Министерство образования Республики Беларусь

Учреждение высшего образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

ОТЧЕТ

по лабораторной работе №4

на тему

РАБОТА С ПАКЕТОМ SPARK

|  |  |
| --- | --- |
| Студенты группы 150503:  Проверила: | Ходосевич М.А.  Семков А. Д.  Герман Ю. О. |

Минск 2023

1. Цель работы

Изучить технику обработки текста в SPARK Scala.

1. Краткие теоретические сведения

Команда для чтения текстового файла и ее работа приведены на рисунке 2.1. Путь к файлу прописан полный, строки разделяются знаком «\\».

val lines = spark.read.textFile("C:\\BSUIR\\3course\\fprogr\\laba4\\sport.txt")

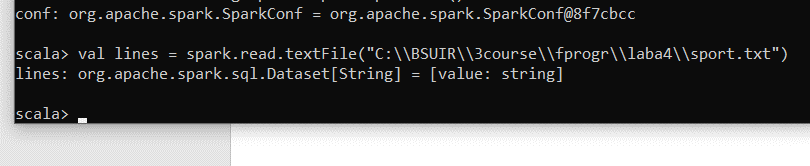


Рисунок 2.1 – Чтение текстового файла

Далее на рисунке 2.2 приведена команда получения массива всех строк и результат ее работы.

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

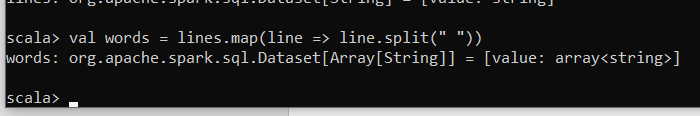


Рисунок 2.2 – Получение массива строк

Ниже приведены строки, которые реализуют вывод первого предложения, содержащего не менее 3 слов. Результат работы приведен на рисунке 2.3.

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

val maxWordCount = wordCounts.max()

val longestSentence = lines.filter(line => line.split(“ ”).length >= 3).first()

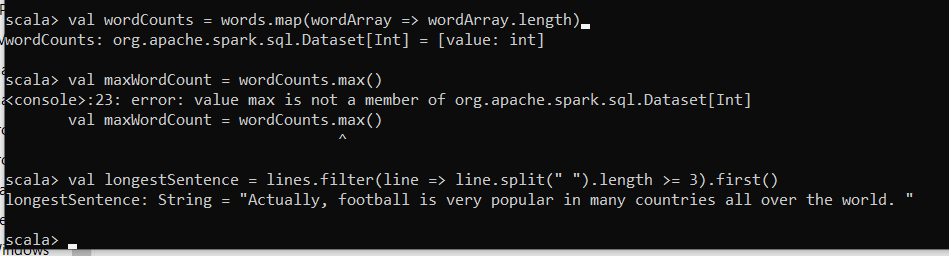


Рисунок 2.3 – Результат работы

Вывод строк из текстового файла(рис. 2.4):

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

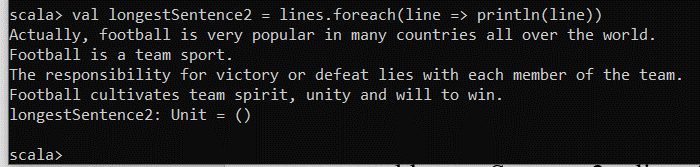


Рисунок 2.4 – Вывод строк из файла

Далее продолжаем работу с этим же текстом. Расщепляем текст на отдельные слова. При этом используется регулярное выражение W+ (означает любую последовательность символов).

val wordsRDD = lines.flatMap(line => line.split(“\\W+”))

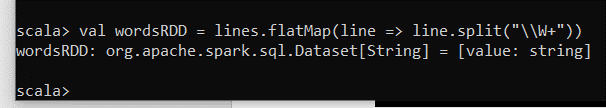


Рисунок 2.5 – Расщепление текста на слова

Приведение всех слов к нижнему регистру (рис. 2.6):

