

ASSIGNMENT 19.1

Task 1

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

```
acadgild@localhost:~  
[acadgild@localhost ~]$ cat sample.txt  
Building your Website  
Basics of Public Relations  
Basics of Online Advertising  
Basics of Offline Marketing  
Basics of Search Engine Optimization  
Basics of Landing Page Optimization  
The Importance of Email Campaigns  
Using Social Networks  
Measure and Act  
Interpreting your Metrics  
Tracking and Closing Sales
```

Above screen shot, shows a text file **sample.txt** present in home acadgild folder.

To read this text file we will create an RDD with name **readLine** and command used is
val readLine = sc.textFile ("file:///home/acadgild/sample.txt")
and to count the number of rows in the text file is **:readLine.count ()**

```
acadgild@localhost:~  
scala> val readLine = sc.textFile("file:///home/acadgild/sample.txt")  
readLine: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/sample.txt MapPartitionsRDD[7] at textFile at <console>:24  
  
scala> readLine.count()  
res4: Long = 11  
  
scala> █
```

Above screen, shot shows the no of rows in text file are 11

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

To print the no. of words in the given text file we used the following code by creating a flat map and used split function to separate the words and count all the words.

```
val wordCount = readLine.flatMap(x => x.split(" "))  
println(wordCount.count())
```

```
scala> val wordCount = readLine.flatMap(x => x.split(" "))  
wordCount: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[10] at flatMap at <console>:26  
  
scala> println(wordCount.count())  
43  
  
scala> █
```

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.

For above problem we have a text document **test.txt** as shown below in screen shot. This document is “-” separated as you can see in the screen shot.

```
acadgild@localhost:~  
[acadgild@localhost ~]$ cat test.txt  
This-is-a-wonderful-BDHS-Session  
This-is-a-wonderful-BDHS-Session  
This-is-a-wonderful-BDHS-Session  
This-is-a-wonderful-BDHS-Session  
This-is-a-wonderful-BDHS-Session  
This-is-a-wonderful-BDHS-Session  
This-is-a-wonderful-BDHS-Session  
This-is-a-wonderful-BDHS-Session
```

To read this document and to count the number of words we used below code to get the result.

```
val test = sc.textFile("file:///home/acadgild/test.txt")  
val words = test.flatMap(x => x.split("-"))  
println(words.count())
```

```
acadgild@localhost:~  
scala> val test = sc.textFile("file:///home/acadgild/test.txt")  
test: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/test.txt MapPartitionsRDD[146] at textFile  
  
scala> val words = test.flatMap(x => x.split("-"))  
words: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[147] at flatMap at <console>:26  
  
scala> println(words.count())  
48  
  
scala> █
```

As shown above, number of words in test.txt document are 48.

Task 2

Problem Statement 1:

We have **19_Dataset.txt** text file of student marks data present in the home acadgild folder.

```
acadgild@localhost:~  
[acadgild@localhost ~]$ cat 19_Dataset.txt  
Mathew,science,grade-3,45,12  
Mathew,history,grade-2,55,13  
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-1,24,12  
Lisa,history,grade-3,86,13  
Andrew,maths,grade-1,34,13  
Andrew,science,grade-3,26,14  
Andrew,history,grade-1,74,12  
Mathew,science,grade-2,55,12  
Mathew,history,grade-2,87,12  
Mark,maths,grade-1,92,13  
Mark,science,grade-2,12,12  
John,history,grade-1,67,13  
John,maths,grade-1,35,11  
Lisa,science,grade-2,24,13  
Lisa,history,grade-2,98,15  
Andrew,maths,grade-1,23,16  
Andrew,science,grade-3,44,14  
Andrew,history,grade-2,77,11
```

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

To read the above text file we used the following code by creating the RRD with name **student**:

```
val student = sc.textFile ("file:///home/acadgild/19_Dataset.txt")
```

```
acadgild@localhost:~  
scala> val student = sc.textFile("file:///home/acadgild/19_Dataset.txt")  
student: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/19_Dataset.txt MapPartitionsRDD[19] at textFile at <console>:24  
scala> █
```

