БГУИР

Кафедра ЭВМ

Отчет по лабораторной работе № 4

Тема: «Работа с пакетом SPARK»

Выполнили:

студенты гр. №150502

Альхимович Н.Г.

Скалозуб К.А.

Проверила:

Герман Ю.О.

Минск

2023

**1 ЦЕЛЬ РАБОТЫ**

Изучить технику обработки текста в SPARK Scala.

**2 КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ**

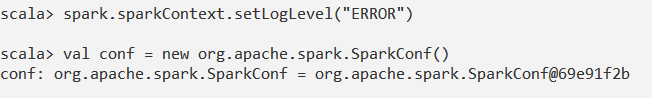
Создаем журнал регистрации ошибок:

spark.sparkContext.setLogLevel("ERROR")

Далее создаем объект-конференции:

scala> val conf = new org.apache.spark.SparkConf()

conf: org.apache.spark.SparkConf = org.apache.spark.SparkConf@2a60cd41



Читаем текстовый файл:

scala> val lines = spark.read.textFile("c:\\Users\\kseni\\lab4.txt")

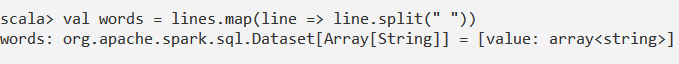
lines: org.apache.spark.sql.Dataset[String] = [value: string]



Получаем массив всех строк:

scala> val words = lines.map(line => line.split(" "))

words: org.apache.spark.sql.Dataset[Array[String]] = [value: array<string>]



scala> val wordCounts = words.map(wordArray => wordArray.length)

wordCounts: org.apache.spark.sql.Dataset[Int] = [value: int]

Выводим предложение, содержащее не менее трех слов:

scala> val longestSentence = lines.filter(line => line.split(" ").length >= 3).first()

longestSentence: String = But eyes are blind. You have to look with the heart.



Выводим весь текст:

scala> val longestSentence2 = lines.foreach(line => println(line))

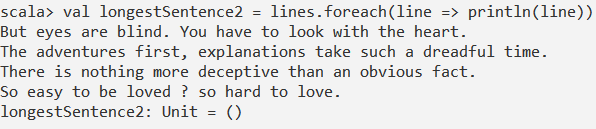
But eyes are blind. You have to look with the heart.

The adventures first, explanations take such a dreadful time.

There is nothing more deceptive than an obvious fact.

So easy to be loved ? so hard to love.

longestSentence2: Unit = ()



Расщепляем текст на отдельные слова.При этом используем регулярное выражение\\W+ (означает любую последовательность символов):

scala> val wordsRDD = lines.flatMap(line => line.split("\\W+"))

wordsRDD: org.apache.spark.sql.Dataset[String] = [value: string]



Слова приводим к нижнему регистру:

scala> val lowercaseWordsRDD = wordsRDD.map(word => word.toLowerCase())

lowercaseWordsRDD: org.apache.spark.sql.Dataset[String] = [value: string]



Создаем множество stop-слов:

scala> val stopWords = Set("a", "an", "the", "and", "but", "or", "for", "of")

stopWords: scala.collection.immutable.Set[String] = Set(for, but, a, or, an, of, and, the)



Удаляем стоп-слова из текста:

scala> val filteredWordsRDD = lowercaseWordsRDD.filter(word => !stopWords.contains(word))

filteredWordsRDD: org.apache.spark.sql.Dataset[String] = [value: string]



Выводим отфильтрованное множество слов:

scala> filteredWordsRDD.collect().foreach(println)

eyes

are

blind

you

have

to

look

with

heart

adventures

first

explanations

take

such

dreadful

time

there

is

nothing

more

deceptive

than

obvious

fact

so

easy

to

be

loved

so

hard

to

love



Отфильтруем слова, начинающиеся на s:

scala> val filteredWordsRDD2 = wordsRDD.filter(word => word.startsWith("s"))

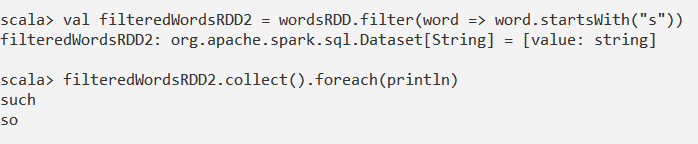
filteredWordsRDD2: org.apache.spark.sql.Dataset[String] = [value: string]

Выведем их:

scala> filteredWordsRDD2.collect().foreach(println)

such

so



Создадим паттерн:

scala> val myPattern = "^d.\*p\*e$"

myPattern: String = ^d.\*p\*e$



Этот паттерн определяет все слова, начинающиеся на d, содержащие где-то посередине p и заканчивающиеся на e.

