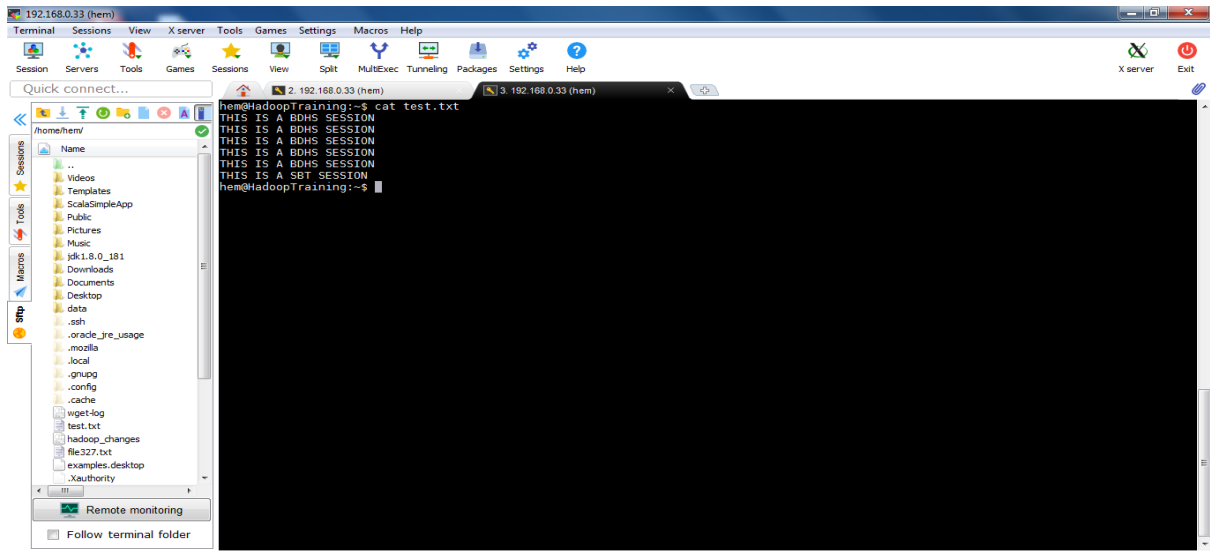


## Task 1

### Text File Used for Task 1



The screenshot shows a MobaXterm window with a terminal session. The terminal title bar indicates the connection is to 192.168.0.33 (hem). The terminal content shows a user at a Hadoop training machine running the command `cat test.txt`. The output of the command is a list of session identifiers: `THIS IS A BDHS SESSION` repeated five times, followed by `THIS IS A SBT SESSION`. The left sidebar of the MobaXterm window displays a file explorer view of the user's home directory, showing various folders like Videos, Templates, and files like `test.txt` and `hadoop_changes`. At the bottom of the window, there is a notice for the unregistered version of MobaXterm.

```
hem@HadoopTraining:~$ cat test.txt
THIS IS A BDHS SESSION
THIS IS A BDHS SESSION
THIS IS A BDHS SESSION
THIS IS A BDHS SESSION
THIS IS A BDHS SESSION
THIS IS A SBT SESSION
hem@HadoopTraining:~$
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

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

Spark Solution-:

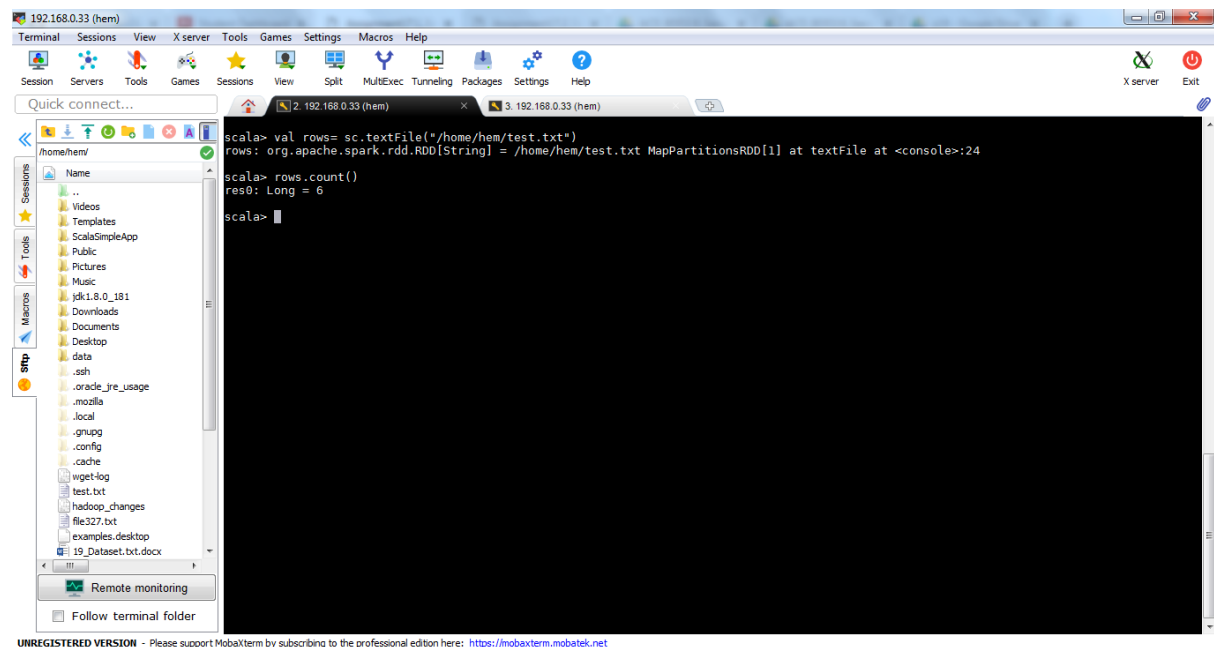
```
scala> val rows= sc.textFile("/home/hem/test.txt")
```

```
rows: org.apache.spark.rdd.RDD[String] = /home/hem/test.txt MapPartitionsRDD[1] at textFile at <console>:24
```

```
scala> rows.count()
```

```
res0: Long = 6
```

Output-:



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

Spark Solution-:

