Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**
МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/12 Интеллектуальный анализ больших данных в системах поддержки принятия решений**

О Т Ч Е Т
по лабораторной работе № 7

Название: Строки. Регулярные выражения

Дисциплина: Языки программирования для работы с большими данными

Студент ИУ6-23М (Группа) _____ (Подпись, дата) М.А. Гейне (И.О. Фамилия)

Преподаватель _____ (Подпись, дата) П.В. Степанов (И.О. Фамилия)

Москва, 2023

ЗАДАНИЕ

Вариант 1:

1. В тексте слова заданной длины заменить указанной подстрокой, длина которой может не совпадать с длиной слова.
2. В тексте после k-го символа вставить заданную подстроку.

Вариант 2:

1. Найти, каких букв, гласных или согласных, больше в каждом предложении текста.
2. В стихотворении найти количество слов, начинающихся и заканчивающихся гласной буквой.

Вариант 3:

1. Во всех вопросительных предложениях текста найти и напечатать без повторений слова заданной длины.
2. В каждом предложении текста поменять местами первое слово с последним, не изменяя длины предложения.

Вариант 4:

1. В тексте исключить подстроку максимальной длины, начинающуюся и заканчивающуюся заданными символами.
2. Заменить все одинаковые рядом стоящие символы в тексте одним символом.

Задача 1

Для решения с консоли читается длина слов, которые нужно заменить, и подстрока, на которую будет производиться замена. Далее читается файл с текстом и открывается файл для записи. В регулярное выражение подставляется заданная длина рядом с токеном, обозначающим любую букву, таким образом ограничивая число последовательно встречающихся букв. Перед и после слова ожидаются пробелы. Код решения приведён ниже.

```
import java.io.{FileNotFoundException, IOException, PrintWriter, File}
import scala.io.StdIn
import scala.util.{Try, Success, Failure, Using}
import scala.annotation.tailrec


def test(input: String, output:String): Unit =
  println("Enter word length: ")
  val len = StdIn.readLine().toInt
  println("Enter substitution string: ")
  val sub = StdIn.readLine()

  Using.Manager {use =>
    val source = use(scala.io.Source.fromFile(input))
    val outputFile = new File(output)
    outputFile.getParentFile().mkdirs()
    val writer = use(new PrintWriter(outputFile))
    val text = source.mkString

    val regex = s"\\b\\w{$len}\\b".r
    val replacedText = regex.replaceAllIn(text, sub)
    writer.print(replacedText)
  } match
    case Failure(exception) => println(s"Exeption occured when working with file:
${exception.toString()}")
    case Success(_) => ()
```

Результат решения задачи представлен ниже.

```
Input:
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis dapibus, velit vel
ultricies ultrices, tellus lectus rutrum ex, vel bibendum ipsum elit sed velit.
Sed posuere aliquam nulla, vel ornare mauris lobortis vel. Donec condimentum velit
vitae ex fermentum, vel fringilla quam auctor. Vestibulum bibendum est at nibh
ultricies, eu egestas enim varius. Nulla facilisi. Sed ullamcorper arcu non tellus
venenatis luctus. Proin venenatis mi non magna consequat viverra.

Console:
```

```
Variant 1
Question 4
Enter word length:
5
Enter substitution string:
SUBSTRING
------------------------
Developer: mikeGEINE
Task recieved on: Fri Mar 31 15:05:00 MSK 2023
Task completed (this run) on: Thu May 25 19:44:29 MSK 2023

Output:
SUBSTRING SUBSTRING SUBSTRING sit amet, consectetur adipiscing elit. Duis dapibus,
SUBSTRING vel ultricies ultrices, tellus lectus rutrum ex, vel bibendum SUBSTRING
elit sed SUBSTRING. Sed posuere aliquam SUBSTRING, vel ornare mauris lobortis vel.
SUBSTRING condimentum SUBSTRING SUBSTRING ex fermentum, vel fringilla quam auctor.
Vestibulum bibendum est at nibh ultricies, eu egestas enim varius. SUBSTRING
facilisi. Sed ullamcorper arcu non tellus venenatis luctus. SUBSTRING venenatis mi
non SUBSTRING consequat viverra.
```