Create a tuple RDD by using the map and array function on student RDD and splitting the records by Comma separator. We used the following code to get the result:

```
val studentTuple = student.map(line => line.split(",")).map(array =>  
(array(0),array(1),array(2),array(3).toInt,array(4).toInt))
```

```
scala> val studentTuple = student.map(line => line.split(",")).map(array => {array(0), array(1), array(2), array(3).toInt, array(4).toInt})
studentTuple: org.apache.spark.rdd.RDD[(String, String, String, Int, Int)] = MapPartitionsRDD[21] at map at <console>:26

scala> studentTuple.foreach(println)
(Mathew,science,grade-3,45,12)
(Mathew,history,grade-2,87,12)
(Mark,maths,grade-1,92,13)
(Mark,science,grade-2,12,12)
(John,history,grade-1,67,13)
(John,maths,grade-1,35,11)
(Mathew,history,grade-2,55,13)
(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-1,24,12)
(Lisa,history,grade-3,86,13)
(Andrew,maths,grade-1,34,13)
(Andrew,science,grade-3,26,14)
(Andrew,history,grade-1,74,12)
(Mathew,science,grade-2,55,12)
(Lisa,science,grade-2,24,13)
(Lisa,history,grade-2,98,15)
(Andrew,maths,grade-1,23,16)
(Andrew,science,grade-3,44,14)
(Andrew,history,grade-2,77,11)

scala> █
```

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

Used the count method on student RDD to count the no. of rows
student.count

```
acadgild@localhost:~$
scala> val student = sc.textFile("file:///home/acadgild/19_Dataset.txt")
student: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/19_Dataset.txt MapPartitionsRDD[23] at textFile at <console>:24

scala> student.count
res14: Long = 22

scala> █
```

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

To calculate the distinct number of subjects, we used the map method on **student**, select only subject column, and assigned it to **studentRDD**. And then calculate the distinct subject by calling **distinct** function.

Below are the following commands use to get the different subject in the entire school:

```
val student = sc.textFile ("file:///home/acadgild/19_Dataset.txt")
val studentRDD = student.map(x => x.split(",") (1))
val distinctSubject = studentRDD.distinct
distinctSubject.collect
distinctSubject.foreach (println)
distinctSubject.count
```

Below screen shot shows there are three different subjects are present in the school.

```
acadgild@localhost:~  
scala> val student = sc.textFile("file:///home/acadgild/19_Dataset.txt")  
student: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/19_Dataset.txt MapPartitionsRDD[61] at  
  
scala> val studentRDD = student.map(x => x.split(",") {1})  
studentRDD: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[62] at map at <console>:26  
  
scala> val distinctSubject = studentRDD.distinct  
distinctSubject: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[65] at distinct at <console>:28  
  
scala> distinctSubject.collect  
res15: Array[String] = Array(history, science, maths)  
  
scala> distinctSubject.foreach(println)  
history  
science  
maths  
  
scala> distinctSubject.count  
res17: Long = 3
```

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

To calculate the no. of students whose name is Mathew; we used the **filter** function on **student** to filter name Mathew and his marks 55.

Below are the codes use to calculate the desire result:

```
val student = sc.textFile ("file:///home/acadgild/19_Dataset.txt")  
val mathew55 = student.filter(x => x.contains ("Mathew") && x.contains ("55"))  
mathew55.collect  
mathew55.count
```

```
acadgild@localhost:~  
scala> val student = sc.textFile("file:///home/acadgild/19_Dataset.txt")  
student: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/19_Dataset.txt MapPartitionsRDD[32] at  
  
scala> val mathew55 = student.filter(x => x.contains("Mathew") && x.contains("55"))  
mathew55: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[33] at filter at <console>:26  
  
scala> mathew55.collect  
res21: Array[String] = Array(Mathew,history,grade-2,55,13, Mathew,science,grade-2,55,12)  
  
scala> mathew55.count  
res22: Long = 2
```

Above screen shot shows, there are two students whose name is Mathew and marks is 55.

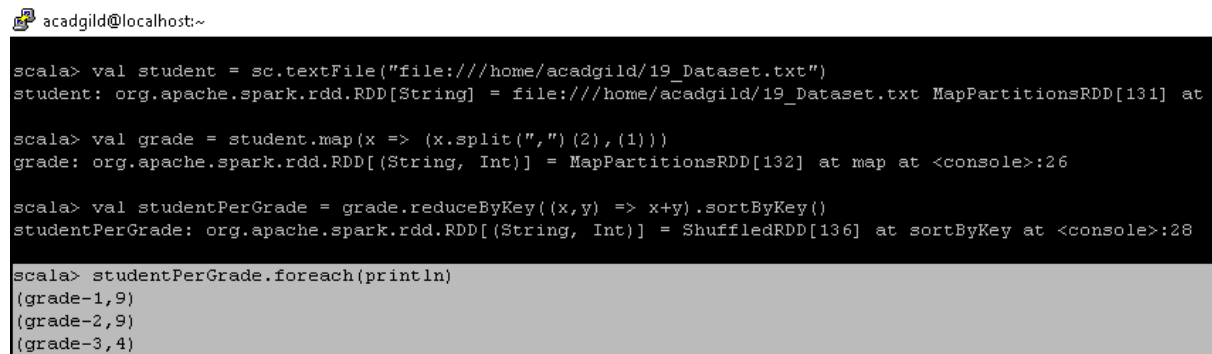
Problem Statement 2:

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

To find no. of students per grade, we used map function on student to select grade column and mapped it value 1. Then we used reduceByKey to sum up all the values of each grade.