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

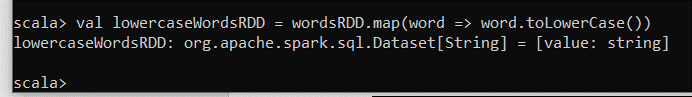


Рисунок 2.6 – Приведение слов к нижнему регистру

Создание множества стоп-слов (рис. 2.7):

val stopWords = Set(“is”, “the”, “of”, “and”)

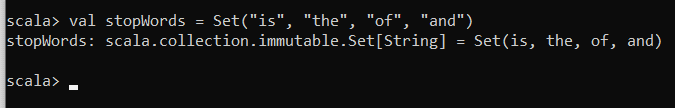


Рисунок 2.7 – Создание множества стоп-слов

Далее удаляем стоп-слова из списка и выводим отфильтрованное множество слов. Результат работы приведен на рисунке 2.8.

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

filteredWordsRDD.collect().foreach(println)

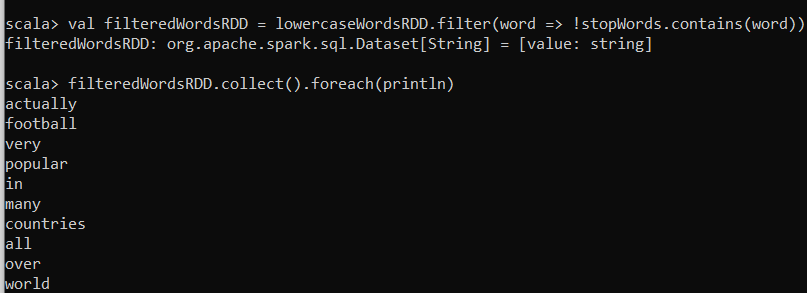


Рисунок 2.8 – Вывод слов после удаления

Далее отфильтруем слова, которые начинаются на “w” и выведем их. Результат работы приведен на рисунке 2.9.

val filteredWordsRDD2 = wordsRDD.filter(word => word.startsWith(“w”))

filteredWordsRDD2.collect().foreach(println)

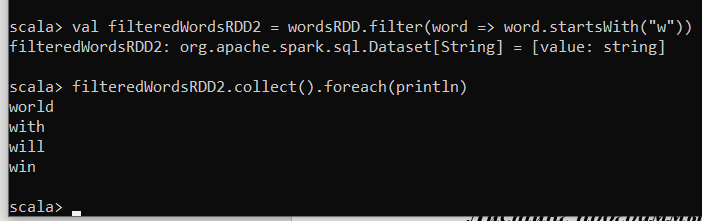


Рисунок 2.9 – Результат работы

Пример работы с паттерном. Первая строчка создает паттерн. Этот паттерн определяет все слова, начинающиеся на w, содержащие где-то посередине r и заканчивающиеся на d. Сформируем отфильтрованное множество по паттерну и выведем результат на экран. Результат работы приведен на рисунке 2.10.

val pattern = “^w.\*r\*d$”

val filteredWordsRDD3 = filteredWordsRDD.filter(word => word.matches(pattern))

filteredWordsRDD3.collect().foreach(println)

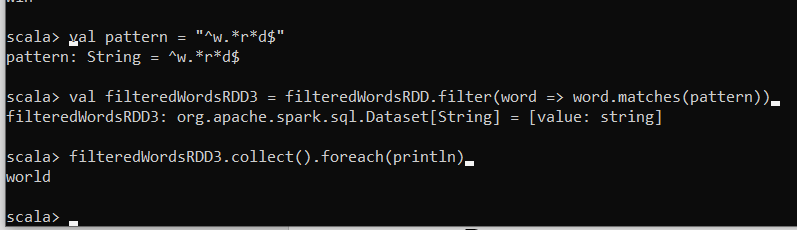


Рисунок 2.10 – Работа с паттерном

1. Ход работы

**Вариант 2.**

Создать собственный текстовый файл на английском или немецком языке – **4-5** предложений. Тема – программирование

***Текст для выполнения заданий:***

In the world of programming, creativity knows no bounds.

From crafting elegant algorithms to designing beautiful user interfaces, every line of code tells a story.