```
scala> val rows= sc.textFile("/home/hem/test.txt")
```

```
rows: org.apache.spark.rdd.RDD[String] = /home/hem/test.txt MapPartitionsRDD[3] at textFile at <console>:24
```

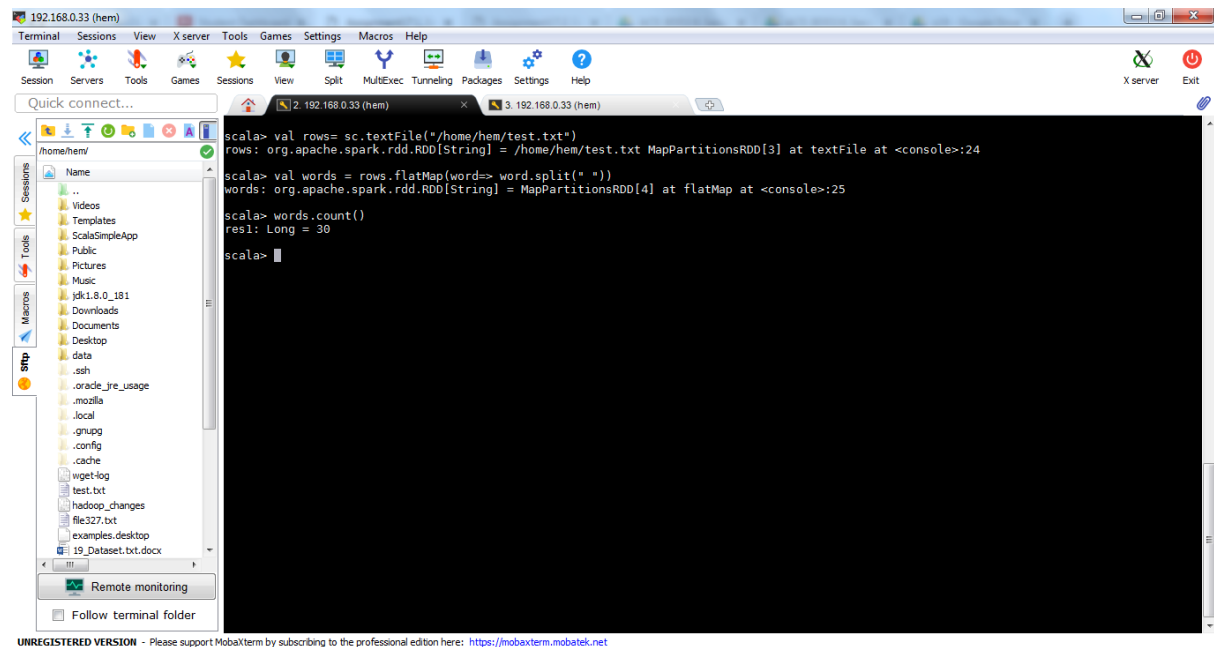
```
scala> val words = rows.flatMap(word=> word.split(" "))
```

```
words: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[4] at flatMap at <console>:25
```

```
scala> words.count()
```

```
res1: Long = 30
```

Output:-



The screenshot shows a MobaXterm window with a terminal session. The terminal output is as follows:

```
scala> val rows= sc.textFile("/home/hem/test.txt")
rows: org.apache.spark.rdd.RDD[String] = /home/hem/test.txt MapPartitionsRDD[3] at textFile at <console>:24

scala> val words = rows.flatMap(word=> word.split(" "))
words: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[4] at flatMap at <console>:25

scala> words.count()
res1: Long = 30

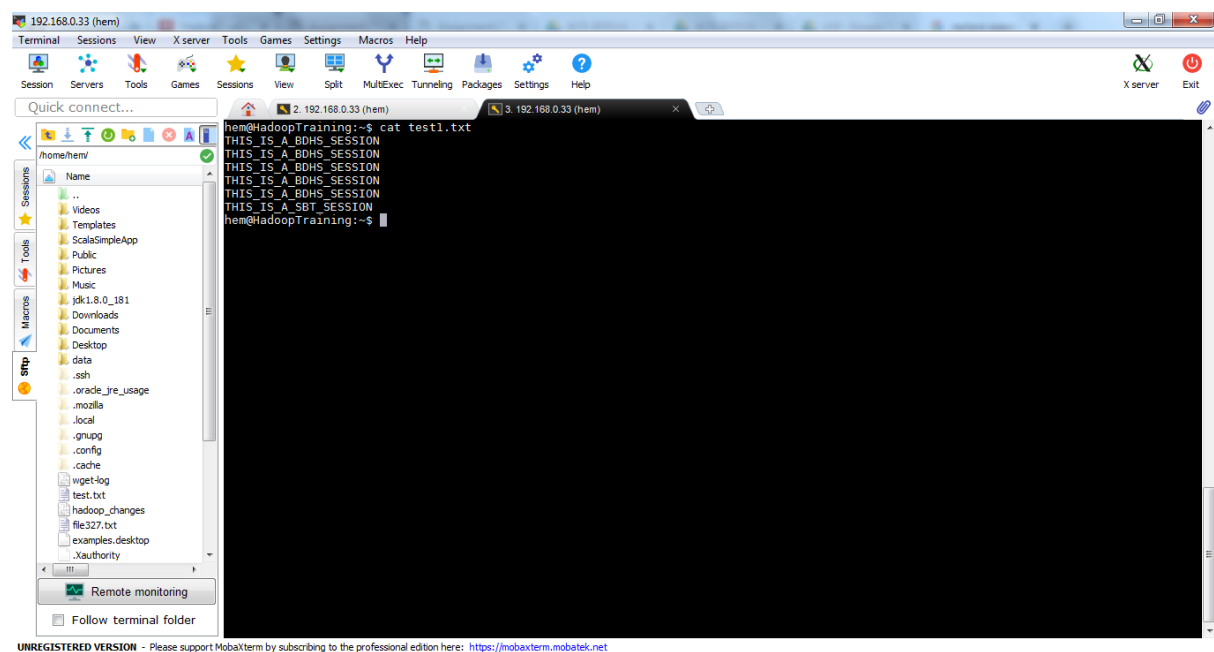
scala>
```

The left sidebar shows a file explorer for the /home/hem directory, listing various files and folders including test.txt, hadoop\_changes, file327.txt, and examples.desktop.

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.

## Task 2

Text Document:-



The screenshot shows a MobaXterm window with a terminal session. The terminal output is as follows:

```
hem@HadoopTraining:~$ cat test1.txt
THIS_IS_A_BOHS_SESSION
THIS_IS_A_BOHS_SESSION
THIS_IS_A_BOHS_SESSION
THIS_IS_A_BOHS_SESSION
THIS_IS_A_BOHS_SESSION
THIS_IS_A_BOHS_SESSION
THIS_IS_A_SBT_SESSION
hem@HadoopTraining:~$
```

The left sidebar shows a file explorer for the /home/hem directory, listing various files and folders including test.txt, hadoop\_changes, file327.txt, and examples.desktop.

Spark Solution-:

```
scala> val base1 = sc.textFile("/home/hem/test1.txt")
```

```
base1: org.apache.spark.rdd.RDD[String] = /home/hem/test1.txt MapPartitionsRDD[14] at textFile at <console>:24
```

