, 4, 5)...

Мутабельные:
Исходные: ArrayBuffer(1, 2, 3, 4, 5)...
Исходные не изменились: ArrayBuffer(1, 2, 3, 4, 5)...

Мутабельные:
Исходные не изменились: List(1, 2, 3, 4, 5)...
Исходные не изменились: List(1, 2, 3, 4, 5)...

Signed out no target Ninja 15:12 05/11/2025
```

Рис 9. Результаты

The screenshot shows the VS Code interface with the Scala extension installed. The Explorer sidebar shows a project structure with files like `.metals`, `.vscode`, `hello-world-template`, `lab2`, `project`, `src`, `main\scala` containing `Main.scala` and `MainTest.scala`, `target`, `.gitignore`, `build.sbt`, `README.md`, `my-project`, `.gitignore`, and `create-scala-project.sh`. The terminal tab is active, displaying the output of Scala code execution. It shows the initial state of a list, its transformation via a map operation, and the resulting state. It also lists the advantages of immutable collections and immutable arrays. The status bar at the bottom right shows the date and time as 05/11/2025.

Рис 10. Результаты

This screenshot is nearly identical to Figure 10, showing the same VS Code environment and Scala code execution results. The terminal output includes the completion message: `[success] Total time: 3 s, completed Nov 5, 2025, 3:07:29 PM nyathi@Moxamed-Idi:/mnt/c/Users/khate/Desktop/Scala/lab2$`.

Рис 11. Результаты

Вывод

В ходе лабораторной работы были успешно освоены основы функционального программирования в Scala, включая работу с коллекциями и функциями высшего порядка. Приобретены практические навыки использования операций map, filter, reduce и каррирования функций.