Сформируем отфильтрованное множество по паттерну:

scala> val filteredWordRDD4 = filteredWordsRDD.filter(word => word.matches(myPattern))

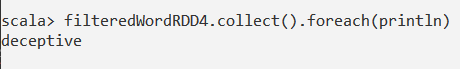
filteredWordRDD4: org.apache.spark.sql.Dataset[String] = [value: string]



Выведем результат на консоль:

scala> filteredWordRDD4.collect().foreach(println)

deceptive



**3 ХОД РАБОТЫ**

**Вариант 1**

1. Вывести все слова из текстового файла, исключая stop-слова.

scala> val text = spark.read.textFile("/Users/nina/Library/CloudStorage/OneDrive-Personal/Документы/Uni/ФПрогр/Lab4.txt")

text: org.apache.spark.sql.Dataset[String] = [value: string]

scala> val words = text.flatMap(line => line.split("\\W+"))

words: org.apache.spark.sql.Dataset[String] = [value: string]

scala> val lowercaseWords = words.map(word => word.toLowerCase())

lowercaseWords: org.apache.spark.sql.Dataset[String] = [value: string]

scala> val stopWords = Set("a", "an", "the", "and", "but", "or", "for", "of")

stopWords: scala.collection.immutable.Set[String] = Set(for, but, a, or, an, of, and, the)

scala> val textWithoutStopOnes = lowercaseWords.filter(word => !stopWords.contains(word))

textWithoutStopOnes: org.apache.spark.sql.Dataset[String] = [value: string]

scala> val printedLyrics = text.foreach(line => println(line))

I know it is going to be lonely,

because everyone keeps turning me down.

Countless new surroundings,

cold eyes keep looking me down.

I am still in the crowd, alien of the town.