```
scala> val words = base1.flatMap(word=> word.split("-"))
```

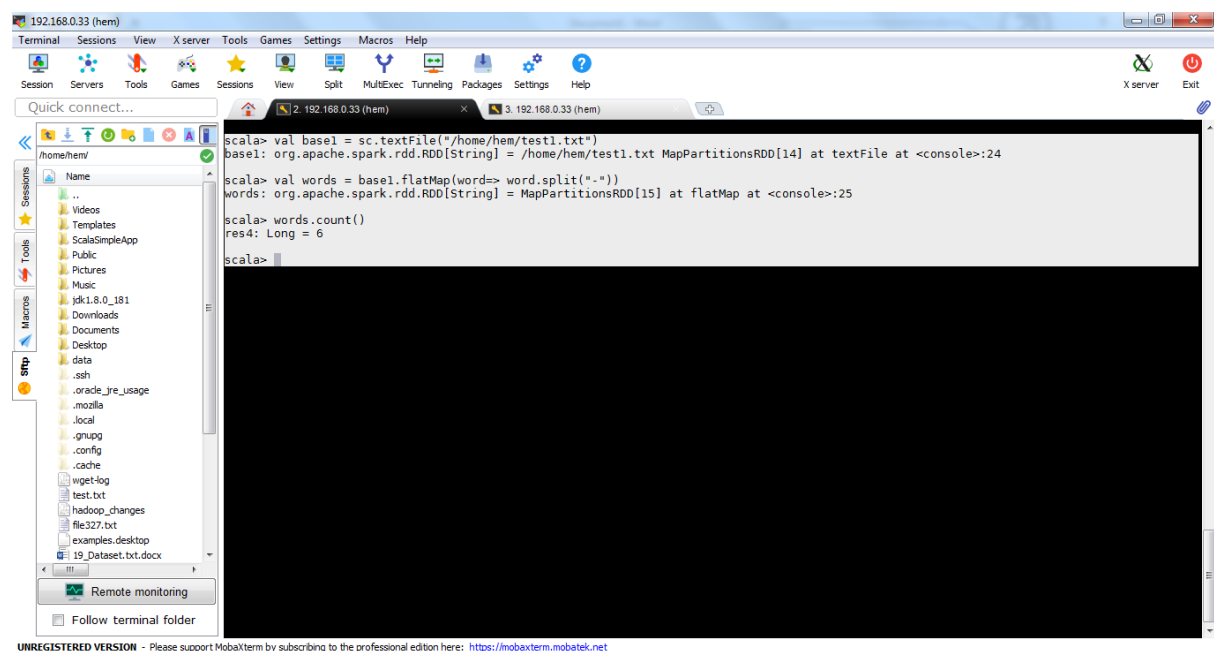
```
words: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[15] at flatMap at <console>:25
```

```
scala> words.count()
```

```
res4: Long = 6
```

```
scala>
```

Output-:



```
scala> val base1 = sc.textFile("/home/hem/test1.txt")
base1: org.apache.spark.rdd.RDD[String] = /home/hem/test1.txt MapPartitionsRDD[14] at textFile at <console>:24

scala> val words = base1.flatMap(word=> word.split("-"))
words: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[15] at flatMap at <console>:25

scala> words.count()
res4: Long = 6

scala>
```

### Problem Statement 1:

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

Spark Solution-:

```
scala> val rows = sc.textFile("/home/hem/19_Dataset.txt").map(x =>
```

```
  | (x.split(",")(0),(x.split(",")(1),x.split(",")(2),x.split(",")(3).toInt,x.split(",")(4).toInt)))
```

```
rows: org.apache.spark.rdd.RDD[(String, (String, String, Int, Int))] = MapPartitionsRDD[18] at map at <console>:24
```

```
scala> rows.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))
```

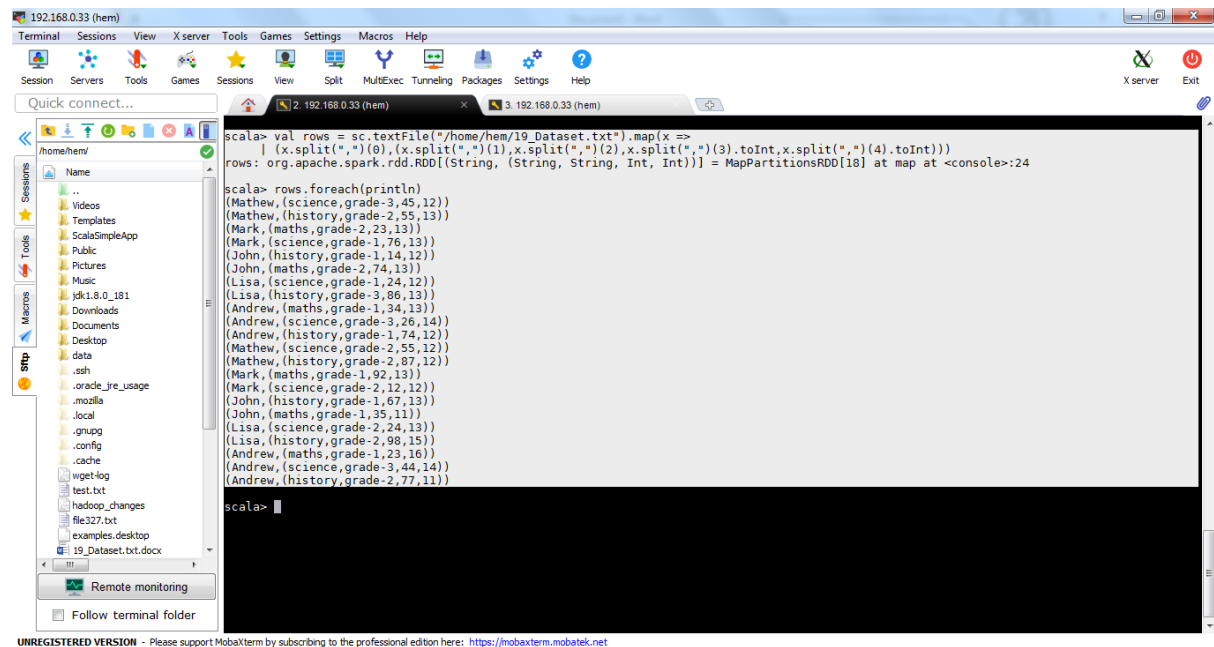
```
(Lisa,(history,grade-2,98,15))
```

```
(Andrew,(maths,grade-1,23,16))
```

(Andrew,(science,grade-3,44,14))

(Andrew,(history,grade-2,77,11))

Output:-



The screenshot shows a MobaXterm terminal window with a file explorer on the left. The terminal displays the following Scala code and its output:

```
scala> val rows = sc.textFile("/home/hem/19_Dataset.txt").map(x =>
  | (x.split(",")(0), (x.split(",")(1), x.split(",")(2), x.split(",")(3).toInt, x.split(",")(4).toInt))
rows: org.apache.spark.rdd.RDD[(String, (String, String, Int, Int))] = MapPartitionsRDD[18] at map at <console>:24

scala> rows.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.

