Problem Statement

Given a dataset of college students as a text file (name, subject, grade, marks):

Dataset:- 17.2_Dataset.txt

```
[acadgild@localhost ~]$ cat /home/acadgild/Downloads/17.2_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[acadgild@localhost ~]$
```

Problem Statement 1:

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

Code:

```
scala> val student_data = sc.textFile("/home/acadgild/Downloads/17.2_Dataset.txt")
student_data: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[15] at textFile at <console>:27

scala> val tupleRDD = student_data.map(x =>{
     |
     |  val student_datas = x.split(",")
     |
     |  val name = student_datas(0)
     |
     |  val subject = student_datas(1)
     |
     |  val grade = student_datas(2)
     |
     |  val marks1 = student_datas(3).toInt
     |
     |  val marks2 = student_datas(4).toInt
     |
     |  val tuple = (name,subject,grade,marks1,marks2)
     |
     | tuple})
tupleRDD: org.apache.spark.rdd.RDD[(String, String, String, Int, Int)] = MapPartitionsRDD[16] at map at <console>:29

scala> tupleRDD.foreach(println)
```

Output:

```
scala> tupleRDD.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>
```

- Find the count of total number of rows present.

Code:

```
scala> val count_rows = student_data.count()
count_rows: Long = 22
```

Output:

```
scala> println(count_rows)
22
```

- What is the distinct number of subjects present in the entire school

Code:

```
scala> val student_data = sc.textFile("/home/acadgild/Downloads/17.2_Dataset.txt")
student_data: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[20] at textFile at <console>:27

scala> val distinct_subjectsRDD = student_data.map(x =>{
     |
     |  val student_datas = x.split(",")
     |
     |  val sub = student_datas(1)
     |
     |  sub
     |
     | })
distinct_subjectsRDD: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[21] at map at <console>:29
```

Output:

```
scala> val count_distint_sub = distinct_subjectsRDD.distinct().count()
count_distint_sub: Long = 3

scala>
```

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

Code:

```
scala> val student_data = sc.textFile("/home/acadgild/Downloads/17.2_Dataset.txt")
student_data: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[26] at textFile at <console>:27

scala> val transRDD = student_data.map(x => x.split(","))
transRDD: org.apache.spark.rdd.RDD[Array[String]] = MapPartitionsRDD[27] at map at <console>:29

scala> val filterRDD = transRDD.filter(x=>((x(0).toLowerCase == "mathew")&&(x(3).toInt == 55)))
filterRDD: org.apache.spark.rdd.RDD[Array[String]] = MapPartitionsRDD[28] at filter at <console>:31
```

Output:

```
scala> println(filterRDD.count)
2

scala>
```

Problem Statement 2:

- What is the count of students per grade in the school?

Code:

```
scala> val studentRDD = student_data.map(x => {
 val student_datas = x.split(",")
 val grade = student_datas(2)
 (grade)
})
studentRDD: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[29] at map at <console>:29

scala> val finalGradeRDD = studentRDD.map(x=>(x,1))
finalGradeRDD: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[30] at map at <console>:31

scala> val result = finalGradeRDD.reduceByKey(_ + _)
result: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[31] at reduceByKey at <console>:33
```

Output:

```
scala> result.collect()
res7: Array[(String, Int)] = Array((grade-3,4), (grade-1,9), (grade-2,9))
```

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

Code:

```
scala> val studentRDD = student_data.map(x => {
     |
     |  val student_datas  = x.split(",")
     |
     |  val name = student_datas (0)
     |
     |  val grade = student_datas (2)
     |
     |  val marks = student_datas (3).toInt
     |
     | ((name,grade),marks)})
studentRDD: org.apache.spark.rdd.RDD[((String, String), Int)] = MapPartitionsRDD[32] at map at <console>:29

scala> val countRDD = studentRDD.map(x =>((x._1),1)).reduceByKey(_+_).sortByKey()
countRDD: org.apache.spark.rdd.RDD[((String, String), Int)] = ShuffledRDD[35] at sortByKey at <console>:31

scala>

scala> val totMarksRDD = studentRDD.reduceByKey(_+_).sortByKey()
totMarksRDD: org.apache.spark.rdd.RDD[((String, String), Int)] = ShuffledRDD[37] at sortByKey at <console>:31

scala>

scala> val totJoinCount = totMarksRDD.join(countRDD)
totJoinCount: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD[40] at join at <console>:35

scala>
```

Output:

```
scala> val result = totJoinCount.map(x => ( ( x._1.toString)+" ===> "+(x._2._1.toInt)/(x._2._2.toInt))).foreach(println)
(Lisa,grade-1) ===> 24
(Mark,grade-2) ===> 17
(Lisa,grade-2) ===> 61
(Mathew,grade-3) ===> 45
(Andrew,grade-2) ===> 77
(Andrew,grade-1) ===> 43
(Lisa,grade-3) ===> 86
(John,grade-1) ===> 38
(John,grade-2) ===> 74
(Mark,grade-1) ===> 84
(Andrew,grade-3) ===> 35
(Mathew,grade-2) ===> 65
result: Unit = ()

scala>
```

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

Code:

```
scala> val marksRDD = student_data.map(x=>{
     |
     |  val student_datas= x.split(",")
     |
     |  val subject = student_datas(1)
     |
     |  val marks = student_datas(3).toInt
     |
     |  (subject,marks)})
marksRDD: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[42] at map at <console>:29
```

```
scala> val countRDD = marksRDD.map(x => ((x._1),1)).reduceByKey(_+_).sortByKey()
countRDD: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[53] at sortByKey at <console>:31

scala>

scala> val marksSubRDD = marksRDD.reduceByKey(_+_).sortByKey()
marksSubRDD: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[55] at sortByKey at <console>:31

scala>

scala> val finJoin = marksSubRDD.join(countRDD)
finJoin: org.apache.spark.rdd.RDD[(String, (Int, Int))] = MapPartitionsRDD[58] at join at <console>:35

scala>
```