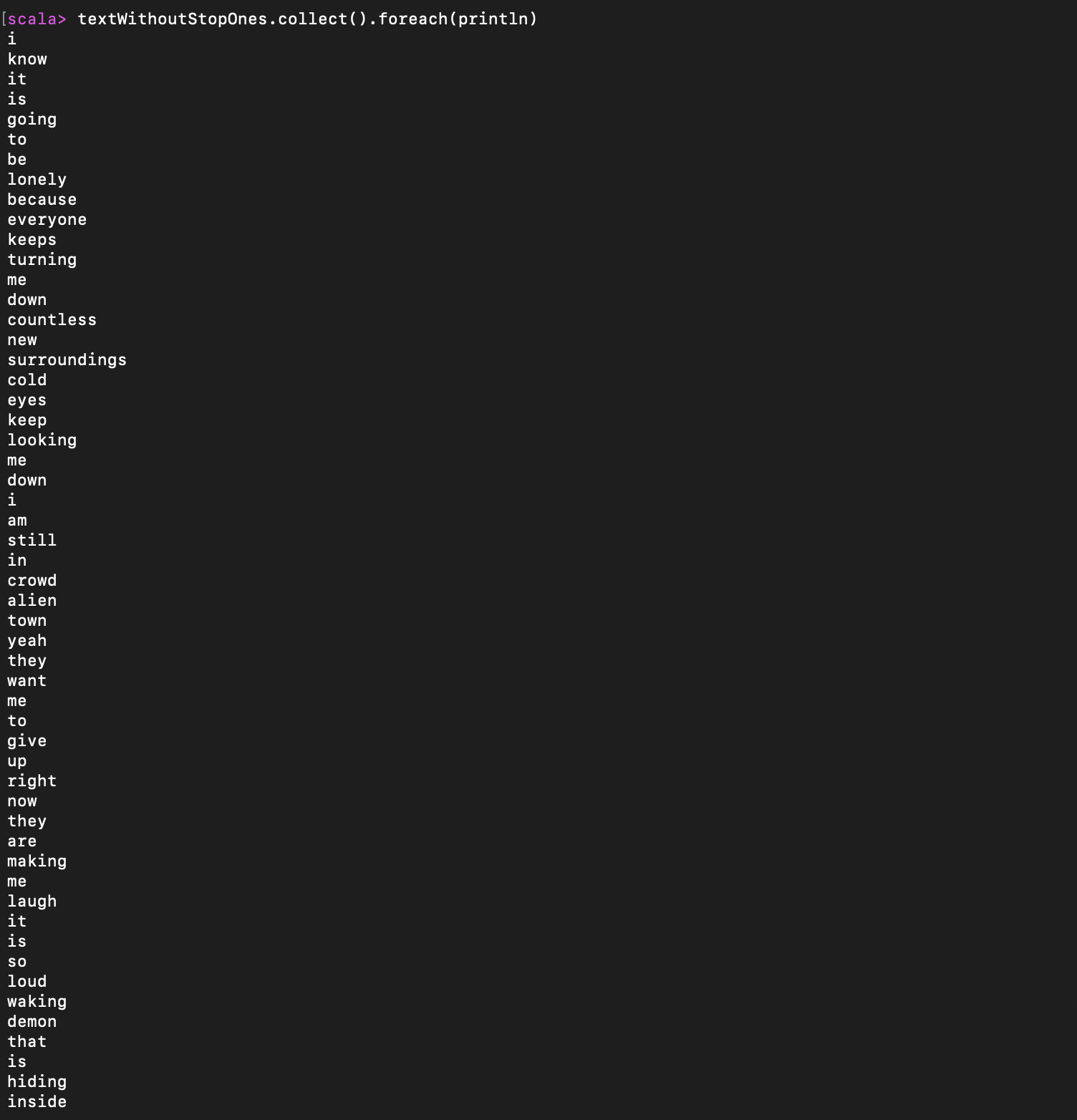
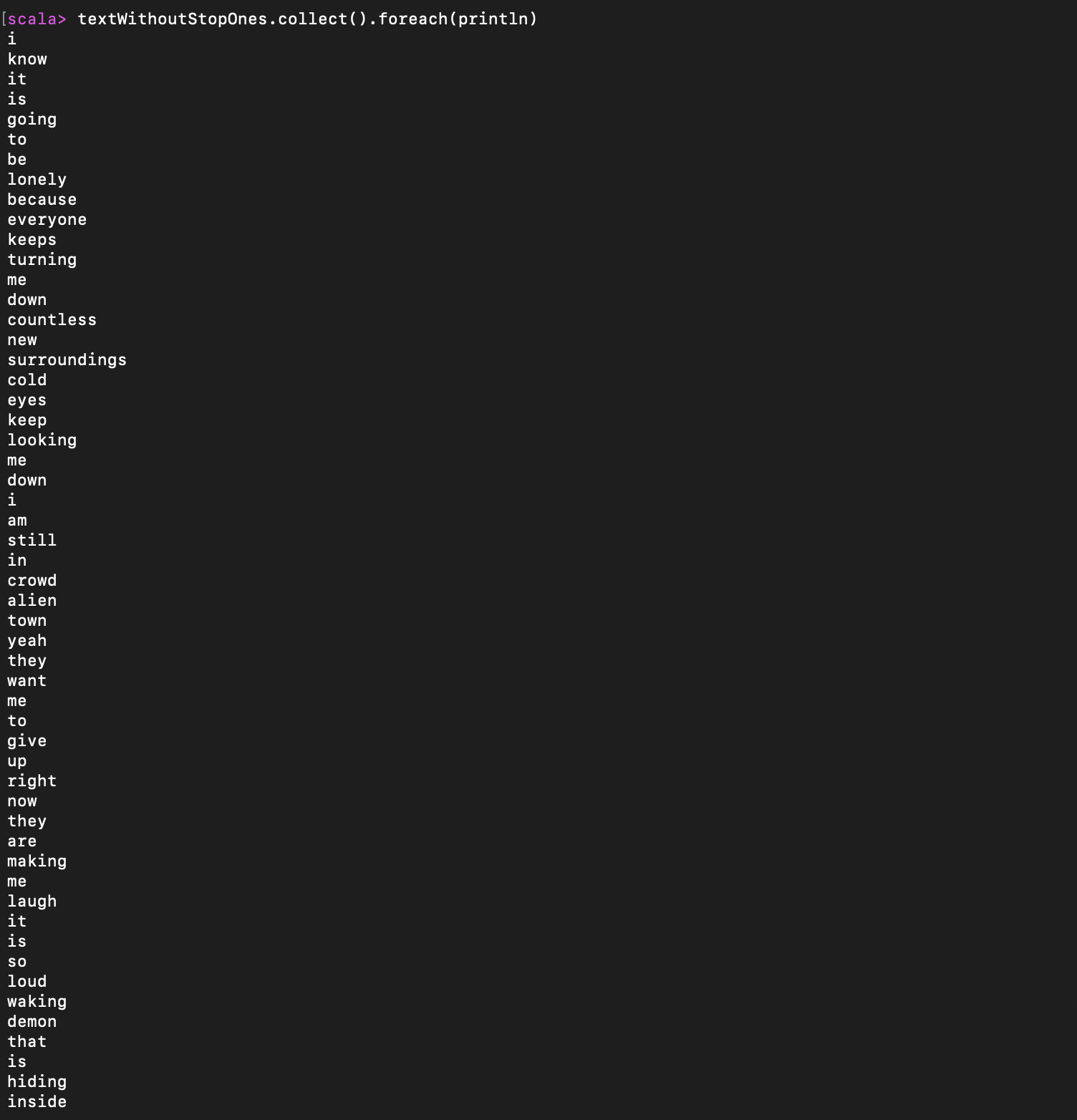
Yeah, they want me to give up right now.

