ЛАБОРАТОРНАЯ РАБОТА 3. РАБОТА СО СТРОКАМИ

Цель. Изучить технику работы со строками в Scala.

**Краткие теоретические сведения**.

Scala предоставляет богатый набор функций для работы со строками. Вот некоторые расширенные функции, которые можно использовать со строками в Scala:

1. replaceAll: эта функция используется для замены всех вхождений строки другой строкой. Функция принимает два аргумента: первый аргумент — это заменяемое регулярное выражение, а второй аргумент — строка замены.

Пример:

val str = "Hello, World!"

val newStr = str.replaceAll("o", "a")

println(newStr) // "Hella, Warld!"

split: Эта функция используется для разделения строки на массив подстрок на основе разделителя. Функция принимает один аргумент — строку-разделитель.

Пример:

val str = "apple,banana,orange"

val arr = str.split(",")

println(arr.mkString(" ")) // "apple banana orange"

1. startsWith и endsWith:

Эти функции используются для проверки того, начинается или заканчивается строка заданной подстрокой. Функции принимают один аргумент — проверяемую подстроку.

Пример:

val str = "Hello, World!"

println(str.startsWith("Hello")) // true

println(str.endsWith("!")) // true

1. substring: Выделяет подстроку из строки. Пример:

val str = "Hello, World!"

val subStr = str.substring(7, 12)

println(subStr) // "World"

1. toCharArray: Преобразует строку в массив символов

Пример:

val str = "Hello, World!"

val arr = str.toCharArray()

println(arr.mkString(" ")) // "H e l l o , W o r l d !"

1. toLowerCase и toUpperCase: Преобразует символы строки в верхний и нижний регистр соответственно.

Пример:

val str = "Hello, World!"

println(str.toLowerCase) // "hello, world!"

println(str.toUpperCase) // "HELLO, WORLD!"

1. trim: Отсекает концевые пробелы.

Пример:

val str = " Hello, World! "

println(str.trim) // "Hello, World!"

1. indexOf иlastIndexOf: Получает первый и последний индекс подстроки в строке (то есть номер позиции, с которой начинается подстрока).

Пример:

val str = "Hello, World!"

println(str.indexOf("o")) // 4

println(str.lastIndexOf("o")) // 8

1. charAt: Определяет символ, стоящий на указанной позиции.

Пример:

val str = "Hello, World!"

println(str.charAt(7)) // 'W'

1. stripMargin: Удаляет ведущие пробелы перед строкой.

Пример:

val str =

"""

|Hello,

|World!

|""".stripMargin

println(str) // "Hello,\nWorld!\n"

Теперь обратимся к регулярным выражениям.

В Scala регулярные выражения представлены классом scala.util.matching.Regex, который предоставляет множество методов для сопоставления строк и управления ими на основе регулярных выражений.

Вот пример, демонстрирующий некоторые основные функции регулярных выражений в Scala:

val regex = """(\d{3})-(\d{2})-(\d{4})""".r

val str1 = "123-45-6789"

val str2 = "abc-12-3456"

val match1 = regex.findFirstMatchIn(str1)

val match2 = regex.findFirstMatchIn(str2)

match1 match {

case Some(m) => println(s"Match found: ${m.group(0)}")

case None => println("No match found")

}

match2 match {

case Some(m) => println(s"Match found: ${m.group(0)}")

case None => println("No match found")

}

В этом примере мы определяем шаблон регулярного выражения, который соответствует номеру социального страхования в формате XXX-XX-XXXX, где X — цифра. Затем мы пытаемся сопоставить этот шаблон с двумя разными строками: «123-45-6789» и «abc-12-3456».

Метод findFirstMatchIn возвращает объект Option[Match], представляющий первое совпадение шаблона в заданной строке, если таковое имеется. Мы используем сопоставление с образцом, чтобы извлечь совпадающую подстроку из объекта Match и распечатать ее.

Когда мы запускаем этот пример, мы получаем следующий вывод:

Match found: 123-45-6789

No match found

В этом случае первая строка соответствует шаблону регулярного выражения, поэтому мы получаем объект соответствия с совпадающей подстрокой «123-45-6789». Вторая строка не соответствует шаблону, поэтому мы получаем объект None вместо объекта соответствия.

Обратите внимание, что регулярные выражения могут быть довольно мощными и сложными, и в классе Regex доступно гораздо больше функций и методов для работы с ними.

In Scala, you can use the findAllIn method of the Regex class to find all occurrences of a substring in a string using a regular expression. Here's an example:

val str = "The quick brown fox jumps over the lazy dog"

val pattern = "fox".r

val matches = pattern.findAllIn(str)

matches.foreach(println)

In this example, we define a string str containing the text "The quick brown fox jumps over the lazy dog". We then define a regular expression pattern pattern using the .r method, which matches the substring "fox". We use the findAllIn method of the Regex class to find all occurrences of the pattern in the string, and store the matches in the matches variable.

Finally, we loop through the matches variable using the foreach method and print each match to the console.

***Порядок выполнения работы***

1. Изучить теоретическую часть.
2. Получить индивидуальное задание у преподавателя из прилагаемого списка.
3. Создать и отладить приложение в среде Scala.
4. Написать отчет.
5. Защитить работу.

