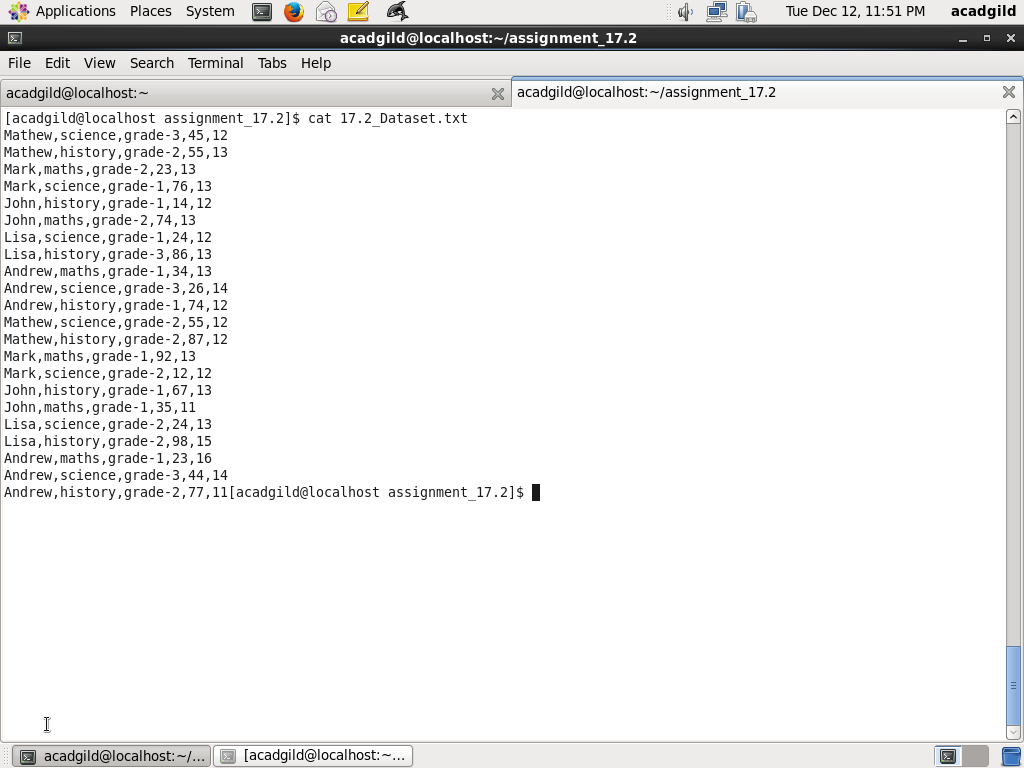
Copy the dataset to /home/acadgild/assignment\_17.2/17.2\_Dataset.txt. Contents of Dataset is as below:



Task 1.1: Read the file and create a tuple RDD

First create a RDD named baseRDD from the text file /home/acadgild/assignment\_17.2/17.2\_Dataset.txt.

Next, craate a tuple RDD using the map function on baseRDD and splitting records by separator ,. First four fields of each tuple are name, subject,grade, marks