They are making me laugh, it is so loud,

waking the demon that is hiding inside.

printedLyrics: Unit = ()

scala> textWithoutStopOnes.collect().foreach(println)

1. Вывести все слова, содержащие букву t.

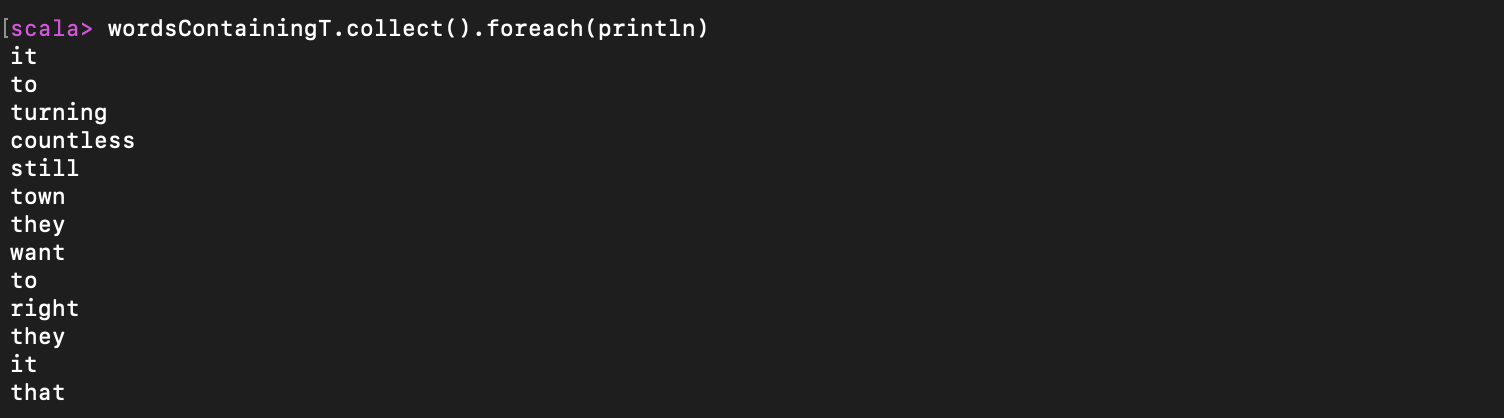
scala> val patternWithT = "t"

patternWithT: String = t

scala> val wordsContainingT = textWithoutStopOnes.filter(word => word.contains(patternWithT))

wordsContainingT: org.apache.spark.sql.Dataset[String] = [value: string]

scala> wordsContainingT.collect().foreach(println)