***Индивидуальные задания***

Для всех вариантов:

Вывести суммарное число всех гласных в собственном тексте.

**Вариант 1.**

1. Дан текст: ‘Hello to everybody’. C помощью техники регулярных выражений заменить латинские буквы на русские (или на цифры, если русский шрифт не поддерживается)
2. Найти в тексте “When executing the exercise extract all extra words” все слова, начинающиеся на ext.
3. В тексте ‘A big round ball fall to the ground’ заменить артикль the на a.
4. Записать все слова в тексте в обратном порядке.
5. Дан текст: ‘Hello to everybody’. Выбросить все гласные.
6. Дан текст: ‘Hello to everybody’. Удвоить каждую букву в слове
7. Дан текст: ‘Hello to everybody’. Удалить все вхождения буквы y
8. Дан текст: ‘Hello to everybody’. Вставить слова with heartness чтобы получить

Hello with heartness to everybody

**Вариант 2.**

1. Дан текст: ‘1+1=2’. C помощью техники регулярных выражений заменить цифры на слова: 1- one, 2 - two
2. Найти в тексте ‘Being quiet buetiful girl she never thought of buety’ все слова, начинающиеся на bue.
3. В тексте ‘one plus one is something’ заменить one на two.
4. Поменять местами первое и последнее слово в тексте world is nice.
5. Дан текст: ‘Hello to all my friends’. Выбросить все согласные.
6. Дан текст: ‘Hello to everybody’. Удалить каждую вторую букву в слове
7. Дан текст: ‘Hello to everybody’. Удалить все вхождения буквы e
8. Дан текст: ‘Be healthy’. Вставить слова always чтобы получить

Be always healthy

**Вариант 3.**

1. Дан текст: ‘Passport AB-123-456’. C помощью техники регулярных выражений найти числа в этом тексте
2. Найти в тексте ‘Being strong means nothing’ все слова, заканчивающиеся на ing.
3. В тексте ‘123+723=846’ заменить 3 на 4, 6 – на 8.
4. Записать все слова в тексте в случайном порядке. Текст: big dreadful dog runs after small poor cat. Всех слов 8. Номера 1,2,3, …,8. Заменить слово с номером 1 на слово с номером 7, слово с номером 2 на слово с номером 6, слово с номером 4 на слово с номером 5. Замена обоюдная, то есть слова просто меняются местами.
5. Дан текст: ‘Hello to everybody’. Выбросить каждую третью букву.
6. Дан текст: ‘Hello to everybody’. Удвоить каждую букву в слове
7. Дан текст: ‘Hello to everybody’. Удалить все вхождения буквы o
8. Дан текст: ‘Hello to everybody Katty, Mikky’. Вывести все слова, начинающиеся с заглавной буквы.

**Вариант 4.**

1. Дан текст: ‘Passport AB-123-436’. C помощью техники регулярных выражений найти число вхождений цифры 3.
2. Дан текст: ‘Passport AB-123-436’. C помощью техники регулярных выражений найти сумму всех цифр в тексте.
3. В тексте ‘123+723=846’ выделить все символы арифметических операций.
4. Записать все слова в тексте в случайном порядке. Текст: big black cat runs after small poor catty. Всех слов 8. Номера 1,2,3, …,8. Заменить слово с номером 1 на слово с номером 4, слово с номером 2 на слово с номером 7, слово с номером 3 на слово с номером 6. Замена обоюдная, то есть слова просто меняются местами.
5. Дан текст: ‘Hello to Ronny old nail’. Найти удвоенные вхождения согласных.
6. Дан текст: ‘Hello to Ronny old nail’. Найти число слов, записанных с большой буквы.
7. Дан текст: ‘Hello to everybody’. Подсчитать число всех вхождений буквы o
8. Дан текст: ‘Hello agaiN to everybody Katty, Mikky’. Вывести все слова, начинающиеся со строчной буквы.

Для всех вариантов:

Вывести суммарное число всех гласных в собственном тексте.

***Контрольные вопросы***

1. Что такое регулярное выражение?

Регулярное выражение – это мощный инструмент для работы с текстовыми данными, представляющий собой строку символов, которая описывает шаблон поиска текста.

Основная идея регулярных выражений заключается в создании шаблона, который определяет, какие строки соответствуют этому шаблону, и затем применении этого шаблона к тексту для выполнения соответствующих операций.

1. Приведите несколько шаблонов для регулярных выражений?
2. Как использовать класс StringBuilder – пример?

StringBuilder – это класс, который используется для эффективной конкатенации (соединения) строк. Он предоставляет методы для добавления, удаления и изменения символов внутри строки без создания новых строк.

// Создаем объект StringBuilder

StringBuilder stringBuilder = new StringBuilder();

// Добавляем строки в объект StringBuilder

stringBuilder.append("Привет, ");

stringBuilder.append("мир!");

// Преобразуем объект StringBuilder в строку

String result = stringBuilder.toString();

// Выводим результат

System.out.println(result); // Выведет: Привет, мир!

1. Как вывести на экран коллекцию слов, найденных с помощью регулярного выражения - пример?