Министерство образования Республики Беларусь
Учреждение образования Белорусский государственный университет информатики и радиоэлектроники

Кафедра ЭВМ

Отчёт по лабораторной работе №3
“ Работа со строками ”

Проверил: Выполнил:

Минск 2023

1. Цель

Изучить технику работы со строками в Scala.

2. 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"
```

3. 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
```

4. substring: Выделяет подстроку из строки. Пример:

```
val str = "Hello, World!"
val subStr = str.substring(7, 12)
println(subStr) // "World"
```

5. 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 !"
```

6. 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!"
```

7. trim: Отсекает концевые пробелы.

Пример:

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

8. 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
```

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

Пример:

```
val str = "Hello, World!"
println(str.charAt(7)) // 'W'
```

10. 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.

3. Индивидуальное задание

Вариант 1

1. Дан текст: ‘Hello to everybody’. С 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

Листинг программы:

Код программы::

```
object TextManipulation {
  def main(args: Array[String]): Unit = {
    val text1 = "Hello to everybody"
    val text2 = "When executing the exercise extract all extra words"
    val text3 = "A big round ball fall to the ground"
    val text4 = "A big round ball fall to the ground"
    val text5 = "Hello to everybody"
    val text6 = "Hello to everybody"
    val text7 = "Hello to everybody"
    val text8 = "Hello to everybody"

    val replacedText1 = text1.map {
      case c if c.isLetter && c <= 'z' => ('a' + (c - 'a')).toChar
      case c if c.isLetter && c <= 'Z' => ('A' + (c - 'A')).toChar
      case c => c
    }
    println(s"Задача 1: $replacedText1")

    val extWords = "\\bext\\w*\\b".r.findAllIn(text2).toList
    println(s"Задача 2: $extWords")

    val replacedText3 = text3.replaceAll("the", "a")
    println(s"Задача 3: $replacedText3")

    val reversedWords = text4.split("\\s+").reverse.mkString(" ")
    println(s"Задача 4: $reversedWords")

    val withoutVowels = text5.replaceAll("[aeiouyAEIOU]", "")
    println(s"Задача 5: $withoutVowels")

    val doubledLetters = text6.split("\\s+").map(word => word.map(c =>
c.toString * 2).mkString(" ")).mkString(" ")
```

```
    println(s"Задача 6: $doubledLetters")

    val withoutY = text7.replaceAll("y", "")
    println(s"Задача 7: $withoutY")

    val withHeartness = text8.replace("to", "with heartness to")
    println(s"Задача 8: $withHeartness")
  }
}
```

Главное меню:

1. Дан текст: ‘Hello to everybody’. С помощью техники регулярных выражений заменить латинские буквы на русские (или на цифры, если русский шрифт не поддерживается)

```
val replacedText1 = text1.map {
  case c if c.isLetter && c <= 'z' => ('a' + (c - 'a')).toChar
  case c if c.isLetter && c <= 'Z' => ('A' + (c - 'A')).toChar
  case c => c
}
println(s"Задача 1: $replacedText1")
```

Работа функции:

```
Задача 1: Здлло уо дхдсшбогш
```

2. Найти в тексте “When executing the exercise extract all extra words” все слова, начинающиеся на ext.

```
val extWords = "\\bext\\w*\\b".r.findAllIn(text2).toList
println(s"Задача 2: $extWords")
```

Работа функции:

```
Задача 2: List(extract, extra)
```

3. В тексте ‘A big round ball fall to the ground’ заменить артикль the на a.

```
val replacedText3 = text3.replaceAll("the", "a")
println(s"Задача 3: $replacedText3")
```

Работа функции:

```
Задача 3: A big round ball fall to a ground
```

4. Записать все слова в тексте в обратном порядке.

```
val reversedWords = text4.split("\\s+").reverse.mkString(" ")
println(s"Задача 4: $reversedWords")
```

Работа функции:

```
Задача 4: ground the to fall ball round big A
```

5. Дан текст: ‘Hello to everybody’. Выбросить все гласные.

```
val withoutVowels = text5.replaceAll("[aeiouyAEIOU]", "")
println(s"Задача 5: $withoutVowels")
```

Работа функции:

```
Задача 5: Hll t vrbd
```

6. Дан текст: ‘Hello to everybody’. Удвоить каждую букву в слове

```
val doubledLetters = text6.split("\\s+").map(word => word.map(c => c.toString
* 2).mkString(" ")).mkString(" ")
println(s"Задача 6: $doubledLetters")
```

Работа функции:

```
Задача 6: HH ee ll ll oo tt oo ee vv ee rr yy bb oo dd yy
```

7. Дан текст: ‘Hello to everybody’. Удалить все вхождения буквы y

```
val withoutY = text7.replaceAll("y", "")
println(s"Задача 7: $withoutY")
```

Работа функции:

```
Задача 7: Hello to everbod
```

8. Дан текст: ‘Hello to everybody’. Вставить слова with heartness, чтобы получить: Hello with heartness to everybody

Работа функции:

```
Задача 8: Hello with heartness to everybody
```

Вывод

Познакомились с техникой работы со строками в Scala.