Below are the following codes:

```
val student = sc.textFile("file:///home/acadgild/19_Dataset.txt")
val grade = student.map(x => (x.split(",")(2),1))
val studentPerGrade = grade.reduceByKey((x,y) => x+y).sortByKey()
studentPerGrade.foreach(println)
```



```
acadgild@localhost:~
scala> val student = sc.textFile("file:///home/acadgild/19_Dataset.txt")
student: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/19_Dataset.txt MapPartitionsRDD[131] at

scala> val grade = student.map(x => (x.split(",")(2),1))
grade: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[132] at map at <console>:26


scala> val studentPerGrade = grade.reduceByKey((x,y) => x+y).sortByKey()
studentPerGrade: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[136] at sortByKey at <console>:28

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

Above screenshot shows number of students in every grade.

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

To calculate the average score of each student in different grade, first we have created the RDD with name student as show below in the screen shot:



```
acadgild@localhost:~
scala> val student = sc.textFile("file:///home/acadgild/19_Dataset.txt")
student: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/19_Dataset.txt MapPartitionsRDD[97] at textFile at <console>:24
```

We used map function on **student** to create RDD **subjectGrade** for selecting the name and grade as key and marks as value. Each value of marks is mapped with the value 1. For that, we use the following code:

```
val subjectGrade = student.map(x =>((x.split(",")(0),x.split(",")(2)),(x.split(",")(3).toInt,1)))
```

```
scala> val subjectGrade = student.map(x =>((x.split(",")(0),x.split(",")(2)),(x.split(",")(3).toInt,1)))
subjectGrade: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = MapPartitionsRDD[98] at map at <console>:26

scala> subjectGrade.foreach(println)
((Mathew,grade-3),(45,1))
((Mathew,grade-2),(55,1))
((Mark,grade-2),(23,1))
((Mark,grade-1),(76,1))
((John,grade-1),(14,1))
((John,grade-2),(74,1))
((Lisa,grade-1),(24,1))
((Lisa,grade-3),(86,1))
((Andrew,grade-1),(34,1))
((Andrew,grade-3),(26,1))
((Andrew,grade-1),(74,1))
((Mathew,grade-2),(55,1))
((Mathew,grade-2),(87,1))
((Mark,grade-1),(92,1))
((Mark,grade-2),(12,1))
((John,grade-1),(67,1))
((John,grade-1),(35,1))
((Lisa,grade-2),(24,1))
((Lisa,grade-2),(98,1))
((Andrew,grade-1),(23,1))
((Andrew,grade-3),(44,1))
((Andrew,grade-2),(77,1))
```

Now, we used **reduceByKey** on subjectGrade to sum up the marks for each key of name and grade:

```
val reduceSubject = subjectGrade.reduceByKey((x,y) => (x._1 + y._1,x._2 + y._2))
```

```
acadgild@localhost:~
scala> val reduceSubject = subjectGrade.reduceByKey((x,y) => (x._1 + y._1,x._2 + y._2))
reduceSubject: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = ShuffledRDD[99] at reduceByKey at

scala> reduceSubject.foreach(println)
((Lisa,grade-1),(24,1))
((Andrew,grade-2),(77,1))
((John,grade-1),(116,3))
((John,grade-2),(74,1))
((Mathew,grade-2),(197,3))
((Mark,grade-2),(35,2))
((Lisa,grade-2),(122,2))
((Mathew,grade-3),(45,1))
((Andrew,grade-1),(131,3))
((Lisa,grade-3),(86,1))
((Mark,grade-1),(168,2))
((Andrew,grade-3),(70,2))
```

Now, we calculate the average marks by dividing the marks by its count for each key.

