

Task 1

1. Write a program to read a text file and print the number of rows of data in the document.

```
scala> val data = sc.textFile("hdfs://localhost:8020/user/spark/dataset.txt")
data: org.apache.spark.rdd.RDD[String] = hdfs://localhost:8020/user/spark/dataset.txt MapPartitionsRDD[69] at textFile at <console>:24

scala> data.collect()
res60: Array[String] = Array(Mathew,science,grade-3,45,12, Mathew,history,grade-2,55,11, Mark,maths,grade-2,23,13, Mark,science,grade-1,76,13, John,history,grade-1,14,12, John,maths,grade-2,74,13, Lisa,science,grade-1,24,11, Lisa,history,grade-3,86,13, Andrew,maths,grade-1,34,11, Andrew,science,grade-3,25,14, Andrew,history,grade-1,74,12, Mathew,science,grade-2,55,11, Mathew,history,grade-2,87,11, Mark,maths,grade-1,92,13, Mark,science,grade-2,12,12, John,history,grade-1,67,13, John,maths,grade-1,15,11, Lisa,science,grade-2,24,11, Lisa,history,grade-2,98,15, Andrew,maths,grade-1,73,16, Andrew,science,grade-3,44,14, Andrew,history,grade-2,77,11)

scala> data.count
res61: Long = 22

scala>
```

2. Write a program to read a text file and print the number of words in the document.

```
scala> val data = sc.textFile("hdfs://localhost:8020/user/spark/dataset.txt")
data: org.apache.spark.rdd.RDD[String] = hdfs://localhost:8020/user/spark/dataset.txt MapPartitionsRDD[102] at textFile at <console>:24

scala> val counts = data.flatMap(line => line.split(",")).map(word => (word,1)).reduceByKey(_+_).collect.foreach(println)
(maths,6)
(14,3)
(67,1)
(98,1)
(15,1)
(Mark,4)
(grade-3,4)
(grade-1,9)
(35,1)
(history,8)
(Andrew,6)
(45,1)
(55,1)
(Mathew,4)
(16,1)
(86,1)
(92,1)
(84,1)
(science,8)
(26,1)
(87,1)
(grade-2,9)
(44,1)
(12,8)
(13,9)
(24,2)
(76,1)
(John,4)
(77,1)
(74,2)
(11,2)
(Lisa,4)
```

3. We have a document where the word separator is -, instead of space. Write a spark code, to obtain the count of the total number of words present in the document.

```
scala> val data = sc.textFile("hdfs://localhost:8020/user/spark/dataset.txt")
data: org.apache.spark.rdd.RDD[String] = hdfs://localhost:8020/user/spark/dataset.txt MapPartitionsRDD[107] at textFile at <console>:24

scala> val counts = data flatMap(line => line.split("-")) map(word => (word,1)) reduceByKey(_+_).collect.foreach(println)
(John,history,grade,2)
(1,34,13,1)
(Lisa,history,grade,2)
(2,55,13,1)
(1,23,16,1)
(2,87,12,1)
(1,24,12,1)
(1,67,13,1)
(1,76,13,1)
(Mark,maths,grade,2)
(Andrew,science,grade,2)
(1,35,11,1)
(3,86,13,1)
(Mathew,history,grade,2)
(2,77,11,1)
(3,44,14,1)
(John,maths,grade,2)
(3,45,12,1)
(Lisa,science,grade,2)
(2,23,13,1)
(2,98,15,1)
(Mark,science,grade,2)
(1,74,12,1)
(2,12,12,1)
(3,26,14,1)
(2,74,13,1)
(2,55,12,1)
(1,14,12,1)
(Mathew,science,grade,2)
(2,24,13,1)
(Andrew,maths,grade,2)
```

Task 2

Problem Statement 1:

1. Read the text file, and create a tupled rdd.

```

scala> val data = sc.textFile("hdfs://localhost:8020/user/spark/dataset.txt")
data: org.apache.spark.rdd.RDD[String] = hdfs://localhost:8020/user/spark/dataset.txt MapPartitionsRDD[121] at textFile at <console>:24

scala> val pairs = data.map(s => Tuple2(s,1))
pairs: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[122] at map at <console>:26

scala> pairs.collect().foreach(println)
(Mathew.science,grade-3,45,12,1)
(Mathew.history,grade-2,55,13,1)
(Mark.maths,grade-2,23,13,1)
(Mark.science,grade-1,75,13,1)
(John.history,grade-1,14,12,1)
(John.maths,grade-2,74,13,1)
(Lisa.science,grade-1,24,12,1)
(Lisa.history,grade-3,85,13,1)
(Andrew.maths,grade-1,34,13,1)
(Andrew.science,grade-3,26,14,1)
(Andrew.history,grade-1,74,12,1)
(Mathew.science,grade-2,55,12,1)
(Mathew.history,grade-2,87,12,1)
(Mark.maths,grade-1,92,13,1)
(Mark.science,grade-2,12,12,1)
(John.history,grade-1,67,13,1)
(John.maths,grade-1,35,11,1)
(Lisa.science,grade-2,24,13,1)
(Lisa.history,grade-2,99,15,1)
(Andrew.maths,grade-1,23,16,1)
(Andrew.science,grade-3,44,14,1)
(Andrew.history,grade-2,77,11,1)

```

2. Find the count of total number of rows present.

```

scala> val data = sc.textFile("hdfs://localhost:8020/user/spark/dataset.txt")
data: org.apache.spark.rdd.RDD[String] = hdfs://localhost:8020/user/spark/dataset.txt MapPartitionsRDD[132] at textFile at <console>:24

scala> val pairs = data.map(s => Tuple2(s,1))
pairs: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[133] at map at <console>:26

scala> val counts = pairs.reduceByKey((a,b) => a + b)
counts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[134] at reduceByKey at <console>:28

scala> counts.count
res76: Long = 22

scala>