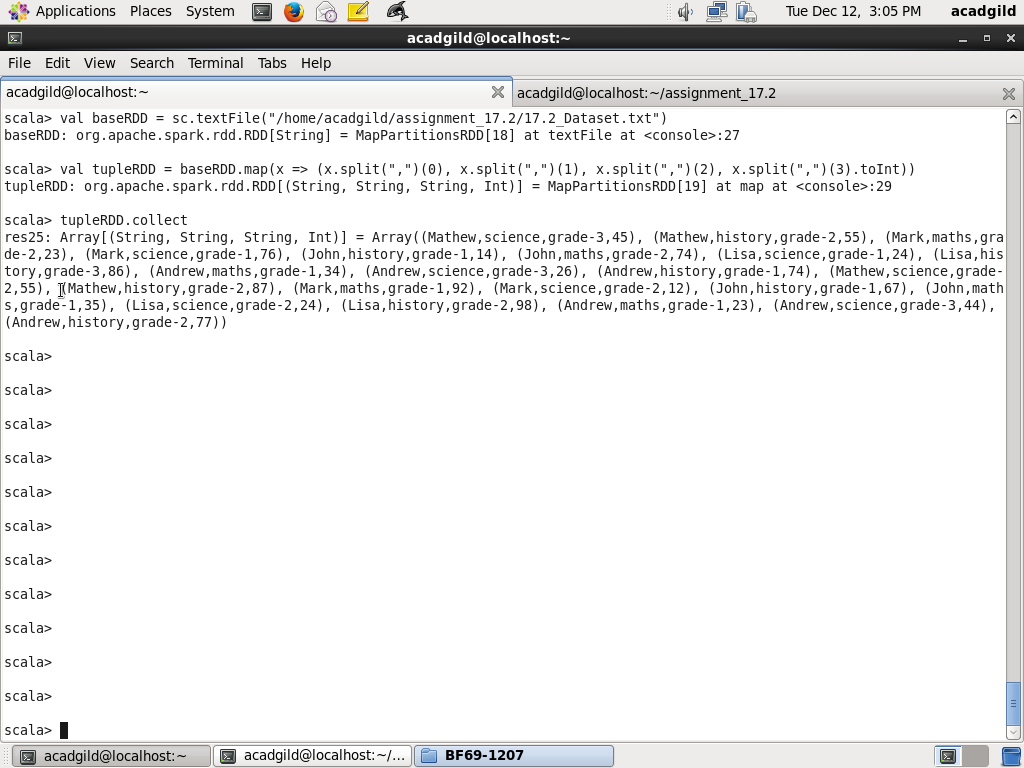
Code is as below:

val baseRDD = sc.textFile("/home/acadgild/assignment\_17.2/17.2\_Dataset.txt")

val tupleRDD = baseRDD.map(x => (x.split(",")(0), x.split(",")(1), x.split(",")(2), x.split(",")(3).toInt))

tupleRDD.collect

Screenshot is as below:

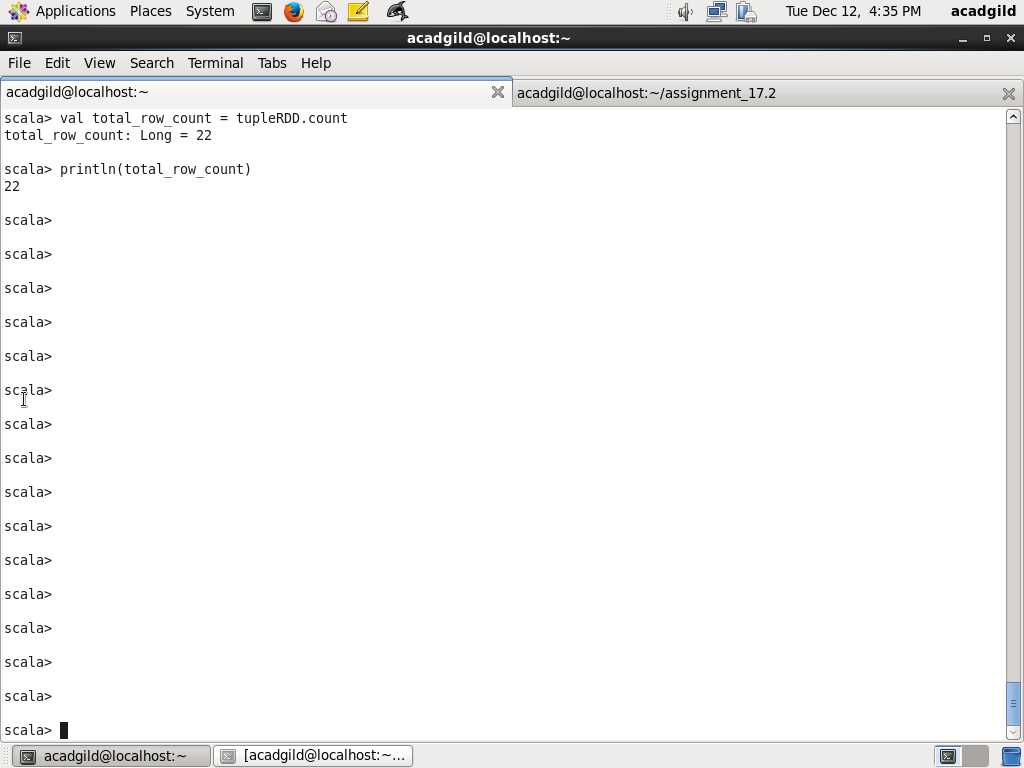


Task 1.2: Find count of total number of rows

Using count method on tupleRDD, count number of rows and assign to total\_row\_count

val total\_row\_count = tupleRDD.count

println(total\_row\_count)



