

Assignment 13

Make sure you have set up both Hadoop and Spark. As Spark framework is based on Hadoop, you need to install Hadoop first and then install Spark by checking the compatibility of Spark Framework version with the Hadoop version.

Turn on Hadoop server using the following command:

start-all.sh

Type the following command to start the Spark shell:

spark-shell

```
v1ack@ubuntu:~$ spark-shell
2023-05-15 17:10:57,972 WARN util.Utils: Your hostname, ubuntu resolves to a loopback address
: 127.0.1.1; using 10.0.2.15 instead (on interface enp0s3)
2023-05-15 17:10:57,973 WARN util.Utils: Set SPARK_LOCAL_IP if you need to bind to another ad
dress
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
2023-05-15 17:11:01,504 WARN util.NativeCodeLoader: Unable to load native-hadoop library for
your platform... using builtin-java classes where applicable
Spark context Web UI available at http://10.0.2.15:4040
Spark context available as 'sc' (master = local[*], app id = local-1684150862087).
Spark session available as 'spark'.
Welcome to

      /--\
     /  V \_/_/_/_/_/_/_/_/_/_/_\_
    /___/\_./_\/_/_/_/_/_/_/_/_\_   version 3.2.4
     /___/\_./_\/_/_/_/_/_/_/_/_\_

Using Scala version 2.12.15 (OpenJDK 64-Bit Server VM, Java 1.8.0_362)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

Now, run the following commands to get the output of average of numbers.

```
val rdd = sc.parallelize(Seq(1, 2, 3, 4, 5))  
val sum = rdd.sum()  
val count = rdd.count()  
val avg = sum / count  
println(avg)
```

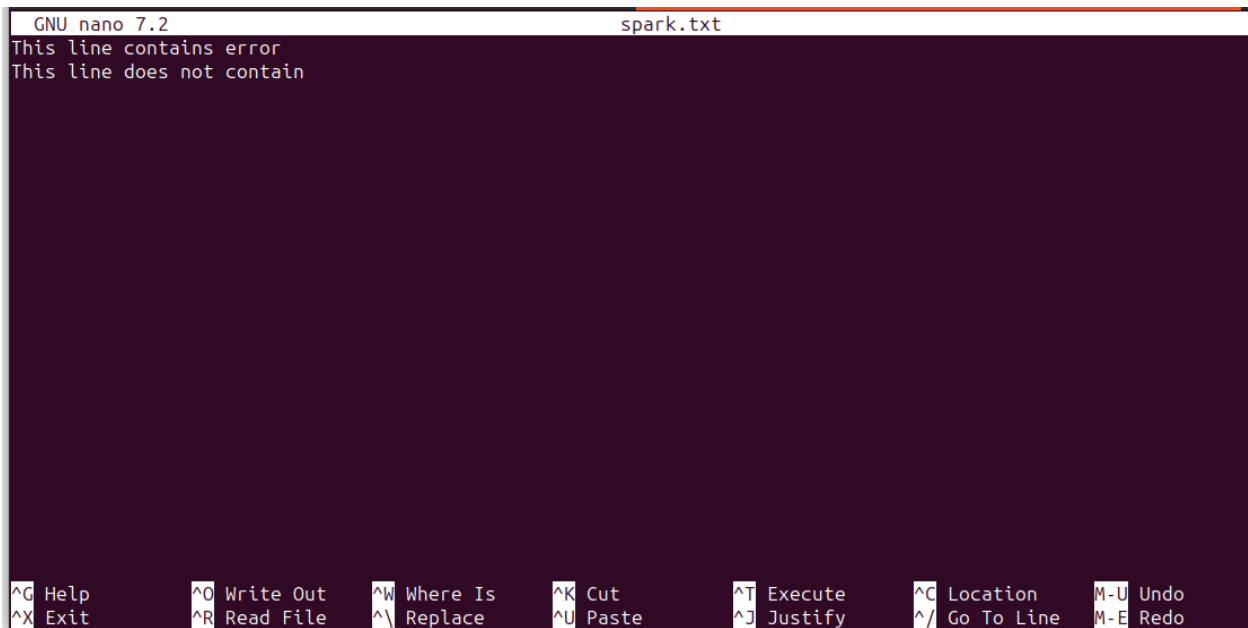
```
scala> val rdd = sc.parallelize(Seq(1,2,3,4,5))  
rdd: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[0] at parallelize at <console>:23  
  
scala>  
  
scala> val sum = rdd.sum()  
sum: Double = 15.0  
  
scala> val count = rdd.count()  
count: Long = 5  
  
scala> val avg = sum / count  
avg: Double = 3.0  
  
scala> println(avg)  
3.0
```

A program which prints all the lines that does not contain the word 'error' in it.

Run the following command to create new file and move it to the Hadoop file system.

```
cd Desktop  
nano spark.txt
```

Add whatever text you want.



```
GNU nano 7.2 spark.txt
This line contains error
This line does not contain

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^_ Go To Line  M-E Redo
```

Ctrl + O

Enter

Ctrl + X

Above commands will save the sentences you wrote into spark.txt

Now, create new directory in Hadoop file system and move the spark.txt to that newly created directory.

Use the following commands:

```
hadoop fs -mkdir /spark
```

```
hadoop fs -put spark.txt /spark
```

Now, go to the spark shell and enter following code to execute:

```
val textFile = sc.textFile("path/to/textfile")  
  
val filteredLines = textFile.filter(line => !line.contains("error"))  
  
filteredLines.collect()
```

Output:

```
scala> val textFile = sc.textFile("/spark/spark.txt")  
textFile: org.apache.spark.rdd.RDD[String] = /spark/spark.txt MapPartitionsRDD[3] at textFile at <console>:23  
  
scala> val filteredLines = textFile.filter(line => !line.contains("error"))  
filteredLines: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[4] at filter at <console>:23  
  
scala> filteredLines.collect()  
res1: Array[String] = Array(This line does not contain)
```