```
val avgOfEachStudent = reduceSubject.mapValues{ case (x,y) =>(x.toFloat)/y}.sortByKey()
```

```
avgOfEachStudent.foreach(println)
```

acadgild@localhost:~

```
scala> val avgOfEachStudent = reduceSubject.mapValues( case (x,y) => (x.toFloat)/y ).sortByKey()
avgOfEachStudent: org.apache.spark.rdd.RDD[(String, String), Float]] = ShuffledRDD[103] at sortByKey at
scala> avgOfEachStudent.foreach(println)
({Andrew,grade-1},43.666668)
({Andrew,grade-2},77.0)
({Andrew,grade-3},35.0)
({John,grade-1},38.666668)
({John,grade-2},74.0)
({Lisa,grade-1},24.0)
({Lisa,grade-2},61.0)
({Lisa,grade-3},86.0)
({Mark,grade-1},84.0)
({Mark,grade-2},17.5)
({Mathew,grade-2},65.666664)
({Mathew,grade-3},45.0)
```

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

To find the average score of student in each subject we follow the same procedure as shown in above question.

First, we create the RDD **student** to read the text file, and then we used map functions on student by creating RDD **subjectGrade** to select name and subject as **key** and marks as **value** and mapped the value 1 to it.

Now, we create RDD **reduceStudent** to sum all the marks with respect to keys name and subject.

Now, we calculate the average by dividing the sum of marks by its no of count.

We used the following code to get the result;

```
val student = sc.textFile("file:///home/acadgild/19_Dataset.txt")
val studentGrade = student.map(x=>((x.split(",")(0),x.split(",")(1)),(x.split(",")(3).toInt,1)))
val reduceStudent = studentGrade.reduceByKey((x,y) => (x._1 + y._1,x._2 + y._2))
val avgOfEachSubject = reduceStudent.mapValues( case(x,y) => (x.toFloat)/y ).sortByKey()
```

acadgild@localhost:~

```
scala> val student = sc.textFile("file:///home/acadgild/19_Dataset.txt")
student: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/19_Dataset.txt MapPartitionsRDD[12] at textFile at <console>:24

scala> val studentGrade = student.map(x=>((x.split(",")(0),x.split(",")(1)),(x.split(",")(3).toInt,1)))
studentGrade: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = MapPartitionsRDD[13] at map at <console>:26

scala> val reduceStudent = studentGrade.reduceByKey((x,y) => (x._1 + y._1,x._2 + y._2))
reduceStudent: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = ShuffledRDD[14] at reduceByKey at <console>:28

scala> val avgOfEachSubject = reduceStudent.mapValues( case(x,y) => (x.toFloat)/y ).sortByKey()
avgOfEachSubject: org.apache.spark.rdd.RDD[(String, String), Float]] = ShuffledRDD[18] at sortByKey at <console>:30
```


We used `foreach(println)` to show the results of average marks of each student as shown below:

`avgOfEachSubject.foreach(println)`

```
scala> avgOfEachSubject.foreach(println)
((Lisa,science),24.0)
((Mark,maths),57.5)
((Mark,science),44.0)
((Mathew,history),71.0)
((Mathew,science),50.0)
((Andrew,history),75.5)
((Andrew,maths),28.5)
((Andrew,science),35.0)
((John,history),40.5)
((John,maths),54.5)
((Lisa,history),92.0)

scala>
```

4. What is the average score of students in each subject per grade?

To find the average score of student in each subject we follow the same procedure as shown in above question.

First, we create the RDD **student** to read the text file, and then we used map functions on student by creating RDD **subjectGrade** to select subject and grade as **key** and marks as **value** and mapped the value 1 to it.

Now, we create RDD **reduceStudent** to sum all the marks with respect to keys subject and grade.

Now, we calculate the average by dividing the sum of marks by its no of count.

We used the following code to get the result;

```
val student = sc.textFile("file:///home/acadgild/19_Dataset.txt")
val studentGrade = student.map(x=>((x.split(",")(1),x.split(",")(2)),(x.split(",")(3).toInt,1)))
val reduceStudent = studentGrade.reduceByKey((x,y) => (x._1 + y._1,x._2 + y._2))
val avgOfEachSubjectPerGrade = reduceStudent.mapValues{ case(x,y) => (x.toFloat)/y
}.sortByKey()
```

acadgild@localhost:~

