

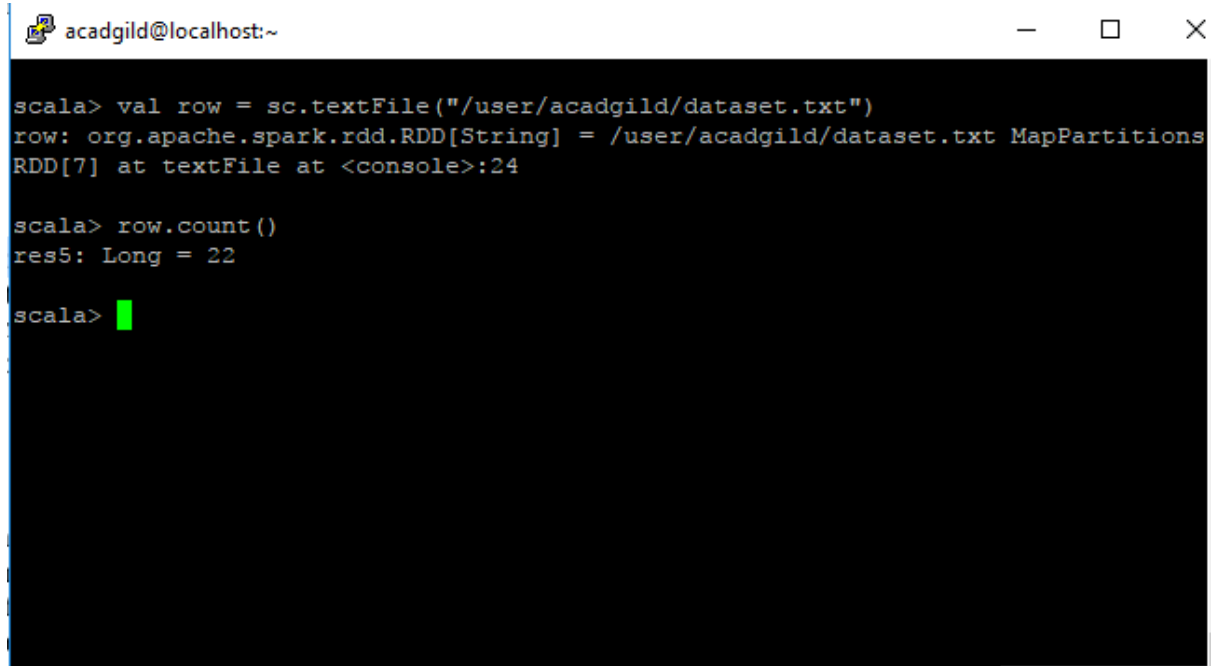
## Assignment 19

### Task 1

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

```
val row = sc.textFile("/user/acadgild/dataset.txt")
```

```
row.count()
```

A screenshot of a Scala REPL window. The window title is "acadgild@localhost:~". The terminal shows the following commands and output:

```
scala> val row = sc.textFile("/user/acadgild/dataset.txt")
row: org.apache.spark.rdd.RDD[String] = /user/acadgild/dataset.txt MapPartitions
RDD[7] at textFile at <console>:24

scala> row.count()
res5: Long = 22

scala>
```

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

```
val base = sc.textFile("/user/acadgild/dataset.txt")
```

```
val words = base.flatMap(x => x.split(","))
```

```
words.count()
```

```
acadgild@localhost:~  
scala> val base = sc.textFile("/user/acadgild/dataset.txt")  
base: org.apache.spark.rdd.RDD[String] = /user/acadgild/dataset.txt MapPartitionsRDD[67] at tex  
tFile at <console>:24  
  
scala> val words = base.flatMap(x => x.split("-"))  
words: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[68] at flatMap at <console>:26  
  
scala> words.count()  
res29: Long = 44  
  
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.**

**val base = sc.textFile("/user/acadgild/dataset.txt")**

**val words = base.flatMap(x => x.split("-"))**

**words.count()**

```
acadgild@localhost:~  
scala> val base = sc.textFile("/user/acadgild/dataset.txt")  
base: org.apache.spark.rdd.RDD[String] = /user/acadgild/dataset.txt MapPartitionsRDD[67] at tex  
tFile at <console>:24  
  
scala> val words = base.flatMap(x => x.split("-"))  
words: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[68] at flatMap at <console>:26  
  
scala> words.count()  
res29: Long = 44  
  
scala> █
```

## Task 2

### Problem Statement 1:

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

**val baseRDD =**