Task 1.3: Count no of distinct subjects taught

Using map method on tupleRDD, take only subject field and call distinct method to get distinct subjects and assign to distinctSubjectRDD. Using count method on distinctSubjectRDD, get count of distinct subjects

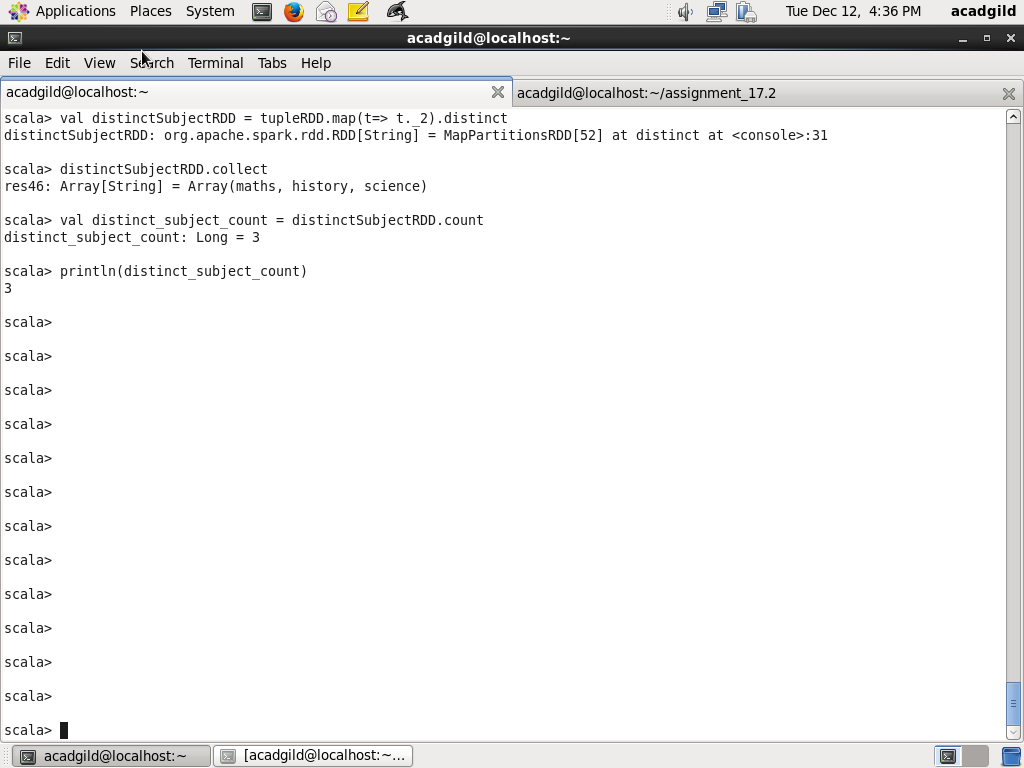
val distinctSubjectRDD = tupleRDD.map(t=> t.\_2).distinct

distinctSubjectRDD.collect

val distinct\_subject\_count = distinctSubjectRDD.count

println(distinct\_subject\_count)

Screenshot is as below:



Task 1.4 Count number of students in school whole name is Mathew and marks is 55

Use filter method on tupleRDD and check for condition if student name is Mathew and marks is 55, assign to studentRDD. Using count method calculate student count and assign to student\_count. Using println function print student\_count

Code is as below:

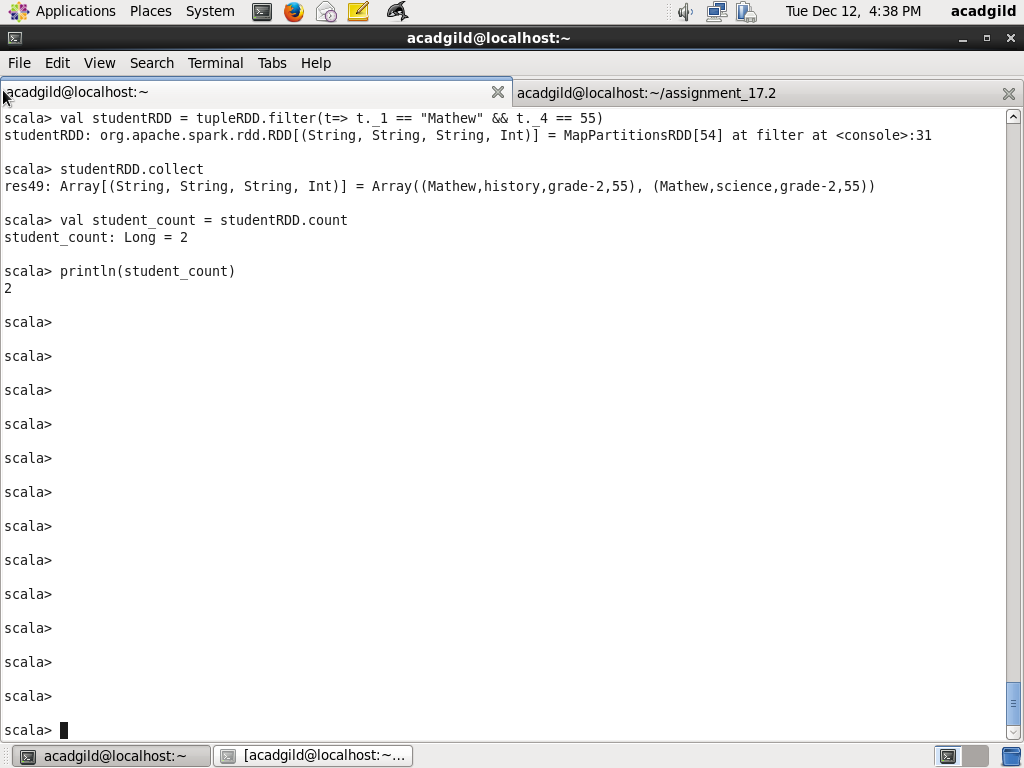
val studentRDD = tupleRDD.filter(t=> t.\_1 == "Mathew" && t.\_4 == 55)

studentRDD.collect

val student\_count = studentRDD.count

println(student\_count)

Screenshot is as below:



Task 2.1: What is the count of students per grade in the school

Using map method on tupleRDD, create a RDD with tuples having fields student grade, name and assign to gradeStudentRDD. Use distinct method to get all the distinct tuples, assign to distinctGradeStudentRDD. On distinctGradeStudentRDD use map method to form tuples of grade and 1. Next on = distinctGradeStudentMapCountRDD use method reduceByKey to get count of students per grade. Sort gradeStudentCountRDD by key and assign to sortedGradeStudentCountRDD. Next print each tuple in RDD

val gradeStudentRDD = tupleRDD.map(t => (t.\_3, t.\_1))

val distinctGradeStudentRDD = gradeStudentRDD.distinct

val distinctGradeStudentMapCountRDD = distinctGradeStudentRDD.map(t => (t.\_1, 1))

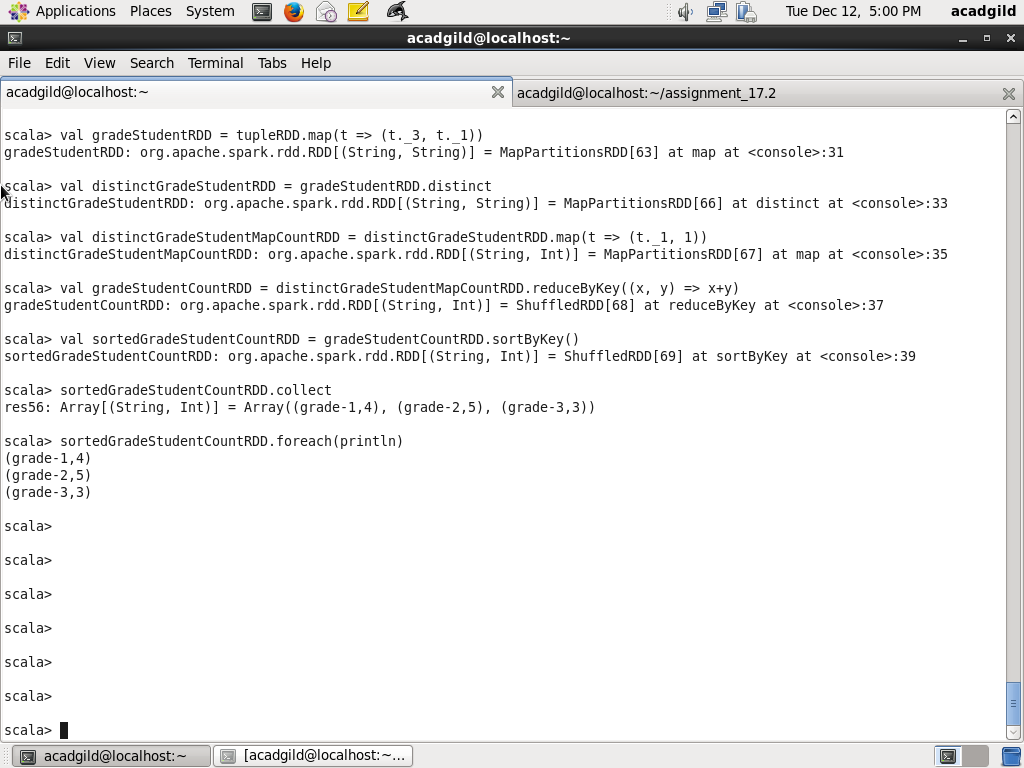
val gradeStudentCountRDD = distinctGradeStudentMapCountRDD.reduceByKey((x, y) => x+y)