```
scala> val student = sc.textFile("file:///home/acadgild/19_Dataset.txt")
student: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/19_Dataset.txt MapPartitionsRDD[20] at textFile at <console>:24

scala> val studentGrade = student.map(x=>((x.split(",")(1),x.split(",")(2)),(x.split(",")(3).toInt,1)))
studentGrade: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = MapPartitionsRDD[21] at map at <console>:26

scala> val reduceStudent = studentGrade.reduceByKey((x,y) => (x._1 + y._1,x._2 + y._2))
reduceStudent: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = ShuffledRDD[22] at reduceByKey at <console>:28

scala> val avgOfEachSubjectPerGrade = reduceStudent.mapValues{ case(x,y) => (x.toFloat)/y }.sortByKey()
avgOfEachSubjectPerGrade: org.apache.spark.rdd.RDD[(String, String), Float] = ShuffledRDD[26] at sortByKey at <console>:30
```

We used `foreach(println)` to show the results of average marks of each student per grade as shown below:

`avgOfEachSubjectPerGrade.foreach(println)`

```
scala> avgOfEachSubjectPerGrade.foreach(println)
[Stage 28:>                                     (0 + 0) / 2] ((maths,grade-2),48.5)
((science,grade-1),50.0)
((science,grade-2),30.333334)
((science,grade-3),38.333332)
((history,grade-1),51.666668)
((history,grade-2),79.25)
((history,grade-3),86.0)
((maths,grade-1),46.0)
scala>
```

5. For all students in grade-2, how many have average score greater than 50?

To find the average score of student in each subject we follow the below procedure

First, we create the RDD **student** to read the text file, and then we used map functions on student by creating RDD **subjectGrade** to select name and grade as **key** and marks as **value** and mapped the value 1 to it.

Now, we create RDD **reduceStudent** to sum all the marks with respect to keys subject and grade.

Now, we calculate the average by dividing the sum of marks by its no of count.

After getting the average value, we used **filter** function to filter the marks greater than the average value 50 and grade is grade-2.

```
val student = sc.textFile("file:///home/acadgild/19_Dataset.txt")
```

```
val subjectGrade = student.map(x=>((x.split(",")(0),x.split(",")(2)),(x.split(",")(3).toInt,1)))
```

```
val reduceSubject = subjectGrade.reduceByKey((x,y) => (x._1 + y._1,x._2 + y._2))
```

```
val avgOfEachStudent = reduceSubject.mapValues{ case (x,y) =>(x.toFloat)/y}.sortByKey()
```

```
val avgOfEachStudentInGrade2 = avgOfEachStudent.filter(x=> x._1._2=="grade-2" && x._2 > 50)
```

acadgild@localhost:~

```
scala> val student = sc.textFile("file:///home/acadgild/19_Dataset.txt")
student: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/19_Dataset.txt MapPartitionsRDD[28] at textFile at <console>:24

scala> val subjectGrade = student.map(x => ((x.split(",") (0), x.split(",") (2)), (x.split(",") (3).toInt, 1)))
subjectGrade: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = MapPartitionsRDD[29] at map at <console>:26

scala> val reduceSubject = subjectGrade.reduceByKey((x,y) => (x._1 + y._1, x._2 + y._2))
reduceSubject: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = ShuffledRDD[30] at reduceByKey at <console>:28

scala> val avgOfEachStudent = reduceSubject.mapValues( case (x,y) => (x.toFloat)/y).sortByKey()
avgOfEachStudent: org.apache.spark.rdd.RDD[(String, String), Float] = ShuffledRDD[34] at sortByKey at <console>:30

scala> val avgOfEachStudentInGrade2 = avgOfEachStudent.filter(x => x._1._2 == "grade-2" && x._2 > 50)
avgOfEachStudentInGrade2: org.apache.spark.rdd.RDD[(String, String), Float] = MapPartitionsRDD[35] at filter at <console>:32
```

We used foreach(println) to show the results of average marks of each student per grade-2 and no. of such students as shown below:

```
avgOfEachStudentInGrade2.foreach(println)
avgOfEachStudentInGrade2.count()
```

```
scala> avgOfEachStudentInGrade2.foreach(println)
(Lisa,grade-2),61.0
(Mathew,grade-2),65.666664
(Andrew,grade-2),77.0
(John,grade-2),74.0

scala> avgOfEachStudentInGrade2.count()
res9: Long = 4
```

Problem Statement 3:

Are there any students in the college that satisfy the below criteria:

1. Average score per student_name across all grades is same as average score per student_name per grade

Hint - Use Intersection Property

For above problem, we have to calculate the average score per student across all grades first, second, we have to calculate the average score per student per grade and then we have to check intersection between these two scores.

Below code is used to calculate average score per student across all grades:

```
val student = sc.textFile("file:///home/acadgild/19_Dataset.txt")

val studentMarks = student.map(x => (x.split(",") (0), (x.split(",") (3).toInt, 1)))
```