```
sc.textFile("/user/acadgild/dataset.txt").map(=>(x.split(",")(0),  
x.split(",")(1), x.split(",")(2), x.split(",")(3).toInt, x.split(",")(4).toInt))
```

**baseRDD.foreach(println)**

```
scala> val baseRDD = sc.textFile("/user/acadgild/dataset.txt").map(x=>(x.split(",")  
(0), (x.split(",") (1), x.split(",") (2), x.split(",") (3).toInt, x.split(",") (4).to  
Int)))  
baseRDD: org.apache.spark.rdd.RDD[(String, (String, String, Int, Int))] = MapPar  
titionsRDD[20] at map at <console>:24  
  
scala> baseRDD.foreach(println)  
(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))  
  
scala> █
```

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

```
val baseRDD = sc.textFile("/user/acadgild/dataset.txt").map(=>(x.split(",")(0),  
x.split(",")(1), x.split(",")(2), x.split(",")(3).toInt, x.split(",")(4).toInt))
```

**baseRDD.count()**

```
acadgild@localhost:~  
scala> val baseRDD = sc.textFile("/user/acadgild/dataset.txt").map(x=>(x.split(",")(0), (x.split(",")(1), x.split(",")(2), x.split(",")(3).toInt, x.split(",")(4).toInt)))  
baseRDD: org.apache.spark.rdd.RDD[(String, (String, String, Int, Int))] = MapPartitionsRDD[29] at map at <console>:24  
  
scala> baseRDD.count()  
res13: Long = 22  
  
scala> █
```

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

**val baseRDD =**

**sc.textFile("/user/acadgild/dataset.txt").map(x=>(x.split(",")(1),1))**

**val RDDreduce = baseRDD.reduceByKey((x,y)=>(x+y))**

**RDDreduce.foreach(println)**

```
acadgild@localhost:~  
scala> val baseRDD = sc.textFile("/user/acadgild/dataset.txt").map(x=>(x.split(",")(1),1))  
baseRDD: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[32] at map at <console>:24  
  
scala> val RDDreduce = baseRDD.reduceByKey((x,y)=>(x+y))  
RDDreduce: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[33] at reduceByKey at <console>:26  
  
scala> RDDreduce.foreach(println)  
(maths,6)  
(history,8)  
(science,8)  
  
scala> █
```

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

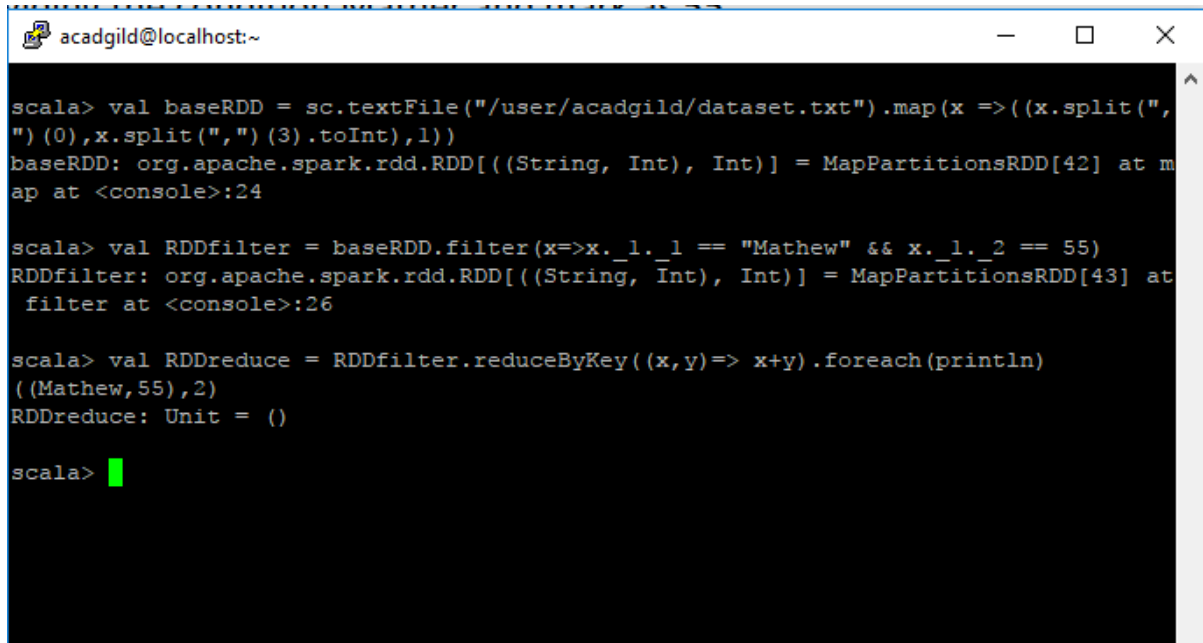
```