Задача 2

С консоли читается позиция вставки и строка для вставки. Исходный текст бьётся в позиции вставки, а затем собирается новая строка, состоящая из части до разбиения, строки вставки и части после разбиения. Решение задачи приводится ниже.

```scala
import java.io.{FileNotFoundException, IOException, PrintWriter, File}
import scala.io.StdIn
import scala.util.{Try, Success, Failure, Using}
import scala.annotation.tailrec

def test(input: String, output:String): Unit =
  println("Enter insertion position: ")
  val pos = StdIn.readLine().toInt
  println("Enter insertion string: ")
  val str = StdIn.readLine()

  Using.Manager {use =>
    val source = use(scala.io.Source.fromFile(input))
    val outputFile = new File(output)
    outputFile.getParentFile().mkdirs()
    val writer = use(new PrintWriter(outputFile))
    val text = source.mkString

    val (pre, post) = text.splitAt(pos)
    val replacedText = s"$pre$str$post"
    writer.print(replacedText)
  } match
```

```
    case Failure(exception) => println(s"Exeption occured when working with file:
${exception.toString()}")
    case Success(_) => ()
```

Результат работы программы приведён ниже.

```
Input:
I have go

Console:
Variant 1
Question 4
Enter insertion position:
7
Enter insertion string:
to
------------------------
Developer: mikeGEINE
Task recieved on: Fri Mar 31 15:05:00 MSK 2023
Task completed (this run) on: Thu May 25 19:44:29 MSK 2023

Output:
I have to go
```

Задание 3

Исходный текст разбивается по символам, которые обычно заканчивают предложения. Далее каждое предложение обрабатывается tail-рекурсивной функцией, которая в каждом предложении регулярным выражением ищет количество гласных символов, а количество согласных определяется путём вычитания количества гласных букв из общего числа букв в предложении. Два значения сравниваются, а результат сравнения записывается в файл. Решение задачи приведено ниже.

```
import java.io.{FileNotFoundException, IOException, PrintWriter, File}
import scala.io.StdIn
import scala.util.{Try, Success, Failure, Using}
import scala.annotation.tailrec


def test(input: String, output:String): Unit =
  Using.Manager {use =>
    val source = use(scala.io.Source.fromFile(input))
    val outputFile = new File(output)
    outputFile.getParentFile().mkdirs()
    val writer = use(new PrintWriter(outputFile))
```

```
    val sentences = source.mkString.split("[.!?]").toList

    @tailrec def counter(sentences: List[String]): Unit =
      val vowels_regex = """[eyuioaEYUIOA]""".r
      sentences match
        case head :: next => {
          val vowels = vowels_regex.findAllIn(head).length
          val consonants = """\w""".r.findAllIn(head).length - vowels
          if vowels > consonants then
            writer.println("vowels")
          else
            writer.println("consonants")

          counter(next)
        }
        case Nil => ()

    counter(sentences = sentences)
  } match
    case Failure(exception) => println(s"Exeption occured when working with file:
${exception.toString()}")
    case Success(_) => ()
```

Результат работы программы приведён ниже.

```
Input:
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis dapibus, velit vel
ultricies ultrices, tellus lectus rutrum ex, vel bibendum ipsum elit sed velit.
Sed posuere aliquam nulla, vel ornare mauris lobortis vel. Donec condimentum velit
vitae ex fermentum, vel fringilla quam auctor. Vestibulum bibendum est at nibh
ultricies, eu egestas enim varius. Nulla facilisi. Sed ullamcorper arcu non tellus
venenatis luctus. Proin venenatis mi non magna consequat viverra.

Aeyutgs uionas. Oiaert Yupose.

Output:
consonants
consonants
consonants
consonants
consonants
consonants
consonants
consonants
vowels
vowels
```

Задача 4

Для решения задачи записывается регулярное выражение, в котором первым символом является пробел, затем прописная или строчная гласная буква, затем идёт произвольное количество любых букв, далее одна строчная гласная и снова пробел. В тексте ищутся все непересекающиеся совпадения, подсчитывается их количество и записывается в файл вывода. Код задачи приведён ниже.

```
import java.io.{FileNotFoundException, IOException, PrintWriter, File}
import scala.io.StdIn
import scala.util.{Try, Success, Failure, Using}
import scala.annotation.tailrec

def test(input: String, output:String): Unit =
  Using.Manager {use =>
    val source = use(scala.io.Source.fromFile(input))
    val outputFile = new File(output)
    outputFile.getParentFile().mkdirs()
    val writer = use(new PrintWriter(outputFile))
    val text = source.mkString

    val regex = s"\\b[eyuioaEYUIOA]\\w*[eyuioa]\\b".r
    writer.print(regex.findAllIn(text).length)
  } match
    case Failure(exception) => println(s"Exeption occured when working with file:
${exception.toString()}")
    case Success(_) => ()
```