****

1. Вывести все слова, заканчивающиеся на ing.

scala> val patternEndingIng = ".\*ing$"

patternEndingIng: String = .\*ing$

scala> val wordsIng = textWithoutStopOnes.filter(word => word.matches(patternEndingIng))

wordsIng: org.apache.spark.sql.Dataset[String] = [value: string]

scala> wordsIng.collect().foreach(println)



1. Вывести все слова, вторая буква которых “а”.

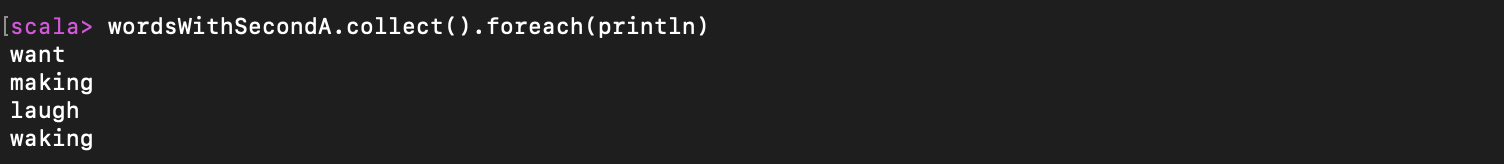
scala> val patternSecondLetterA = "^.[aA].\*$"

patternSecondLetterA: String = ^.[aA].\*$

scala> val wordsWithSecondA = textWithoutStopOnes.filter(word => word.matches(patternSecondLetterA))

wordsWithSecondA: org.apache.spark.sql.Dataset[String] = [value: string]

scala> wordsWithSecondA.collect().foreach(println)



1. Вывести все слова, последняя буква которых s.

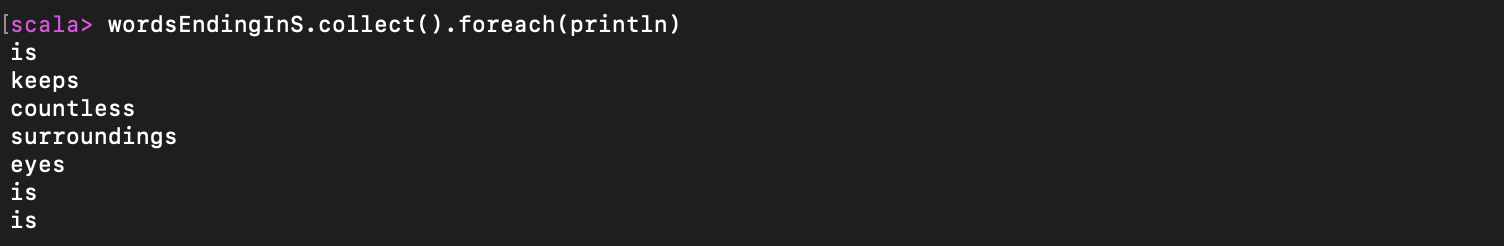
scala> val patternLastS = ".\*[sS]$"

patternLastS: String = .\*[sS]$

scala> val wordsEndingInS = textWithoutStopOnes.filter(word => word.matches(patternLastS))

wordsEndingInS: org.apache.spark.sql.Dataset[String] = [value: string]

scala> wordsEndingInS.collect().foreach(println)



1. Вывести каждое второе слово.