val baseRDD =
sc.textFile("/user/acadgild/dataset.txt").map(x=>((x.split(",")(0),x.split(",")(3)
).toInt),1))

val RDDfilter = baseRDD.filter(x=>x._1._1 == "Mathew" && x._1._2 == 55)

val RDDreduce == RDDfilter.reduceByKey((x,y)=>x+y).foreach(println)

```



```

acadgild@localhost:~
scala> val baseRDD = sc.textFile("/user/acadgild/dataset.txt").map(x => ((x.split(",")
") (0),x.split(",") (3) .toInt),1))
baseRDD: org.apache.spark.rdd.RDD[((String, Int), Int)] = MapPartitionsRDD[42] at m
ap at <console>:24

scala> val RDDfilter = baseRDD.filter(x=>x._1._1 == "Mathew" && x._1._2 == 55)
RDDfilter: org.apache.spark.rdd.RDD[((String, Int), Int)] = MapPartitionsRDD[43] at
filter at <console>:26

scala> val RDDreduce = RDDfilter.reduceByKey((x,y)=> x+y).foreach(println)
(Mathew,55),2)
RDDreduce: Unit = ()

scala>

```

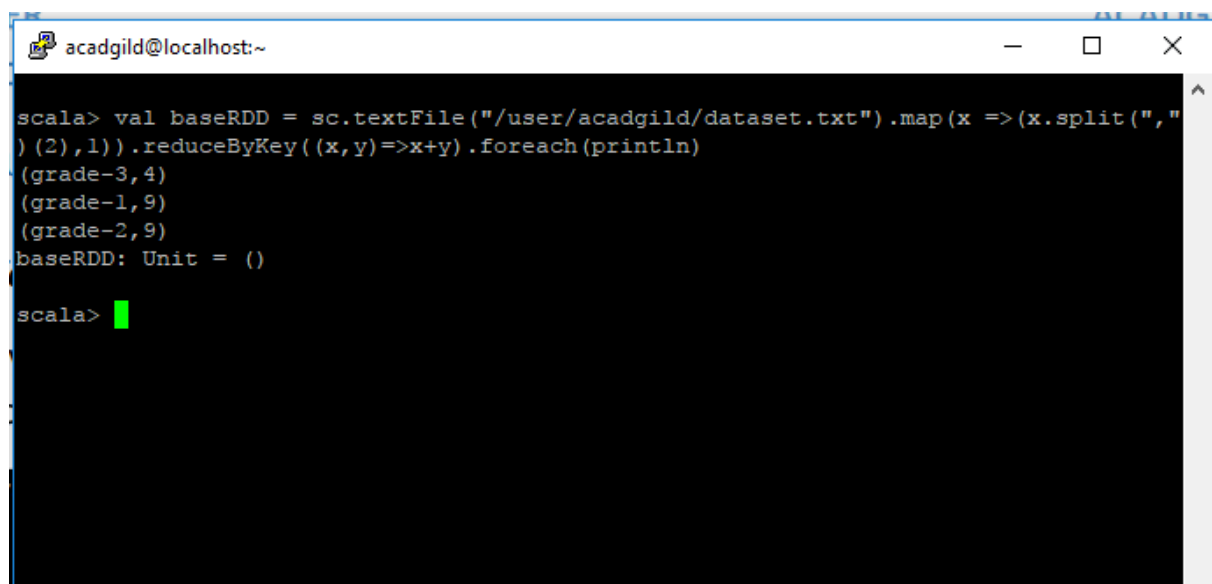
## Problem Statement 2:

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

```

val baseRDD = sc.textFile("/user/acadgild/dataset.txt").map(x =>
(x.split(",")(2),1)).reduceByKey((x,y)=>x+y).foreach(println)

```