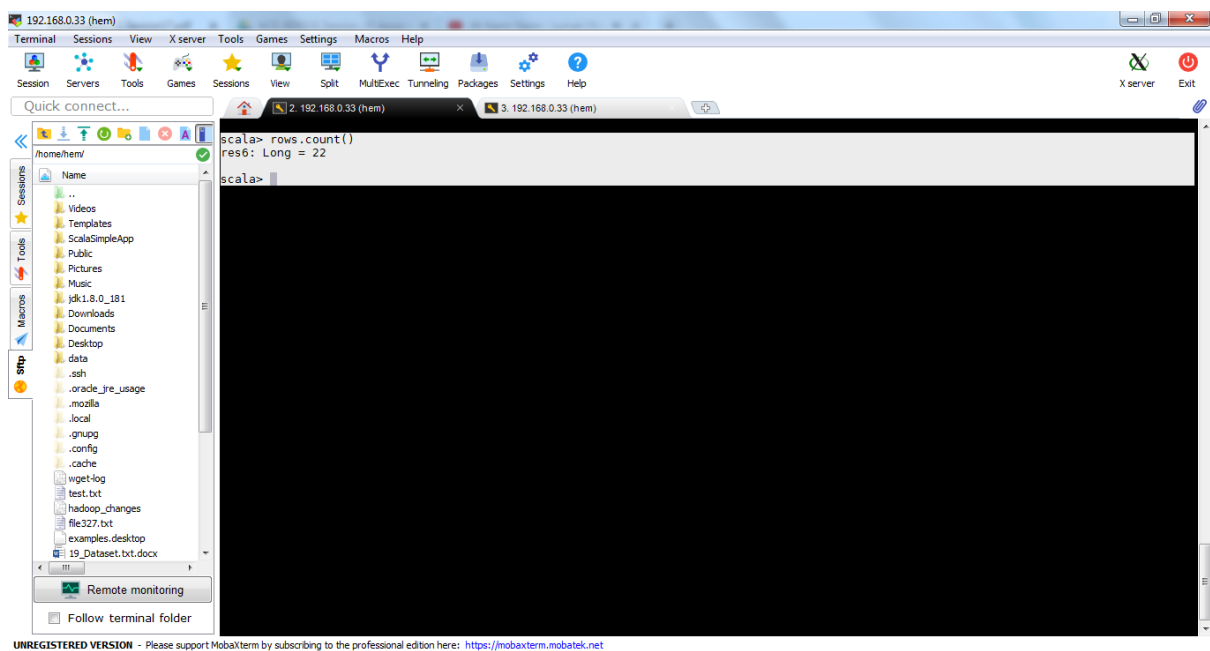
Spark Solution:-

```
scala> rows.count()
```

res6: Long = 22

```
scala>
```

Output:-



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

Spark Solution:-

```
scala> val rows = sc.textFile("/home/hem/19_Dataset.txt").map(x=> (x.split(",")(1),1))
```

```
rows: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[21] at map at <console>:24
```

```
scala> val output = rows.reduceByKey((x,y)=>(x+y))
```

```
output: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[22] at reduceByKey at <console>:25
```

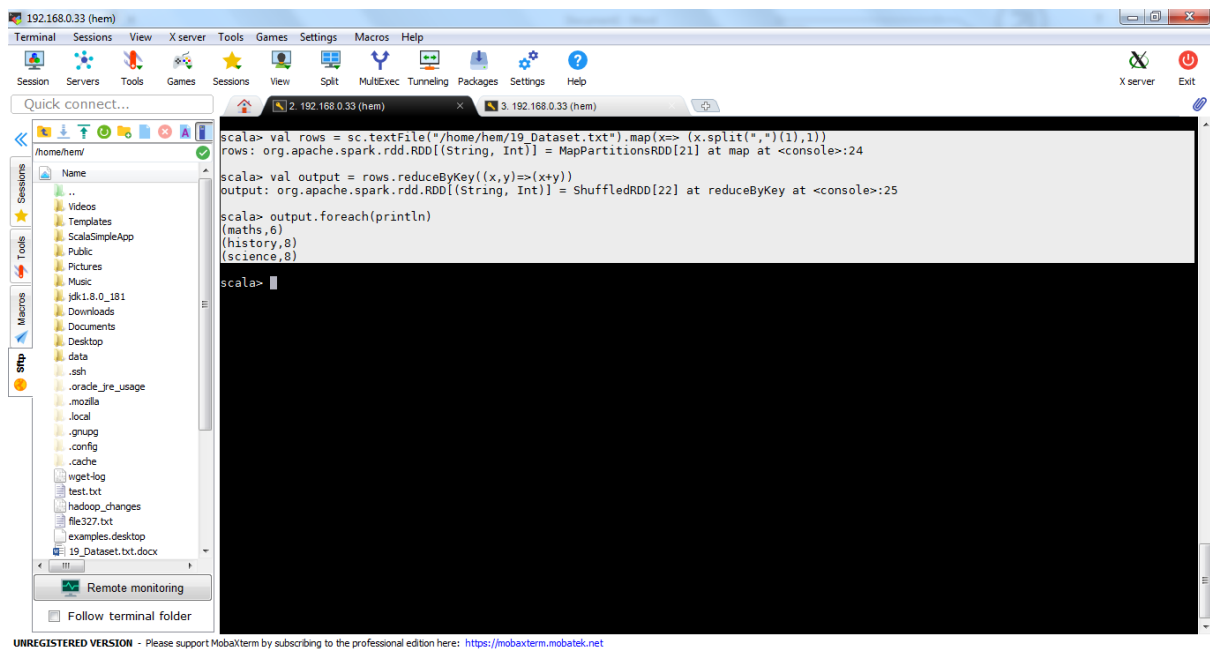
```
scala> output.foreach(println)
```

```
(maths,6)
```

```
(history,8)
```

```
(science,8)
```