Результат работы программы приведён ниже.

```
Input:
A language for the code-minded,
Scala's elegance can't be denied,
With syntax crisp and refined,
It's a language for the refined.

Concise and powerful,
Scala's features are bountiful,
From functional to object-oriented,
It's a language that's truly reinvented.

With immutability at its core,
Scala helps you write code that's secure,
From pattern matching to type inference,
It's a language that's truly a difference.

Efficient and expressive,
Scala's power is truly impressive,
```

```
From Spark to Akka,
It's a language that won't be slacka.

So if you want to code with style,
Scala's the language that will beguile,
From the JVM to the world at large,
It's a language that's truly in charge.

Output:
9
```

Задача 5

Для решения задачи в тексте сначала находятся все вопросительные предложения, т.е. последовательности символов, начинающиеся с заглавной буквы, содержащие произвольное количество знаков и пробелов и заканчивающиеся знаком вопроса. Далее для поиска слов заданной длины используется регулярное выражение из задачи 1, а к полученному списку применяется фильтр `distinct`. Код задачи приведён ниже.

```
import java.io.{FileNotFoundException, IOException, PrintWriter, File}
import scala.io.StdIn
import scala.util.{Try, Success, Failure, Using}
import scala.annotation.tailrec


def test(input: String, output:String): Unit =
  println("Enter word length: ")
  val len = StdIn.readLine().toInt

  Using.Manager {use =>
    val source = use(scala.io.Source.fromFile(input))
    val outputFile = new File(output)
    outputFile.getParentFile().mkdirs()
    val writer = use(new PrintWriter(outputFile))
    val text = source.mkString
    val questions = """[A-Z][\w\s,']+?\?""".r.findAllIn(text).toList

    @tailrec def counter(sentences: List[String]): Unit =
      sentences match
        case head :: next => {
          val words = s"\\b\\w{$len}\\b".r.findAllIn(head).distinct.toList
          writer.println(words)

          counter(next)
        }
```

```
        case Nil => ()

    counter(sentences = questions)
  } match
    case Failure(exception) => println(s"Exeption occured when working with file:
${exception.toString()}")
    case Success(_) => ()
```

Результат работы программы приведён ниже (длина слов равна 4).

```
Input:
Okay, let's try this. This is a text without repetitions, right?
Right, text! Here are some some repetitions in text, in text?

Output:
List(This, text)
List(Here, some, text)
```

Задача 6

Для решения задачи текст разбивается на предложения регулярным выражением, использовавшимся в прошлой задаче, но в конце предложения ожидается не только знак вопроса, но и точка с восклицательным знаком. Далее в каждом предложении регулярным выражением выбирается первое слово как последовательность символов без разделителей, затем находится последнее слово, т.е. последовательность букв с символом окончания предложения в конце. Далее эти фрагменты заменяют друг друга, не затрагивая знаки препинания в конце. Код задачи приведён ниже.

```
import java.io.{FileNotFoundException, IOException, PrintWriter, File}
import scala.io.StdIn
import scala.util.{Try, Success, Failure, Using}
import scala.annotation.tailrec

def test(input: String, output:String): Unit =
  Using.Manager {use =>
    val source = use(scala.io.Source.fromFile(input))
    val outputFile = new File(output)
    outputFile.getParentFile().mkdirs()
    val writer = use(new PrintWriter(outputFile))
    val text = source.mkString
    val sentences = """[A-Z][\w\s,']+[.!?]""".r.findAllIn(text)

    val swappedSentences = sentences.map(sentence => {
      val first = """^\w+""".r.findFirstIn(sentence).get
```

```
      val last = """\w+(?=[.!?])""".r.findFirstIn(sentence).get

      """^\w+""".r.replaceFirstIn(sentence, last).replaceFirst("""\w+(?=[.!?])""",
first)
    })

    val result = swappedSentences.mkString(" ")
    writer.print(result)
  } match
    case Failure(exception) => println(s"Exeption occured when working with file:
${exception.toString()}")
    case Success(_) => ()
```