```

acadgild@localhost:~
scala> val baseRDD = sc.textFile("/user/acadgild/dataset.txt").map(x => (x.split(",")
) (2),1) ).reduceByKey((x,y)=>x+y) .foreach(println)
(grade-3,4)
(grade-1,9)
(grade-2,9)
baseRDD: Unit = ()

scala>

```

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

```
val baseRDD =  
sc.textFile("/user/acadgild/dataset.txt").map(x=>((x.split(",")(0),x.split(",")  
(2)),x.split(",")(3).toInt))  
  
val RDDmap = baseRDD.mapValues(x=>(x,1))  
  
val RDDreduce = RDDmap.reduceByKey((x,y) => (x._1 + y._1, x._2 + y._2))  
  
val StudAvg = RDDreduce.mapValues{case(sum,count)=>(1.0*sum)/count}
```

acadgild@localhost:~

```
scala> val baseRDD = sc.textFile("/user/acadgild/dataset.txt").map(x =>((x.split(",  
") (0),x.split(",") (2)),x.split(",") (3).toInt))  
baseRDD: org.apache.spark.rdd.RDD[((String, String), Int)] = MapPartitionsRDD[29] a  
t map at <console>:24  
  
scala> val RDDmap = baseRDD.mapValues(x=>(x,1))  
RDDmap: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD  
[30] at mapValues at <console>:26  
  
scala> val RDDreduce = RDDmap.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))  
RDDreduce: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = ShuffledRDD[3  
1] at reduceByKey at <console>:28  
  
scala> val StudAvg = RDDreduce.mapValues{case(sum,count)=>(1.0*sum)/count}  
StudAvg: org.apache.spark.rdd.RDD[((String, String), Double)] = MapPartitionsRDD[32  
] at mapValues at <console>:30  
  
scala> StudAvg.foreach(println)  
(Lisa,grade-1),24.0  
(Mark,grade-2),17.5  
(Lisa,grade-2),61.0  
(Mathew,grade-3),45.0  
(Andrew,grade-2),77.0  
(Andrew,grade-1),43.666666666666664  
(Lisa,grade-3),86.0  
(John,grade-1),38.666666666666664  
(John,grade-2),74.0  
(Mark,grade-1),84.0  
(Andrew,grade-3),35.0  
(Mathew,grade-2),65.66666666666667  
  
scala> █
```

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


```
val baseRDD =
```

```
sc.textFile("/user/acadgild/dataset.txt").map(x=>((x.split(",")(0),x.split(",")(1)),x.split(",")(3).toInt))
```

```
val RDDmap = baseRDD.mapValues(x=>(x,1))
```

```
val RDDreduce = RDDmap.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))
```

```
val SubAvg = RDDreduce.mapValues{case(sum,count)=>(1.0*sum)/count}
```

 acadgild@localhost:~

```
scala> val baseRDD = sc.textFile("/user/acadgild/dataset.txt").map(x => ((x.split(",")
") (0),x.split(",") (1)),x.split(",") (3).toInt))
baseRDD: org.apache.spark.rdd.RDD[(String, String), Int]] = MapPartitionsRDD[11] a
t map at <console>:24

scala> val RDDmap = baseRDD.mapValues(x=>(x,1))
RDDmap: org.apache.spark.rdd.RDD[(String, String), (Int, Int))] = MapPartitionsRDD
[12] at mapValues at <console>:26

scala> val RDDreduce = RDDmap.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))
RDDreduce: org.apache.spark.rdd.RDD[(String, String), (Int, Int))] = ShuffledRDD[1
3] at reduceByKey at <console>:28

scala> val SubAvg = RDDreduce.mapValues{case(sum,count)=>(1.0*sum)/count}
SubAvg: org.apache.spark.rdd.RDD[(String, String), Double]] = MapPartitionsRDD[14]
at mapValues at <console>:30

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

scala> █
```

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


```
val baseRDD =
```

```
sc.textFile("/user/acadgild/dataset.txt").map(x=>((x.split(",")(1),x.split(",")
)(2)),x.split(",")(3).toInt))
```

```
val RDDmapvalue = baseRDD.mapValues(x=>(x,1))
```

```
val RDDreduce = RDDmapvalue.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))
```

```
val Avg_Grade =
RDDreduce.mapValues{case(sum,count)=>(1.0*sum)/count}.foreach(print
ln)
```

 acadgild@localhost:~