Output:-



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

Spark Solution:-

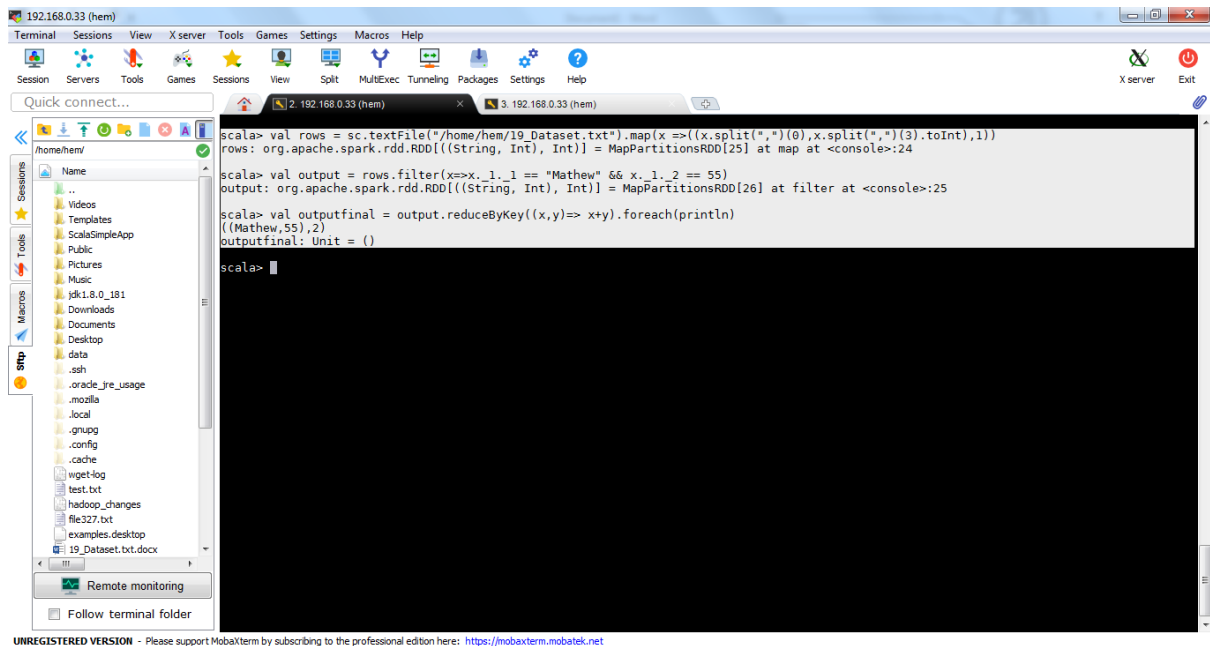
```

val rows = sc.textFile("/home/hem/19_Dataset.txt").map(x=>((x.split(",")(0),x.split(",")(3).toInt),1))
val output = rows.filter(x=>x._1._1 == "Mathew" && x._1._2 == 55)
val outputfinal = output.reduceByKey((x,y)=> x+y).foreach(println)

```

Output:-





```
scala> val rows = sc.textFile("/home/hem/19_Dataset.txt").map(x => (x.split(",")(0), x.split(",")(3).toInt), 1)
rows: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[25] at map at <console>:24

scala> val output = rows.filter(x => x._1 == "Mathew" && x._1._2 == 55)
output: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[26] at filter at <console>:25

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

scala>
```

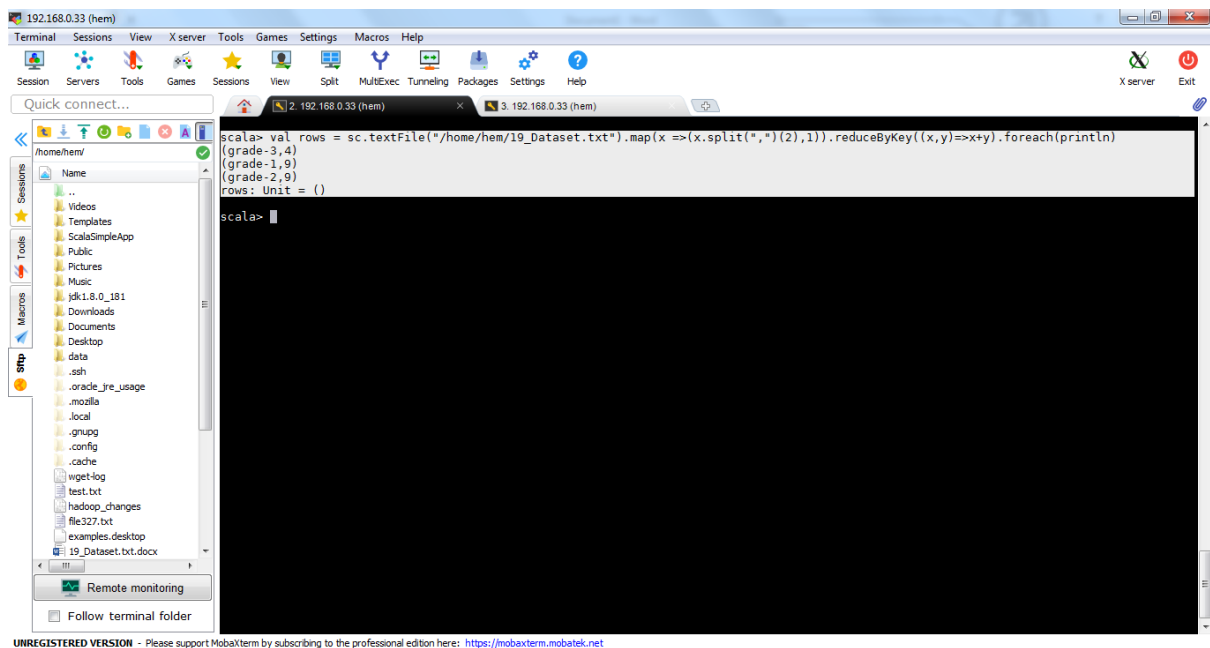
## Problem Statement 2:

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

Spark Solution:-

```
val rows = sc.textFile("/home/hem/19_Dataset.txt").map(x  
=>(x.split(",")(2),1)).reduceByKey((x,y)=>x+y).foreach(println)
```

Output:-



```
scala> val rows = sc.textFile("/home/hem/19_Dataset.txt").map(x => (x.split(",")(2), 1)).reduceByKey((x, y) => x + y).foreach(println)
(grade-3, 4)
(grade-1, 9)
(grade-2, 9)
rows: Unit = ()

scala>
```