Результат работы программы приведён ниже.

```
Input:
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis dapibus, velit vel
ultricies ultrices, tellus lectus rutrum ex, vel bibendum ipsum elit sed velit.
Sed posuere aliquam nulla, vel ornare mauris lobortis vel. Donec condimentum velit
vitae ex fermentum, vel fringilla quam auctor. Vestibulum bibendum est at nibh
ultricies, eu egestas enim varius. Nulla facilisi. Sed ullamcorper arcu non tellus
venenatis luctus. Proin venenatis mi non magna consequat viverra.

Output:
elit ipsum dolor sit amet, consectetur adipiscing Lorem.velit dapibus, velit vel
ultricies ultrices, tellus lectus rutrum ex, vel bibendum ipsum elit sed Duis.vel
posuere aliquam nulla, vel ornare mauris lobortis Sed.auctor condimentum velit
vitae ex fermentum, vel fringilla quam Donec.varius bibendum est at nibh
ultricies, eu egestas enim Vestibulum.facilisi Nulla.luctus ullamcorper arcu non
tellus venenatis Sed.viverra venenatis mi non magna consequat Proin.
```

Задача 7

Для решения задачи с консоли считываются символы, которые должны открывать и закрывать последовательность. Регулярным выражением ищется максимально длинная последовательность символов. Максимальная длина обеспечивается символом *, который производит “жадную” выборку. Код задачи приведён ниже.

```
import java.io.{FileNotFoundException, IOException, PrintWriter, File}
import scala.io.StdIn
import scala.util.{Try, Success, Failure, Using}
import scala.annotation.tailrec


def test(input: String, output:String): Unit =
  println("Enter symbols: ")
  val sep = StdIn.readLine()
```

```
  Using.Manager {use =>
    val source = use(scala.io.Source.fromFile(input))
    val outputFile = new File(output)
    outputFile.getParentFile().mkdirs()
    val writer = use(new PrintWriter(outputFile))
    val text = source.mkString
    val excluded = s"""$sep.*$sep""".r.replaceAllIn(text, "")

    writer.print(excluded)
  } match
    case Failure(exception) => println(s"Exeption occured when working with file:
${exception.toString()}")
    case Success(_) => ()
```

Результат работы программы приведён ниже (удаление последовательности между символами “te”.

```
Input:
Okay, let's try this. This is a text without repetitions, right? Right, text! Here
are some some repetitions in text, in text?

Output:
Okay, let's try this. This is a xt?
```

Задача 4

Для решения задачи в регулярном выражении ищется символ, повторяющийся более 1 раза. Этот символ помещается в отдельный результат поиска регулярным выражением. Далее вся найденная последовательность замещается этим отдельным результатом поиска. Код задачи приведён ниже.

```
import java.io.{FileNotFoundException, IOException, PrintWriter, File}
import scala.io.StdIn
import scala.util.{Try, Success, Failure, Using}
import scala.annotation.tailrec


def test(input: String, output:String): Unit =
  Using.Manager {use =>
    val source = use(scala.io.Source.fromFile(input))
    val outputFile = new File(output)
    outputFile.getParentFile().mkdirs()
    val writer = use(new PrintWriter(outputFile))
    val text = source.mkString

    val pattern = "(.)\\1+".r
```

```
    val result = pattern.replaceAllIn(text, "$1")

    writer.print(result)
  } match
    case Failure(exception) => println(s"Exeption occured when working with file:
${exception.toString()}")
    case Success(_) => ()
```

Результат работы программы приведён ниже.

```
Input:
Looorem ipsssum dolllor sit ametttt. Consectettttur adipiscinng elit. Seddd vel
massa augue. Namm pellentesque mi ut odiiio vestibulum, ut varius justo eleifend?
Integerr eu dolllor id libero vestibulum tempus! Duis malesuada ultriciesss odio,
id finibus libero congue necc.

Output:
Lorem ipsum dolor sit amet. Consectetur adipiscing elit. Sed vel masa augue. Nam
pelentesque mi ut odio vestibulum, ut varius justo eleifend? Integer eu dolor id
libero vestibulum tempus! Duis malesuada ultricies odio, id finibus libero congue
nec.
```

ВЫВОДЫ

Изучены способы работы с регулярными выражениями в Scala.

Изучены способы выбора в тексте слов, предложений, последовательностей символов.

Изучены способы замены найденных совпадений с регулярными выражениями.

Изучены способы выборки части совпадения с регулярным выражением.