val sortedGradeStudentCountRDD = gradeStudentCountRDD.sortByKey()

sortedGradeStudentCountRDD.collect

sortedGradeStudentCountRDD.foreach(println)

Screenshot is as below:



Task 2.2: Find the average of each student

First using map on tupleRDD and create tuple with keys as (name, grade) and values as (marks, 1). Key (name,grade) is required because same name say Lisa is there in multiple grades, but they are all different students. Use reduceByKey method on the RDD = studentGradeMarksRDD, to calculate total marks and total count. Next, use map method on = totalMarksCountByStudentRDD, create tuple with key being (name, grade) and average value is calculated dividing total marks by no of subjects and converting to Float. Next for each tuple print student name and average marks

Code is as below:

val studentGradeMarksRDD = tupleRDD.map(t=> ((t.\_1, t.\_3), (t.\_4, 1)))

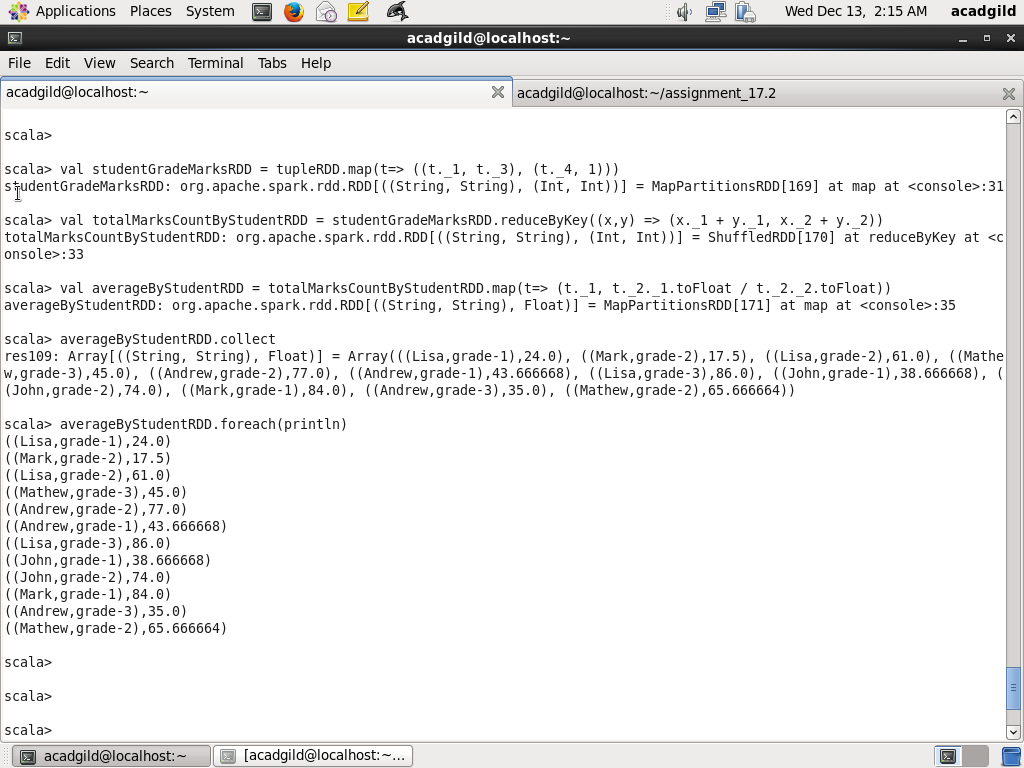
val totalMarksCountByStudentRDD = studentGradeMarksRDD.reduceByKey((x,y) => (x.\_1 + y.\_1, x.\_2 + y.\_2))

