МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное  
образовательное учреждение высшего образования  
«Самарский национальный исследовательский университет  
имени академика С.П. Королева»  
(Самарский университет)   
  
  
Факультет информатики  
Кафедра программных систем  
  
Дисциплина  
**Логическое программирование  
  
  
  
ОТЧЕТ**по лабораторной работе №6

Тема: Scala. Коллекции. Хвостовая рекурсия.

Студент: Колбанов Д.О.   
Группа: 6301-020302D   
  
Преподаватель: Лобанков А.А.  
  
Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Самара 2024

**Теория**

1. Коллекции.

Scala предоставляет широкий спектр коллекций для работы с данными, в том числе списки (List), массивы (Array), наборы (Set), карты (Map) и другие.

Списки (List) в Scala являются неизменяемыми и представляют упорядоченную коллекцию элементов одного типа. Новые элементы можно добавлять или удалять, создавая новые списки.

Пример: val numbers = List(1, 2, 3, 4, 5)

1. Case-классы.

Case-классы в Scala являются особыми классами, которые предназначены для представления данных.

Они автоматически предоставляют методы доступа к полям, методы toString, equals, hashCode, что делает их удобными для работы с данными.

Пример: case class Person(name: String, age: Int)

1. Работа со списками на Scala используя хвостовую рекурсию.

Хвостовая рекурсия в Scala позволяет оптимизировать рекурсивные вызовы функций, избегая переполнения стека.

При использовании хвостовой рекурсии последняя операция в функции является вызовом самой себя.

Пример:

def sumListTailRecursive(list: List[Int], acc: Int = 0): Int = list match {

case Nil => acc

case head :: tail => sumListTailRecursive(tail, acc + head)

}

1. Условные конструкции if-else.

Условные конструкции if-else в Scala используются для выполнения кода в зависимости от некоторого логического выражения.

Они могут быть вложенными и также могут использоваться в качестве выражений.

Пример: val result = if (x > 10) "Больше 10" else "Меньше или равно 10"

1. Конструкция match-case.

Конструкция match-case в Scala представляет собой альтернативу условной конструкции if-else и позволяет сопоставлять значение с различными шаблонами.

Она может использоваться для управления потоком выполнения программы.

Пример:

val day = 2

val dayType = day match {

case 1 => "Понедельник"

case 2 => "Вторник"

case \_ => "Другой день"

}

1. val и var - переменные:

В Scala val используется для создания неизменяемых (immutable) переменных, которые нельзя переопределять после инициализации.

var используется для создания изменяемых (mutable) переменных, значения которых можно изменять.

Пример: val x = 10 или var y = "Hello"

1. Класс object:

Ключевое слово object в Scala используется для определения объекта (singleton object), который может содержать методы и поля, аналогично статическим членам в Java.

Объекты могут использоваться для организации кода и предоставления глобальных функций или констант.

Пример: object Utils { def formatName(name: String): String = name.toUpperCase }

1. Функции def:

Ключевое слово def используется для определения функций в Scala.

Функции могут быть определены как методы у классов, функциональные значения или быть частью объектов.

Пример: def sum(a: Int, b: Int): Int = a + b

1. Вывод в консоль с помощью println():

Функция println() в Scala используется для вывода данных в консоль.

Она может выводить различные типы данных, такие как строки, числа, списки и другие.

Пример: val message = "Hello, World" println(message)

Эти конструкции помогают писать чистый, модульный и выразительный код на языке программирования Scala. Они также способствуют функциональному программированию и принципам иммутабельности данных.

**Задание**

**Цель и постановка задания**

В процессе выполнения заданий ознакомиться с case-классами и работой со списками на Scala, используя хвостовую рекурсию.

**Задание 1**

Придумайте задание и реализуйте программу на Scala, используя коллекции и хвостовую рекурсию. В качестве предметной области была выбрана предметная область из лабораторной работы номер 3.