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

Spark Solution-:

```
val rows
=sc.textFile("/home/hem/19_Dataset.txt").map(x=>((x.split(",")(0),x.split(",")(2)),x.split(",")(3).toInt))

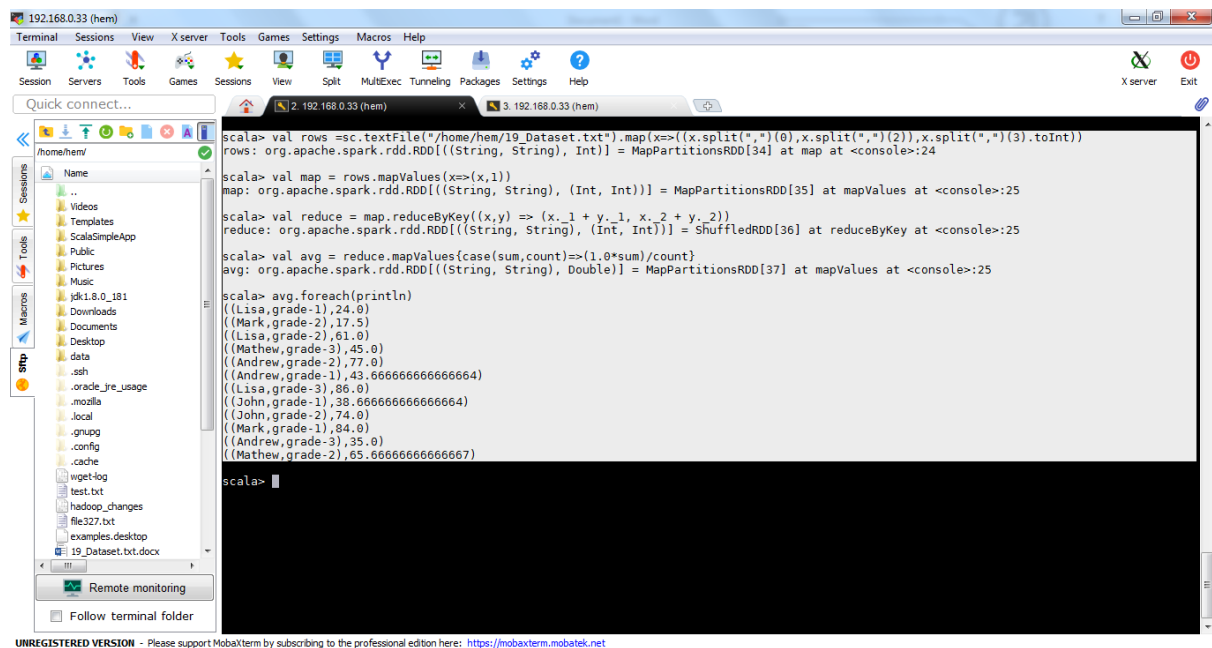
val map = rows.mapValues(x=>(x,1))

val reduce = map.reduceByKey((x,y) => (x._1 + y._1, x._2 + y._2))

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

avg.foreach(println)
```

Output-:



```
scala> val rows = sc.textFile("/home/hem/19_Dataset.txt").map(x=>((x.split(",")(0),x.split(",")(2)),x.split(",")(3).toInt))
rows: org.apache.spark.rdd.RDD[(String, String), Int] = MapPartitionsRDD[34] at map at <console>:24

scala> val map = rows.mapValues(x=>(x,1))
map: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = MapPartitionsRDD[35] at mapValues at <console>:25

scala> val reduce = map.reduceByKey((x,y) => (x._1 + y._1, x._2 + y._2))
reduce: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = ShuffledRDD[36] at reduceByKey at <console>:25

scala> val avg = reduce.mapValues{case(sum,count)=>(1.0*sum)/count}
avg: org.apache.spark.rdd.RDD[(String, String), Double] = MapPartitionsRDD[37] at mapValues at <console>:25

scala> avg.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>
```

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

Spark Solution-:

```
val rows
=sc.textFile("/home/hem/19_Dataset.txt").map(x=>((x.split(",")(0),x.split(",")(1)),x.split(",")(3).toInt))

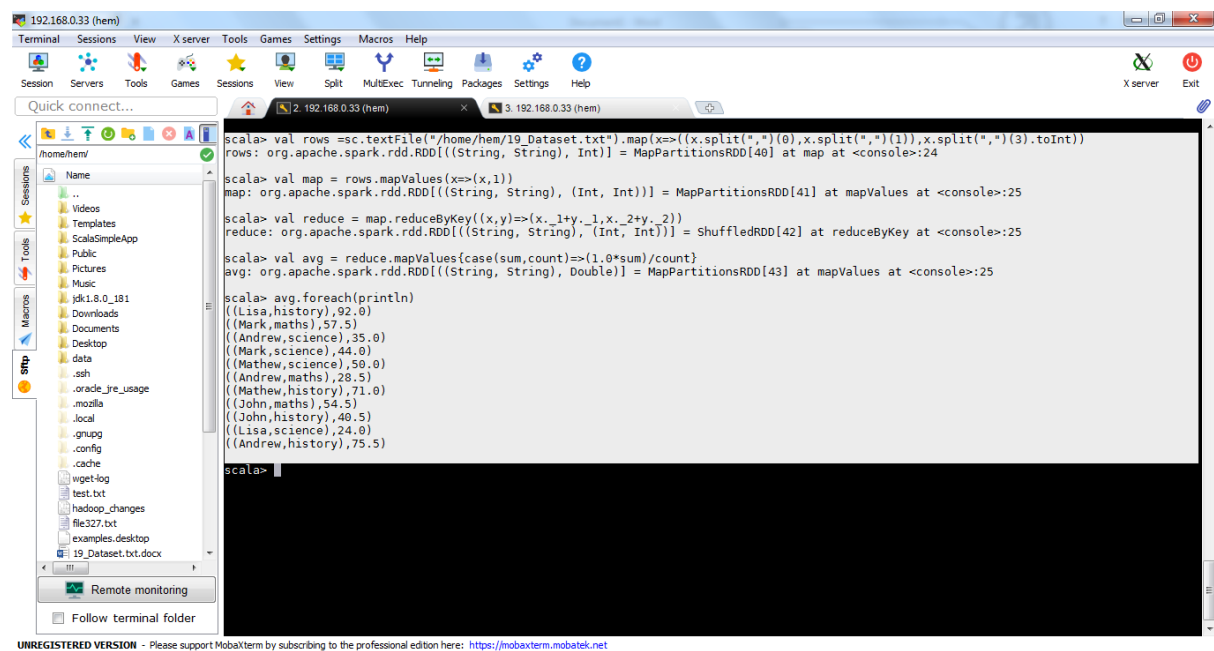
val map = rows.mapValues(x=>(x,1))

val reduce = map.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))

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

avg.foreach(println)
```

Output:-



```
scala> val rows = sc.textFile("/home/hem/19_Dataset.txt").map(x=>((x.split(",")(0),x.split(",")(1)),x.split(",")(3).toInt))
rows: org.apache.spark.rdd.RDD[(String, String), Int] = MapPartitionsRDD[40] at map at <console>:24

scala> val map = rows.mapValues(x=>(x,1))
map: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = MapPartitionsRDD[41] at mapValues at <console>:25

scala> val reduce = map.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))
reduce: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = ShuffledRDD[42] at reduceByKey at <console>:25

scala> val avg = reduce.mapValues{case(sum,count)=>(1.0*sum)/count}
avg: org.apache.spark.rdd.RDD[(String, String), Double] = MapPartitionsRDD[43] at mapValues at <console>:25

scala> avg.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?

Spark Solution:-

```
val rows
= sc.textFile("/home/hem/19_Dataset.txt").map(x=>((x.split(",")(1),x.split(",")(2)),x.split(",")(3).toInt))