Developers are the architects of the digital age, shaping the future one keystroke at a time.

Collaboration and innovation are the cornerstones of successful projects, as teams work together to turn ideas into reality.

With a strong foundation in programming, the possibilities program are endless.heello

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

val data = spark.read.textFile("src/main/scala/task1/text")

val stopWord = Set("world","In","the","From")

val wordsRDD = data.flatMap(l => l.split("\\s"))

val filterData = wordsRDD.filter(word => !stopWord.contains(word))

filterData.foreach(el => println(el))

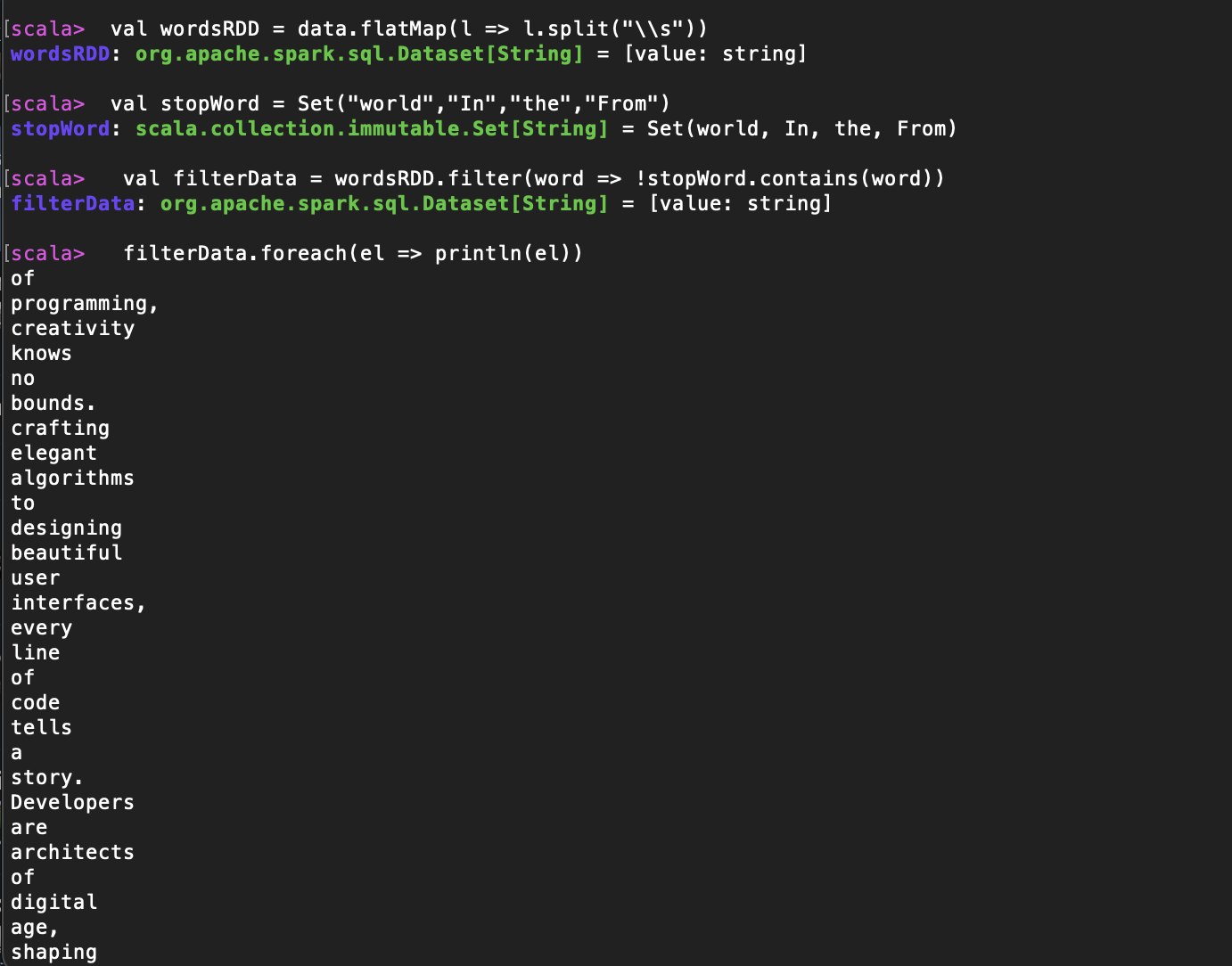


Рисунок 3.1 – Результат работы задания 1

1. Вывести все слова, содержащие вхождение prog

val data = spark.read.textFile("src/main/scala/task1/text")

