- Прочитайте об операторах Spark. Пришлите ответы на вопросы.
- Какие команды отвечают за:
  - сохранение результата в текстовый файл (это Action или Transformation?);
  - получение первых n-элементов массива (Action или Transformation?);
  - объединение двух RDD в один (Action или Transformation?);
  - в чем разница между Reduce и CoGroup-операторами (Action или Transformation?).

Название команды (метода)	Описание	Тип
saveAsTextFile (путь)	Записывает элементы набора данных в виде текстового файла (или набора текстовых файлов) в заданный каталог в локальной файловой системе, HDFS или любой другой файловой системе, поддерживаемой Hadoop. Spark вызывает toString для каждого элемента, чтобы преобразовать его в строку текста в файле.	**Action**
take(n)	возвращает в виде массива первые n элементов датасета	**Action**
union(second_RDD)	объединяет два RDD в один	**Transformation**
cogroup(other)	сгруппировать данные по ключу из **двух наборов**	**Transformation**
reduce()	принимает функцию оперирующую двумя элементами **одного набора RDD** и возвращает новый элемент того же типа. С поимощью reduce() можно найти сумму элементов RDD, определить их количествои выполнить другие виды агрегирования	**Action**

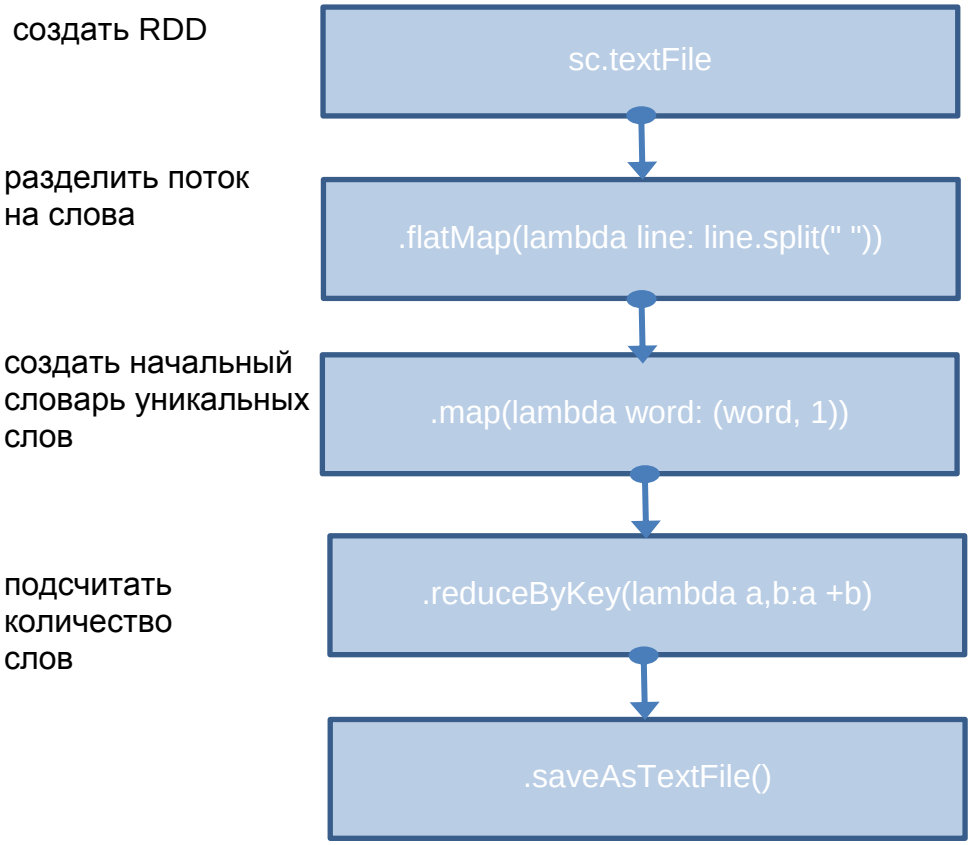
reduce() – выполняет действие над одним набором RDD,

в то время как **cogroup()** –трансформация двух наборов RDD.

Таблица 4.2. Преобразования для двух наборов пар (например: rdd = {(1, 2), (3, 4), (3, 6)} other = {(3, 9)})

Функция	Назначение	Пример	Результат
cogroup	Сгруппировать по ключу данных из двух наборов	rdd.cogroup(other)	{(1, ([2], [])), (3, ([4, 6], [9]))}

- Нарисуйте DAG для Spark для подсчёта количества уникальных слов в файле.



Пример реализации программы:

```
import sys

from pyspark import SparkContext, SparkConf

# создание Spark объекта с необходимой конфигурацией:

sc = SparkContext("local","PySpark Word Count Exmaple")

# чтение данных из текстового файла и разделение его на слова

words = sc.textFile("D:/workspace/spark/input.txt").flatMap(lambda line: line.split(" "))

# подсчёт вхождения каждого слова

wordCounts = words.map(lambda word: (word, 1)).reduceByKey(lambda a,b:a +b)

# сохранение подсчитанных данных в файл

wordCounts.saveAsTextFile("D:/workspace/spark/output/")
```