```
scala> val baseRDD = sc.textFile("/user/acadgild/dataset.txt").map(x => ((x.split(",
") (1),x.split(",") (2)),x.split(",") (3).toInt))
baseRDD: org.apache.spark.rdd.RDD[((String, String), Int)] = MapPartitionsRDD[17] a
t map at <console>:24

scala> val RDDmap = baseRDD.mapValues(x=>(x,1))
RDDmap: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD
[18] at mapValues at <console>:26

scala> val RDDreduce = RDDmap.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))
RDDreduce: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = ShuffledRDD[1
9] at reduceByKey at <console>:28

scala> val Avg_Grade = RDDreduce.mapValues{case(sum,count)=>(1.0*sum)/count}.foreac
h(println)
((history,grade-2),79.25)
((history,grade-3),86.0)
((maths,grade-1),46.0)
((science,grade-3),38.333333333333336)
((science,grade-1),50.0)
((science,grade-2),30.333333333333332)
((history,grade-1),51.666666666666664)
((maths,grade-2),48.5)
Avg_Grade: Unit = ()

scala> █
```

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

```
val baseRDD =
```

```
sc.textFile("/user/acadgild/dataset.txt").map(x=>((x.split(",")(0),x.split(",")
)(2)),x.split(",")(3).toInt))
```

```
val RDDmap = baseRDD.mapValues(x=>(x,1))
```

```
val RDDreduce = RDDmap.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))
```

```
val RDDavg = RDDreduce.mapValues{case(sum,count)=>(1.0*sum)/count}
```

```
val RDDfiltermap = RDDavg.filter(x=>x._1._2 == "grade-2" &&
x._2>50).count()
```



```
val RDDfiltermap = RDDavg.filter(x=>x._1._2 == "grade-2" &&
x._2>50).foreach(println)
```

acadgild@localhost:~

```
scala> val baseRDD = sc.textFile("/user/acadgild/dataset.txt").map(x => ((x.split(",")
(0), x.split(",") (2)), x.split(",") (3).toInt))
baseRDD: org.apache.spark.rdd.RDD[((String, String), Int)] = MapPartitionsRDD[35] a
t map at <console>:24

scala> val RDDmap = baseRDD.mapValues(x=>(x,1))
RDDmap: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD
[36] at mapValues at <console>:26

scala> val RDDreduce = RDDmap.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))
RDDreduce: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = ShuffledRDD[3
7] at reduceByKey at <console>:28

scala> val RDDavg = RDDreduce.mapValues{case (sum,count)=>(1.0*sum)/count}
RDDavg: org.apache.spark.rdd.RDD[((String, String), Double)] = MapPartitionsRDD[38]
at mapValues at <console>:30

scala> val RDDfiltermap = RDDavg.filter(x=>x._1._2 == "grade-2" && x._2>50).count()

RDDfiltermap: Long = 4

scala> val RDDfiltermap = RDDavg.filter(x=>x._1._2 == "grade-2" && x._2>50).foreach
(println)
((Lisa,grade-2),61.0)
((Andrew,grade-2),77.0)
((John,grade-2),74.0)
((Mathew,grade-2),65.66666666666667)
RDDfiltermap: Unit = ()

scala>
```

### 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

We created a paired RDD named as baseRDD1 by extracting only name and marks

```

val baseRDD1 =

sc.textFile("/user/acadgild/dataset.txt").map(x=>(x.split(",")(0),x.split(",")
(3).toInt))

val studAvg = baseRDD1.mapValues(x=>(x,1)).foreach(println)

val studReduce = studAvg.reduceByKey((x,y)=> (x._1+y._1,x._2+y._2))

val Avg_Stud = studReduce.mapValues{case (sum,count) => (1.0 *
sum)/count}

```

 acadgild@localhost:~