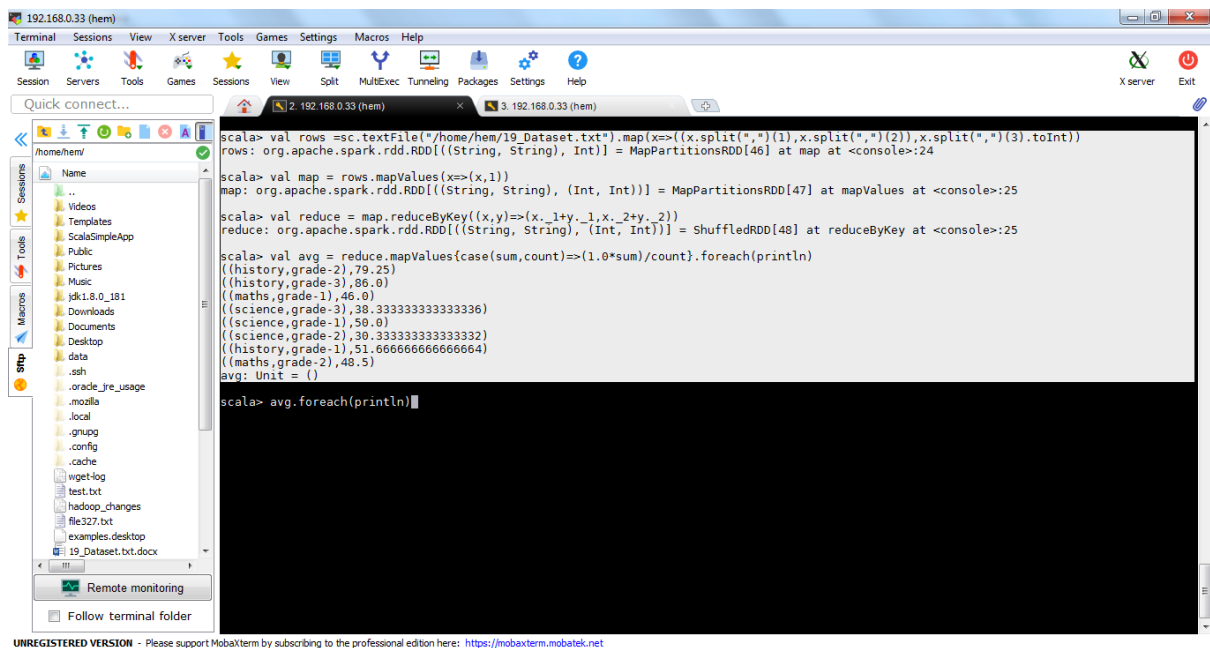
val map = rows.mapValues(x=>(x,1))

val reduce = map.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))

val avg = reduce.mapValues{case(sum,count)=>(1.0*sum)/count}.foreach(println)

avg.foreach(println)
```

Output:-



```
scala> val rows = sc.textFile("/home/hem/19_Dataset.txt").map(x=>((x.split(",")(1),x.split(",")(2)),x.split(",")(3).toInt))
rows: org.apache.spark.rdd.RDD[(String, String), Int] = MapPartitionsRDD[46] at map at <console>:24

scala> val map = rows.mapValues(x=>(x,1))
map: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = MapPartitionsRDD[47] at mapValues at <console>:25

scala> val reduce = map.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))
reduce: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = ShuffledRDD[48] at reduceByKey at <console>:25

scala> val avg = reduce.mapValues{case(sum,count)=>(1.0*sum)/count}.foreach(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: Unit = ()

scala> avg.foreach(println)
```

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

Spark Solution:-

```
val rows
= sc.textFile("/home/hem/19_Dataset.txt").map(x=>((x.split(",")(0),x.split(",")(2)),x.split(",")(3).toInt))

val map = rows.mapValues(x=>(x,1))

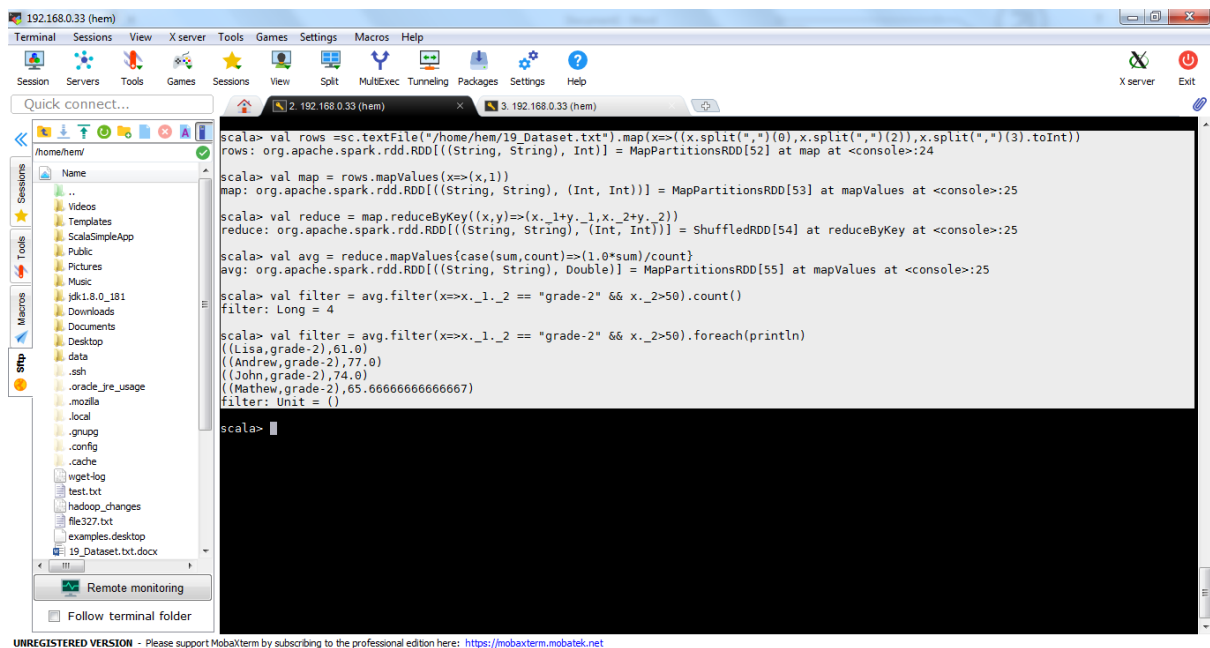
val reduce = map.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))

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

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

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

Output:-



```
scala> val rows = sc.textFile("/home/hem/19_Dataset.txt").map(x=>{(x.split(",")(0),x.split(",")(2)),x.split(",")(3).toInt})
rows: org.apache.spark.rdd.RDD[(String, String), Int] = MapPartitionsRDD[52] at map at <console>:24

scala> val map = rows.mapValues(x=>(x,1))
map: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = MapPartitionsRDD[53] at mapValues at <console>:25

scala> val reduce = map.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))
reduce: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = ShuffledRDD[54] at reduceByKey at <console>:25

scala> val avg = reduce.mapValues{case (sum,count)=>(1.0*sum)/count}
avg: org.apache.spark.rdd.RDD[(String, String), Double] = MapPartitionsRDD[55] at mapValues at <console>:25

scala> val filter = avg.filter(x=>x._1._2 == "grade-2" && x._2>50).count()
filter: Long = 4

scala> val filter = avg.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)
filter: 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

Spark Solution-:

Average Score Per Student Name Across All Grades