```
val studentMarksReduce = studentMarks.reduceByKey((x,y) => (x._1 + y._1,x._2 + y._2))
```

```
val studentMarksCount = studentMarksReduce.mapValues{case (x,y)  
=>(x.toFloat)/y}.sortByKey()
```

```
val studentMarksAllGrade = studentMarksCount.map(x => x._1 + "," + x._2)
```

```
studentMarksAllGrade.foreach(println)
```

acadgild@localhost:~

```
scala> val student = sc.textFile("file:///home/acadgild/19_Dataset.txt")  
student: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/19_Dataset.txt MapPartitionsRDD[23] at textFile at <console>:23  
  
scala> val studentMarks = student.map(x => (x.split(",")(0), (x.split(",")(3).toInt, 1)))  
studentMarks: org.apache.spark.rdd.RDD[(String, (Int, Int))] = MapPartitionsRDD[24] at map at <console>:26  
  
scala> val studentMarksReduce = studentMarks.reduceByKey((x,y) => (x._1 + y._1,x._2 + y._2))  
studentMarksReduce: org.apache.spark.rdd.RDD[(String, (Int, Int))] = ShuffledRDD[25] at reduceByKey at <console>:28  
  
scala> val studentMarksCount = studentMarksReduce.mapValues{case (x,y) => (x.toFloat)/y}.sortByKey()  
studentMarksCount: org.apache.spark.rdd.RDD[(String, Float)] = ShuffledRDD[29] at sortByKey at <console>:30  
  
scala> val studentMarksAllGrade = studentMarksCount.map(x => x._1 + "," + x._2)  
studentMarksAllGrade: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[30] at map at <console>:32
```

```
scala> studentMarksAllGrade.foreach(println)  
Andrew,46.333332  
John,47.5  
Lisa,58.0  
Mark,50.75  
Mathew,60.5
```

Below code is used to calculate average score per student per grade:

```
val subjectGrade = student.map(x => ((x.split(",")(0),x.split(",")(2)),(x.split(",")(3).toInt,1)))
```

```
val subjectGradeReduce = subjectGrade.reduceByKey((x,y) => (x._1 + y._1,x._2 + y._2))
```

```
val subjectGradeReduceCount = subjectGradeReduce.mapValues{case (x,y)  
=>(x.toFloat)/y}.sortByKey()
```

```
val studentMarksPerGrade = subjectGradeReduceCount.map(x => x._1._1 + "," +  
x._2.toDouble)
```

```
studentMarksPerGrade.foreach(println)
```

acadgild@localhost:~

```
scala> val subjectGrade = student.map(x => ((x.split(",")(0),x.split(",")(2)),(x.split(",")(3).toInt,1)))  
subjectGrade: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD[31] at map at <console>:33  
  
scala> val subjectGradeReduce = subjectGrade.reduceByKey((x,y) => (x._1 + y._1,x._2 + y._2))  
subjectGradeReduce: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = ShuffledRDD[32] at reduceByKey at <console>:35  
  
scala> val subjectGradeReduceCount = subjectGradeReduce.mapValues{case (x,y) => (x.toFloat)/y}.sortByKey()  
subjectGradeReduceCount: org.apache.spark.rdd.RDD[((String, String), Float)] = ShuffledRDD[36] at sortByKey at <console>:37  
  
scala> val studentMarksPerGrade = subjectGradeReduceCount.map(x => x._1._1 + "," + x._2.toDouble)  
studentMarksPerGrade: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[37] at map at <console>:32
```


```
scala> studentMarksPerGrade.foreach(println)
Andrew,43.66666793823242
Andrew,77.0
Andrew,35.0
John,38.66666793823242
John,74.0
Lisa,24.0
Lisa,61.0
Lisa,86.0
Mark,84.0
Mark,17.5
Mathew,65.66666412353516
Mathew,45.0
```

We used **intersection** to check whether any common student whose average score across grades is same as average score per grade.

```
val commonStudents = studentMarksAllGrade.intersection(studentMarksPerGrade)
```

```
commonStudents.foreach(println)
```

As shown in the result there is no student whose average score across grades is same as average score per grade.

 acadgild@localhost:~

```
scala> val commonStudents = studentMarksAllGrade.intersection(studentMarksPerGrade)
commonStudents: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[43] at intersection at <console>:42

scala> commonStudents.foreach(println)

scala> commonStudents.collect
res9: Array[String] = Array()

scala> █
```