```

3. What is the distinct number of subjects present in the entire school

```
scala> val data = sc.textFile("hdfs://localhost:8020/user/spark/dataset.txt")
data: org.apache.spark.rdd.RDD[String] = hdfs://localhost:8020/user/spark/dataset.txt MapPartitionsRDD[148] at textFile at <console>:24

scala> val pairs = data.map(s => Tuple2(s,1))
pairs: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[149] at map at <console>:26

scala> var aa = pairs.map(a => a._1.map(_.split(",")).map(h => h(1))).distinct.collect
aa: Array[String] = Array(maths, history, science)

scala>
```

4. What is the count of the number of students in the school, whose name is Mathew and marks is 55

```
scala> val data = sc.textFile("hdfs://localhost:8020/user/spark/dataset.txt")
data: org.apache.spark.rdd.RDD[String] = hdfs://localhost:8020/user/spark/dataset.txt MapPartitionsRDD[218] at textFile at <console>:24

scala> val pairs = data.map(s => Tuple2(s,1))
pairs: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[219] at map at <console>:26

scala> var aa = pairs.map(a => a._1.map(_.split(",")).filter(x => x(0) == "Mathew" && x(3) == "55").count
aa: Long = 2

scala>
```

Problem Statement 2:

1. What is the count of students per grade in the school?

```
scala> val data = sc.textFile("hdfs://localhost:8020/user/spark/dataset.txt")
data: org.apache.spark.rdd.RDD[String] = hdfs://localhost:8020/user/spark/dataset.txt MapPartitionsRDD[268] at textFile at <console>:24

scala> val pairs = data.map(s => Tuple2(s,1))
pairs: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[269] at map at <console>:26

scala> var count = pairs.map(a => a._1.map(_.split(",")).map(a => a(2)).countByValue()
count: scala.collection.Map[String,Long] = Map(grade-1 -> 4, grade-1 -> 9, grade-2 -> 9)

scala> count.foreach(println)
(grade-3,4)
(grade-1,9)
(grade-2,9)

scala>
```

2. Find the average of each student (Note - Mathew is grade-1, is different from Mathew in some other grade!)

```
scala> val data = sc.textFile("hdfs://localhost:8020/user/spark/dataset.txt")
data: org.apache.spark.rdd.RDD[String] = hdfs://localhost:8020/user/spark/dataset.txt MapPartitionsRDD[350] at textFile at <console>:24

scala> val values = data.map(x => (x.toString.split(",")).map(x => ((x(0),x(2)),x(3).toInt)).reduceByKey(_+_))
warning: there was one feature warning; re-run with -feature for details
values: org.apache.spark.rdd.RDD[(String, String), Int] = ShuffledRDD[393] at reduceByKey at <console>:28

scala> val totalLen = values.map(x => ((x._1,1))).reduceByKey(_+_).sortByKey()
totalLen: org.apache.spark.rdd.RDD[(String, String), Int] = ShuffledRDD[396] at sortByKey at <console>:30

scala> val total_mars = values.reduceByKey(_+_).sortByKey()
total_marks: org.apache.spark.rdd.RDD[(String, String), Int] = ShuffledRDD[398] at sortByKey at <console>:30

scala> val join_data = total_marks.join(totalLen)
join_data: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = MapPartitionsRDD[401] at join at <console>:34

scala> val result_avg = join_data.map(x => ((x._1.toString)+ " ==> "+(x._2._1.toInt)/(x._2._2.toInt))).foreach(println)
(Lisa,grade-1) ==> 24
(Mark,grade-2) ==> 33
(Lisa,grade-2) ==> 122
(Mathew,grade-3) ==> 45
(Andrew,grade-2) ==> 77
(Andrew,grade-1) ==> 131
(Lisa,grade-3) ==> 86
(John,grade-1) ==> 110
(John,grade-2) ==> 74
(Mark,grade-1) ==> 168
(Andrew,grade-3) ==> 70
(Mathew,grade-2) ==> 197
result avg: Unit = ()
```

3. What is the average score of students in each subject across all grades?

```
scala> val data = sc.textFile("hdfs://localhost:8020/user/spark/dataset.txt")
data: org.apache.spark.rdd.RDD[String] = hdfs://localhost:8020/user/spark/dataset.txt MapPartitionsRDD[325] at textFile at <console>:24

scala> val pairs = data.map(s => Tuple2(s,1))
pairs: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[326] at map at <console>:26

scala> var values = pairs.map(a => a._1).map(_.split(",")).map(aa => (aa(0), aa(3).toInt))
values: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[329] at map at <console>:36

scala> val mapped = values.mapValues(mark => [mark,1])
mapped: org.apache.spark.rdd.RDD[(String, (Int, Int))] = MapPartitionsRDD[330] at mapValues at <console>:32

scala> val reduced = mapped.reduceByKey((x,y) => (x._1 + y._1, x._2 + y._2))
reduced: org.apache.spark.rdd.RDD[(String, (Int, Int))] = ShuffledRDD[331] at reduceByKey at <console>:38

scala> val average = reduced.map(x => val temp = x._2
  | val total = temp._1
  | val count = temp._2
  | (x._1, total/count)
  | )
average: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[332] at map at <console>:42

scala> average.collect()
res119: Array[(String, Int)] = Array((Mark,50), (Andrew,46), (Mathew,60), (John,47), (Lisa,58))

scala>
```