```
val row1 =sc.textFile("/home/hem/19_Dataset.txt").map(x=>(x.split(",")(0),x.split(",")(3).toInt))
```

```
val map1 = row1.mapValues(x=>(x,1))
```

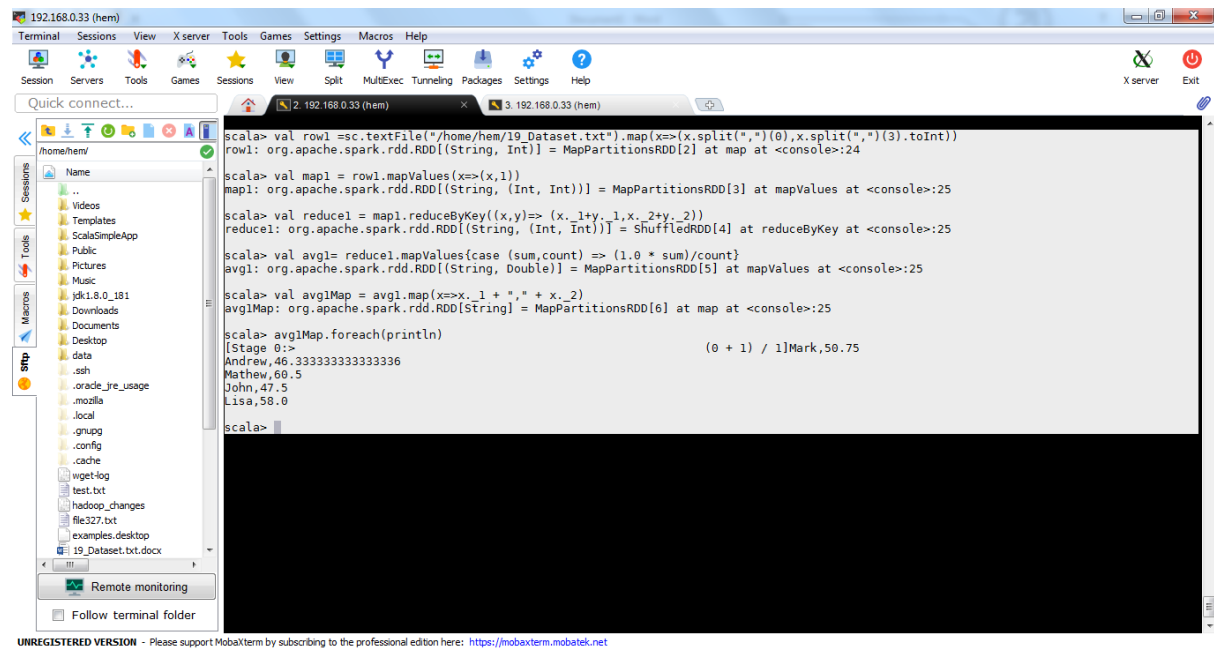
```
val reduce1 = map1.reduceByKey((x,y)=> (x._1+y._1,x._2+y._2))
```

```
val avg1= reduce1.mapValues{case (sum,count) => (1.0 * sum)/count}
```

```
val avg1Map = avg1.map(x=>x._1 + "," + x._2)
```

```
avg1Map.foreach(println)
```

Output:-



```
scala> val row1=sc.textFile("/home/hem/19_Dataset.txt").map(x=>(x.split(",")(0),x.split(",")(3).toInt))
row1: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[2] at map at <console>:24

scala> val map1 = row1.mapValues(x=>(x,1))
map1: org.apache.spark.rdd.RDD[(String, (Int, Int))] = MapPartitionsRDD[3] at mapValues at <console>:25

scala> val reduce1 = map1.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))
reduce1: org.apache.spark.rdd.RDD[(String, (Int, Int))] = ShuffledRDD[4] at reduceByKey at <console>:25

scala> val avg1= reduce1.mapValues(case (sum,count) => (1.0 * sum)/count)
avg1: org.apache.spark.rdd.RDD[(String, Double)] = MapPartitionsRDD[5] at mapValues at <console>:25

scala> val avg1Map = avg1.map(x=>x._1 + "," + x._2)
avg1Map: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[6] at map at <console>:25

scala> avg1Map.foreach(println)
[Stage 0:]
(0 + 1) / 1]Mark,50.75
Andrew,46.333333333333336
Mathew,60.5
John,47.5
Lisa,58.0

scala>
```

Average Score Per Student Name Per Grade

Spark Solution:-

```
val row2
=sc.textFile("/home/hem/19_Dataset.txt").map(x=>((x.split(",")(0),x.split(",")(2)),x.split(",")(3).toInt))

val map2 = row2.mapValues(x=>(x,1))

val reduce2 = map2.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))

val avg2 = reduce2.mapValues(case(sum,count) => (1.0*sum)/count)

val avg2Map = avg2.map(x=>x._1 + "," + x._2.toDouble)

avg2Map.foreach(println)
```

Output:-

The screenshot shows a MobaXterm terminal window with a file explorer on the left. The terminal displays the following Scala code and its output:

```
scala> val row2 = sc.textFile("/home/hem/19_Dataset.txt").map(x=>((x.split(",")(0),x.split(",")(2)),x.split(",")(3).toInt))
row2: org.apache.spark.rdd.RDD[(String, String), (Int)] = MapPartitionsRDD[9] at map at <console>:24

scala> val map2 = row2.mapValues(x=>(x,1))
map2: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = MapPartitionsRDD[10] at mapValues at <console>:25

scala> val reduce2 = map2.reduceByKey((x,y)=>(x._1+y._1,x._2+y._2))
reduce2: org.apache.spark.rdd.RDD[(String, String), (Int, Int)] = ShuffledRDD[11] at reduceByKey at <console>:25

scala> val avg2 = reduce2.mapValues(case(sum,count) => (1.0*sum)/count)
avg2: org.apache.spark.rdd.RDD[(String, String), (Double)] = MapPartitionsRDD[12] at mapValues at <console>:25

scala> val avg2Map = avg2.map(x=> x._1._1 + ", " + x._2.toDouble)
avg2Map: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[13] at map at <console>:25

scala> avg2Map.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>
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

InterSection:-

Spark Solution:-

```
val intersection = avg2Map.intersection(avg1Map)
```

```
intersection.foreach(println)
```

Output:-

The screenshot shows a MobaXterm terminal window with a file explorer on the left. The terminal displays the following Scala code and its output:

```
scala> val intersection = avg2Map.intersection(avg1Map)
intersection: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[19] at intersection at <console>:27

scala> intersection.foreach(println)

scala>
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

There is no output as nothing is common