val wordsRDD = data.flatMap(l => l.split("\\s"))

val filterData = wordsRDD.filter(word => word.startsWith("prog"))

filterData.foreach(el => println(el))

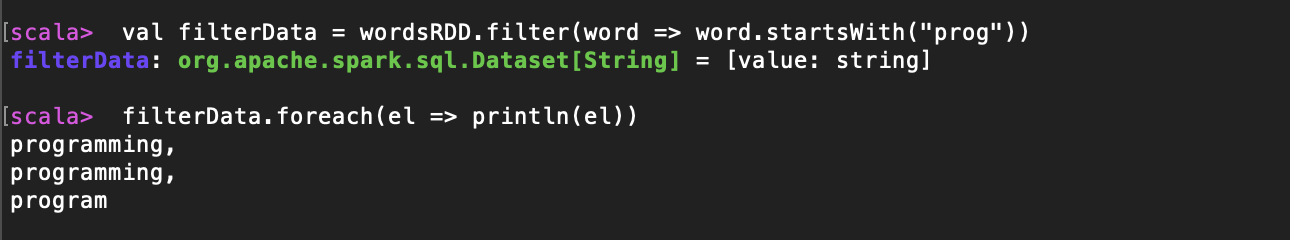


Рисунок 3.2 – Результат работы задания 2

1. Вывести все слова, заканчивающиеся на ion

val data = spark.read.textFile("src/main/scala/task1/text")

val wordsRDD = data.flatMap(l => l.split("\\s")

val filterData = wordsRDD.filter(word => word.endsWith("ion"))

filterData.foreach(el => println(el))

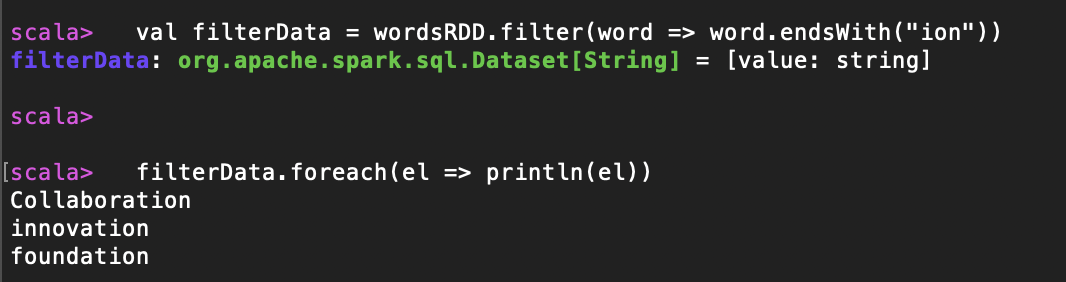


Рисунок 3.3 – Результат работы задания 3

1. Вывести все слова третья буква которых r

val data = spark.read.textFile("src/main/scala/task1/text")

val wordsRDD = data.flatMap(l => l.split("\\s"))(encoder)

val filterData = wordsRDD.filter(word => word.length > 3 && word.charAt(2) == 'r' )

if(!filterData.isEmpty) {

filterData.foreach(el => println(el))