```

scala> val baseRDD1 = sc.textFile("/user/acadgild/dataset.txt").map(x=>(x.split(
",") (0),x.split(",") (3).toInt))
baseRDD1: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[2] at map a
t <console>:24

scala> val studAvg = baseRDD1.mapValues(x=>(x,1))
studAvg: org.apache.spark.rdd.RDD[(String, (Int, Int))] = MapPartitionsRDD[3] at
mapValues at <console>:26

scala> val studReduce = studAvg.reduceByKey((x,y)=> (x._1+y._1,x._2+y._2))
studReduce: org.apache.spark.rdd.RDD[(String, (Int, Int))] = ShuffledRDD[4] at r
educeByKey at <console>:28

scala> val Avg_Stud = studReduce.mapValues{case (sum,count) => (1.0 * sum)/count
}
Avg_Stud: org.apache.spark.rdd.RDD[(String, Double)] = MapPartitionsRDD[5] at ma
pValues at <console>:30

scala> Avg_Stud.foreach(println)
(Mark,50.75)
(Andrew,46.333333333333336)
(Mathew,60.5)
(John,47.5)
(Lisa,58.0)

scala> █

```

we are creating another paired RDD named as baseRDD2 by extracting name and grade as key and marks as value from the input file,

```

val baseRDD2 =

sc.textFile("/user/acadgild/dataset.txt").map(x=>((x.split(",")(0),x.split(",")
)(2)),x.split(",")(3).toInt)).foreach(println)

val grade = baseRDD2.mapValues(x=>(x,1))

val gradeReduce = grade.reduceByKey((x,y)=> (x._1+y._1,x._2+y._2))

```

```
val gradeAvg = gradeReduce.mapValues{case(sum,count) =>
(1.0*sum)/count}
```

 acadgild@localhost:~

```
scala> val baseRDD2 = sc.textFile("/user/acadgild/dataset.txt").map(x=>((x.split(",") (0),x.spli
t(",") (2)),x.split(",") (3).toInt))
baseRDD2: org.apache.spark.rdd.RDD[(String, String), Int] = MapPartitionsRDD[14] at map at <c
onsole>:24

scala> val grade = baseRDD2.mapValues(x=>(x,1))
grade: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = MapPartitionsRDD[15] at mapVa
lues at <console>:26

scala> val gradeReduce = grade.reduceByKey((x,y)=> (x._1+y._1,x._2+y._2))
gradeReduce: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = ShuffledRDD[16] at redu
ceByKey at <console>:28

scala> val gradeAvg = gradeReduce.mapValues{case(sum,count) => (1.0*sum)/count}
gradeAvg: org.apache.spark.rdd.RDD[(String, String), Double] = MapPartitionsRDD[17] at mapVal
ues at <console>:30

scala> gradeAvg.foreach(println)
(Lisa,grade-1),24.0)
(Mark,grade-2),17.5)
(Lisa,grade-2),61.0)
(Mathew,grade-3),45.0)
(Andrew,grade-2),77.0)
(Andrew,grade-1),43.666666666666664)
(Lisa,grade-3),86.0)
(John,grade-1),38.666666666666664)
(John,grade-2),74.0)
(Mark,grade-1),84.0)
(Andrew,grade-3),35.0)
(Mathew,grade-2),65.666666666666667)

scala> █
```

In below step we are using intersection function between flatgradeAvg and flatnameAvg rdd's to find whether any common student is there.

```
val flatgradeAvg = gradeAvg.map(x=> x._1._1 + "," + x._2.toDouble)
```

```
flatgradeAvg.foreach(println)
```

```
val flatAvg_Stud = Avg_Stud.map(x=>x._1+","+x._2)
```

```
flatAvg_Stud.foreach(println)
```

acadmild@localhost:~

```
scala> val flatgradeAvg = gradeAvg.map(x=> x._1._1 + "," + x._2.toDouble)
flatgradeAvg: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[21] at map at <console>:32

scala> flatgradeAvg.foreach(println)
Lisa,24.0
Mark,17.5
Lisa,61.0
Mathew,45.0
Andrew,77.0
Andrew,43.666666666666664
Lisa,86.0
John,38.666666666666664
John,74.0
Mark,84.0
Andrew,35.0
Mathew,65.666666666666667

scala> val flatAvg_Stud = Avg_Stud.map(x=>x._1+","+x._2)
flatAvg_Stud: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[22] at map at <console>:32

scala> flatAvg_Stud.foreach(println)
Mark,50.75
Andrew,46.333333333333336
Mathew,60.5
John,47.5
Lisa,58.0

scala>
```

**val commanval = flatgradeAvg.intersection(flatAvg\_Stud)**

**commanval.foreach(println)**

acadmild@localhost:~

```
scala> val commanval = flatgradeAvg.intersection(flatAvg_Stud)
commanval: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[28] at intersection at <console>:44
:44

scala> commanval.foreach(println)

scala>
```