val averageByStudentRDD = totalMarksCountByStudentRDD.map(t=> (t.\_1.\_1, t.\_2.\_1.toFloat / t.\_2.\_2.toFloat))

averageByStudentRDD.collect

averageByStudentRDD.foreach(println)

Screenshot is as below:



Task 2.3 What is the average of marks of each subject across grades

Use map method on tupleRDD to create tuples whose first element is subject and second element is another tuple with values marks and 1 ( used for counting subject) and assign to studentMarksCountBySubjectRDD . Next use method reduceByKey on studentMarksCountBySubjectRDD to get total of marks and count of subjects with key being subject. Next, use map method on totalMarksCountBySubjectRDD and create tuples whose first elemen is subject, second element is average marks calculated by dividing total marks by number of subjects. Next print each of the tuples

Code is as below:

val studentMarksCountBySubjectRDD = tupleRDD.map(t=> (t.\_2, (t.\_4, 1)))

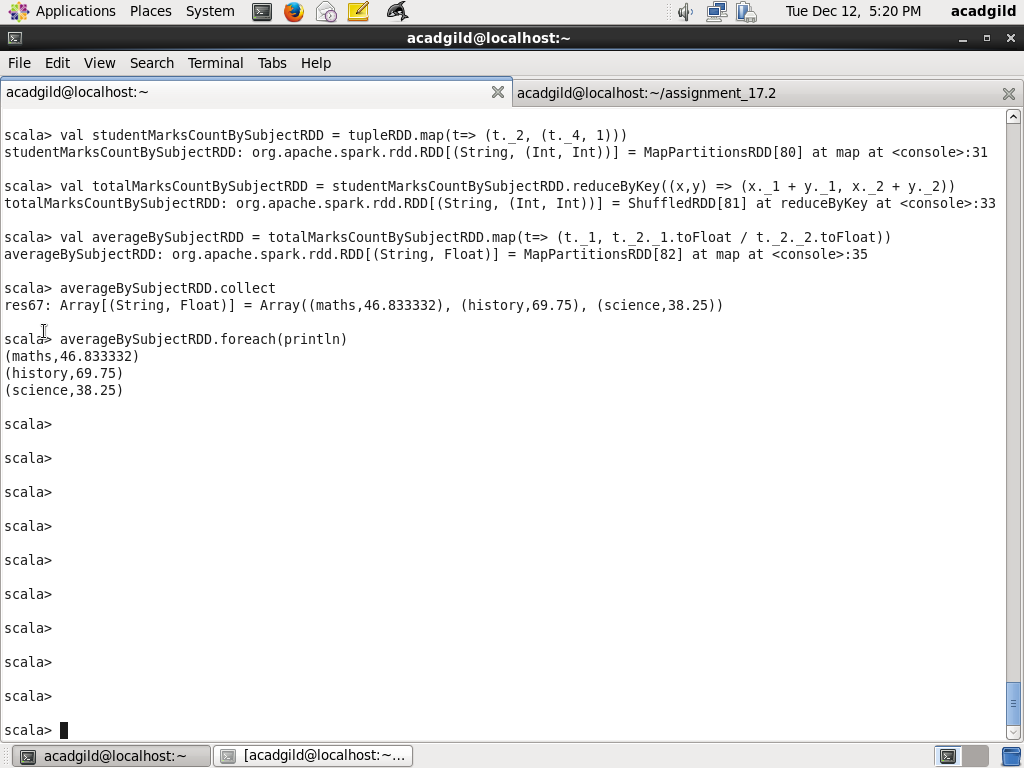
val totalMarksCountBySubjectRDD = studentMarksCountBySubjectRDD.reduceByKey((x,y) => (x.\_1 + y.\_1, x.\_2 + y.\_2))

val averageBySubjectRDD = totalMarksCountBySubjectRDD.map(t=> (t.\_1, t.\_2.\_1.toFloat / t.\_2.\_2.toFloat))

averageBySubjectRDD.collect

averageBySubjectRDD.foreach(println)