}

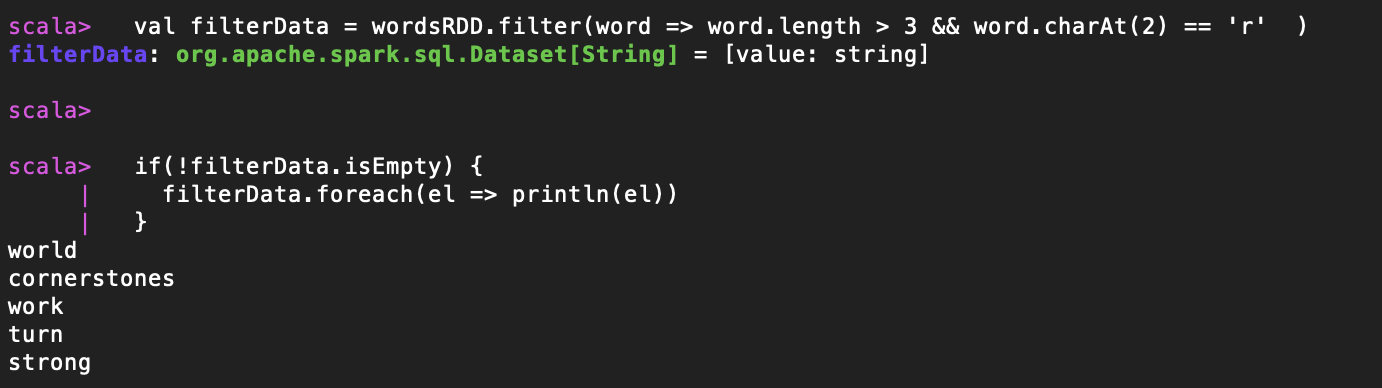


Рисунок 3.4 – Результат работы задания 4

1. Вывести все слова, содержащие как минимум две буквы e

val data = spark.read.textFile("src/main/scala/task1/text")

val wordsRDD = data.flatMap(l => l.split("\\s"))(encoder)

val filterData = wordsRDD.filter(word => word.contains("ee"))

if (!filterData.isEmpty) {

filterData.foreach(el => println(el))

}

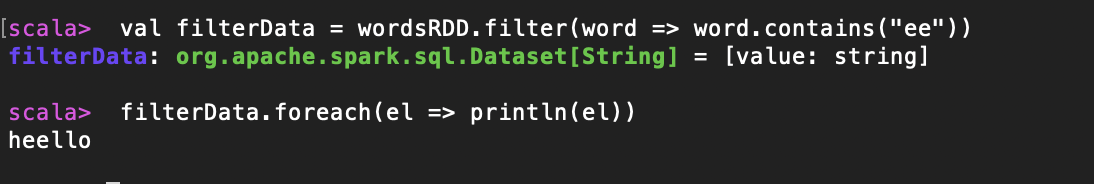


Рисунок 3.5 – Результат работы задания 5

1. Вывести последнее слово

val data = spark.read.textFile("src/main/scala/task1/text")

val wordsRDD = data.flatMap(l => l.split("\\s"))

val lastWord = wordsRDD.rdd.collect()

if(lastWord.length >= 1){

println(lastWord(lastWord.length - 1))

}

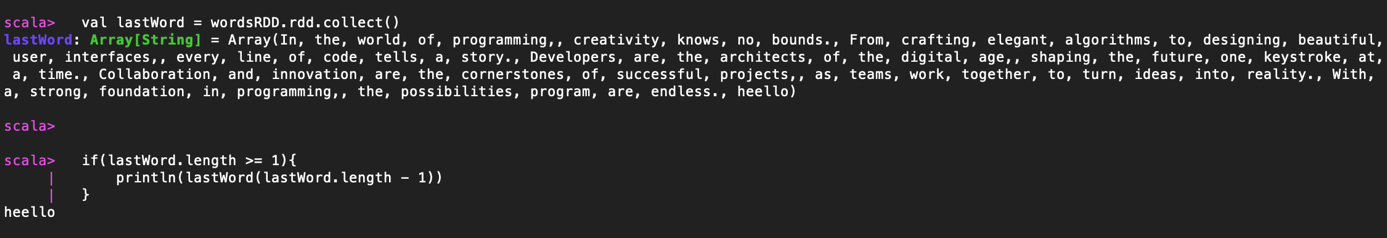


Рисунок 3.6 – Результат работы задания 6

1. **Вывод**

В ходе выполнения данной лабораторной работы нами были проанализированы и изучены техники работы с пакетом SPARK в Scala.