Output:

```
scala> val result = finJoin.map(x => ( ( x._1.toString)+" ===> "+(x._2._1.toInt)/(x._2._2.toInt))).foreach(println)
maths ===> 46
history ===> 69
science ===> 38
result: Unit = ()

scala>
```

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

Code:

```
scala> val transRDD =  student_data.map(x=>{
 val student_datas = x.split(",")
 val grade = student_datas(2)
 val subject = student_datas(1)
 val marks = student_datas(3).toInt
 ((grade,subject),marks)})
transRDD: org.apache.spark.rdd.RDD[((String, String), Int)] = MapPartitionsRDD[60] at map at <console>:29

scala> val countRDD = transRDD.map(x => (x._1,1)).reduceByKey(_+_).sortByKey()
countRDD: org.apache.spark.rdd.RDD[((String, String), Int)] = ShuffledRDD[63] at sortByKey at <console>:31

scala>

scala> val totMarksRDD = transRDD.reduceByKey(_+_).sortByKey()
totMarksRDD: org.apache.spark.rdd.RDD[((String, String), Int)] = ShuffledRDD[65] at sortByKey at <console>:31

scala>

scala> val joinMarks = totMarksRDD.join(countRDD)
joinMarks: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD[68] at join at <console>:35

scala>
```

```
scala> val result = joinMarks.map(x => ( ( x._1.toString)+" ===> "+(x._2._1.toInt)/(x._2._2.toInt)))
result: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[69] at map at <console>:37
```

Output:

```
scala> result.foreach(println)
(grade-1,history) ===> 51
(grade-2,history) ===> 79
(grade-3,history) ===> 86
(grade-3,science) ===> 38
(grade-1,maths) ===> 46
(grade-1,science) ===> 50
(grade-2,science) ===> 30
(grade-2,maths) ===> 48

scala>
```

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

Code:

```
scala> val studentRDD = student_data.map(x=>{
 val student_datas = x.split(",")
 val grade = student_datas(2)
 val marks = student_datas(3).toInt
 val name = student_datas(0)
 ((name,grade),marks)})
studentRDD: org.apache.spark.rdd.RDD[((String, String), Int)] = MapPartitionsRDD[70] at map at <console>:29
```

```
scala> val filterRDD = studentRDD.filter(x => (x._1._2 =="grade-2"))
filterRDD: org.apache.spark.rdd.RDD[((String, String), Int)] = MapPartitionsRDD[82] at filter at <console>:31

scala>

scala> val countRDD = filterRDD.map(x => (x._1,1)).reduceByKey(_+_)sortByKey()
countRDD: org.apache.spark.rdd.RDD[((String, String), Int)] = ShuffledRDD[85] at sortByKey at <console>:33

scala>

scala> val totMarks = filterRDD.reduceByKey(_+_).sortByKey()
totMarks: org.apache.spark.rdd.RDD[((String, String), Int)] = ShuffledRDD[87] at sortByKey at <console>:33

scala>

scala> val finalJoin = totMarks.join(countRDD)
finalJoin: org.apache.spark.rdd.RDD[((String, String), (Int, Int))] = MapPartitionsRDD[90] at join at <console>:37

scala>

scala> val avg = finalJoin.map(x => ((x._2._1.toInt)/(x._2._2.toInt)))
avg: org.apache.spark.rdd.RDD[Int] = MapPartitionsRDD[91] at map at <console>:39

scala>

scala> val finalFiltercount = avg.filter(x => x > 50).count
finalFiltercount: Long = 4

scala>
```

Output:

```
scala> val finalFiltercount = avg.filter(x => x > 50).count
finalFiltercount: Long = 4
```

Problem Statement 3:

- Are there any students in the college that satisfy the below criteria:
  Average score per student_name across all grades is same as average score per student_name per grade
  (Hint - Use Intersection Property.)

Code:

```
scala> val tuplerdd = student_data.map(x => {
val row = x.split(",").toList
(row.apply(0), row.apply(1), row.apply(2), row.apply(3).toInt, row.apply(4).toInt)
})
tuplerdd: org.apache.spark.rdd.RDD[(String, String, String, Int, Int)] = MapPartitionsRDD[93] at map at <console>:29

scala> val in1 = tuplerdd.map(x=> (x._1,(x._4+x._5))).groupByKey.map(x=>(x._1,(x._2.sum.toDouble/(x._2.size*2))))
in1: org.apache.spark.rdd.RDD[(String, Double)] = MapPartitionsRDD[96] at map at <console>:31
```

```
scala> val in2 = tuplerdd.map(x=> ((x._1,x._3),(x._4 + x._5))).groupByKey.map(x => (x._1._1,(x._2.sum.toDouble/(x._2.size*2))
))
in2: org.apache.spark.rdd.RDD[(String, Double)] = MapPartitionsRDD[99] at map at <console>:31

scala>
```

Output:

```
scala> in1.foreach(println)
(Mark,31.75)
(Andrew,29.833333333333332)
(Mathew,36.375)
(John,29.875)
(Lisa,35.625)

scala> in2.foreach(println)
(Lisa,18.0)
(Mark,15.0)
(Lisa,37.5)
(Mathew,28.5)
(Andrew,44.0)
(Andrew,28.666666666666668)
(Lisa,49.5)
(John,25.333333333333332)
(John,43.5)
(Mark,48.5)
(Andrew,24.5)
(Mathew,39.0)

scala>
```

```
scala> val in3 = in1.intersection(in2).collect
in3: Array[(String, Double)] = Array()

scala>
```