scala> val wordsWithInds = textWithoutStopOnes.rdd.zipWithIndex().map {

| case (word, index) => (word, index.toInt)

| }

wordsWithInds: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[18] at map at <console>:23

scala> val wordsAtEvenPositions = wordsWithInds.filter {

| case (\_, index) => index % 2 == 1

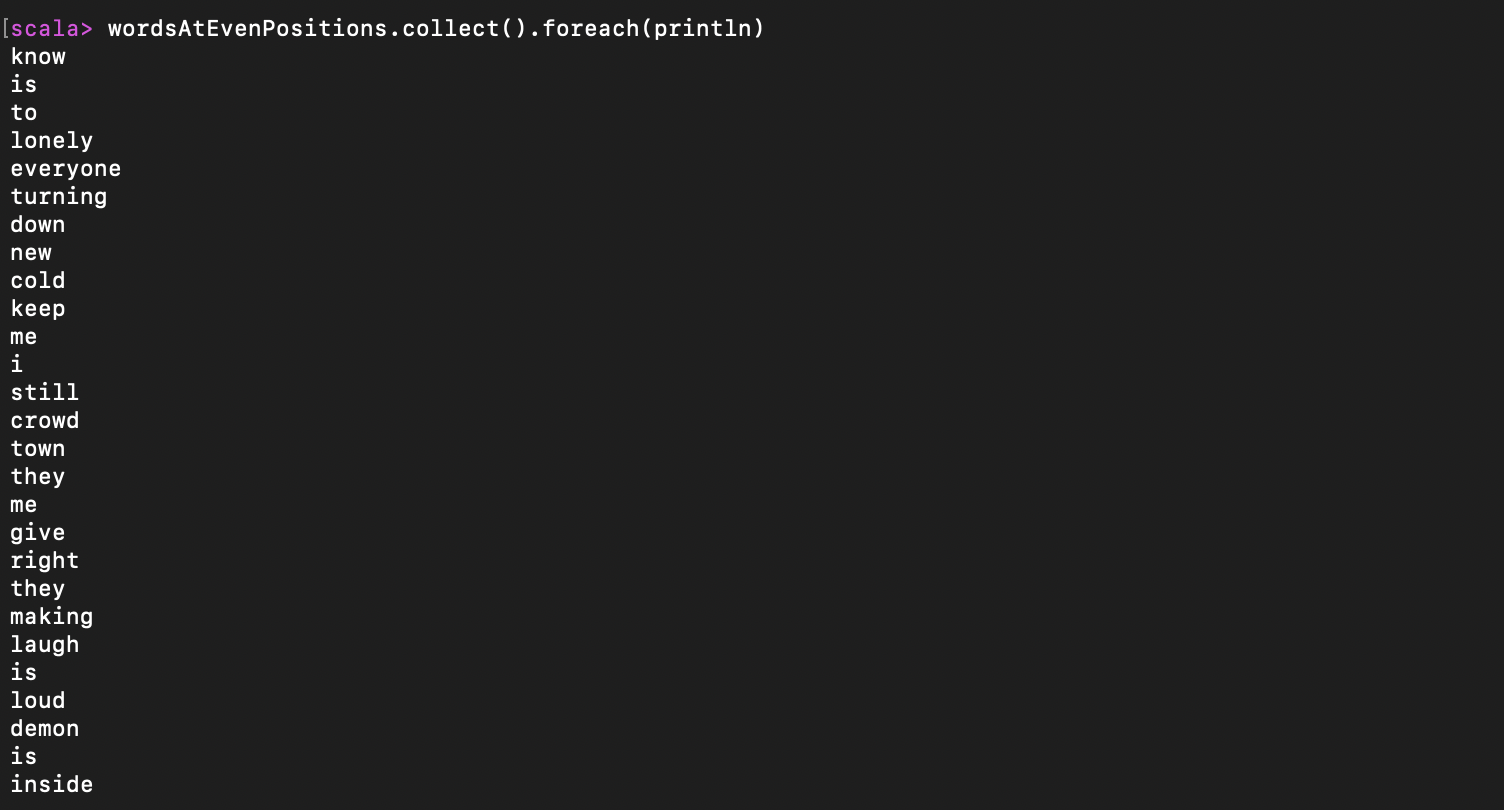
| }.map {

| case (word, \_) => word

| }

wordsAtEvenPositions: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[20] at map at <console>:25

scala> wordsAtEvenPositions.collect().foreach(println)



**4 ВЫВОДЫ**

В процессе выполнения лабораторной работы была изучена техника работы с пакетом Spark для Scala и на основе полученных знаний написан ряд простых S-проектов, осуществляющих обработку текстовых данных.