**Программа**

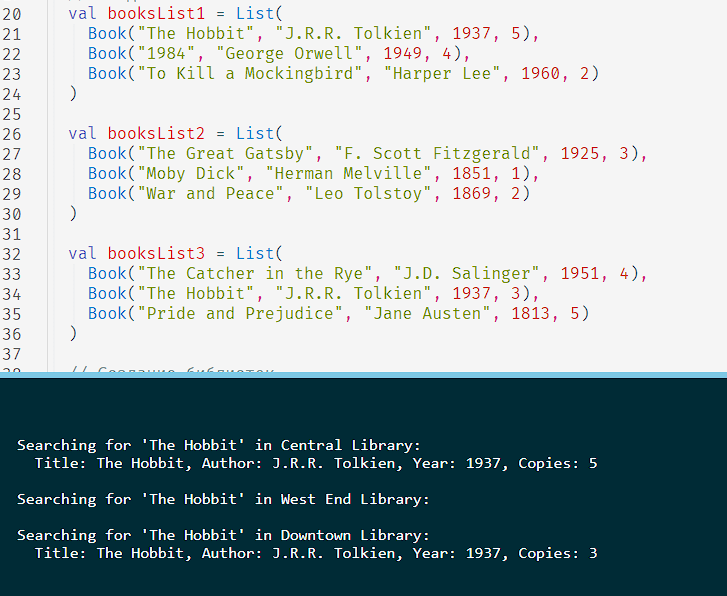
Приведенный код содержит описание двух классов для представления данных о соревнованиях, результатах команд и о самих командах:

* Book представляет информацию о книгах, такую как название, автор, год выпуска, и количество копий;
* Library имеет поля: название и список книг;

Функция findBooksByTitke() находит все книги в библиотеке с заданным названием.

**Результат работы программы**

На рисунке 1 представлен пример вызова функции.

  
Рисунок 1

**Листинг программы**

case class Book(title: String, author: String, year: Int, copies: Int)

class Library(val name: String, val books: List[Book]) {

// Поиск книг с заданным названием с использованием хвостовой рекурсии

def findBooksByTitle(targetTitle: String): List[Book] = {

@annotation.tailrec

def findBooks(remainingBooks: List[Book], foundBooks: List[Book]): List[Book] = remainingBooks match {

case Nil => foundBooks

case head :: tail =>

if (head.title.equalsIgnoreCase(targetTitle)) findBooks(tail, head :: foundBooks)

else findBooks(tail, foundBooks)

}

findBooks(books, Nil)

}

}

object LibraryApp extends App {

// Создание списка книг

val booksList1 = List(

Book("The Hobbit", "J.R.R. Tolkien", 1937, 5),

Book("1984", "George Orwell", 1949, 4),

Book("To Kill a Mockingbird", "Harper Lee", 1960, 2)

)

val booksList2 = List(

Book("The Great Gatsby", "F. Scott Fitzgerald", 1925, 3),

Book("Moby Dick", "Herman Melville", 1851, 1),

Book("War and Peace", "Leo Tolstoy", 1869, 2)

)

val booksList3 = List(

Book("The Catcher in the Rye", "J.D. Salinger", 1951, 4),

Book("The Hobbit", "J.R.R. Tolkien", 1937, 3),

Book("Pride and Prejudice", "Jane Austen", 1813, 5)

)

// Создание библиотек

val library1 = new Library("Central Library", booksList1)

val library2 = new Library("West End Library", booksList2)

val library3 = new Library("Downtown Library", booksList3)

// Список всех библиотек

val libraries = List(library1, library2, library3)

// Поиск всех книг с названием "The Hobbit" в каждой библиотеке

libraries.foreach { library =>

println(s"Searching for 'The Hobbit' in ${library.name}:")

val foundBooks = library.findBooksByTitle("The Hobbit")

foundBooks.foreach(book => println(s" Title: ${book.title}, Author: ${book.author}, Year: ${book.year}, Copies: ${book.copies}"))

println() // Для лучшей читаемости вывода

}

}