Screenshot is as below:



Task 2.4 What is the average of marks of each subject per grade

Use map method on tupleRDD to create tuples whose first element is a tuple (whose first element is subject and second element is grade) and second element is another tuple with values marks and 1 ( used for counting subject) and assign to studentMarksCountBySubjectAndGradeRDD . Next use method reduceByKey studentMarksCountBySubjectAndGradeRDD to get total of marks and count of subjects with key being subject and grade. Next, use map method on totalMarksCountBySubjectRDD and create tuples whose first element is( subject,grade) second element is average marks calculated by dividing total marks by number of subjects. Next print each of the tuples

Code is as below:

val studentMarksCountBySubjectAndGradeRDD = tupleRDD.map(t=> ((t.\_2, t.\_3), (t.\_4, 1)))

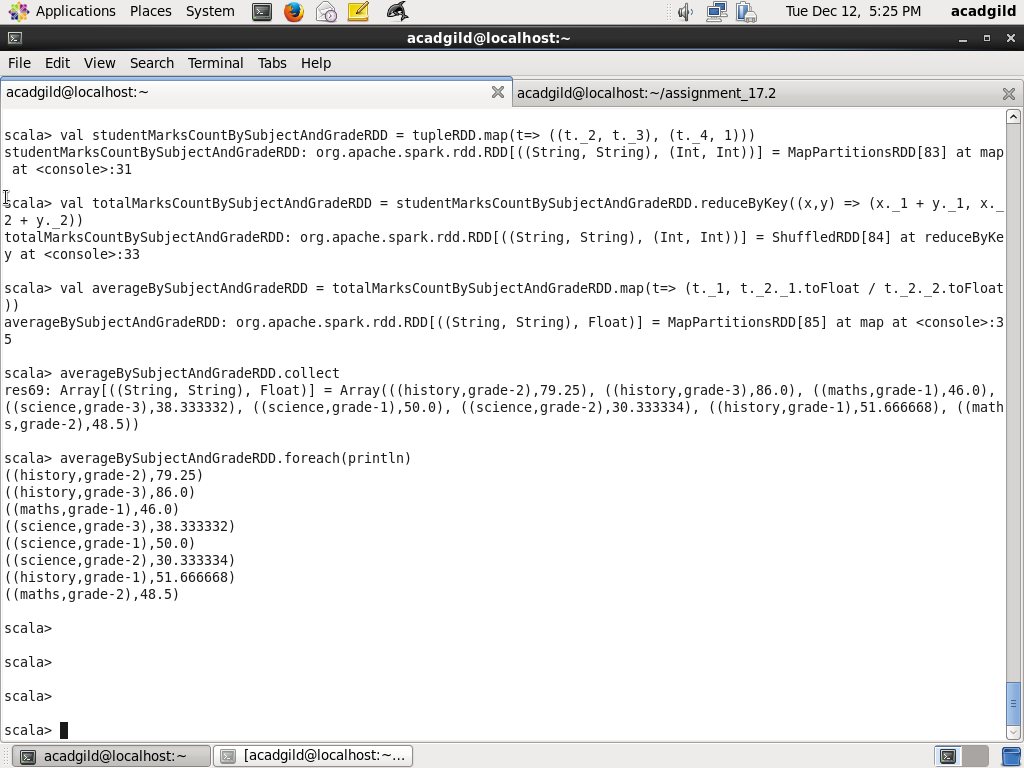
val totalMarksCountBySubjectAndGradeRDD = studentMarksCountBySubjectAndGradeRDD.reduceByKey((x,y) => (x.\_1 + y.\_1, x.\_2 + y.\_2))

val averageBySubjectAndGradeRDD = totalMarksCountBySubjectAndGradeRDD.map(t=> (t.\_1, t.\_2.\_1.toFloat / t.\_2.\_2.toFloat))

averageBySubjectAndGradeRDD.collect

averageBySubjectAndGradeRDD.foreach(println)

Screenshot is as below:



Task 2.5 For all students in grade-2 how many have average more than 50

Use filter method on tupleRDD to filter students who are in grade-2 and assign to grade2StudentRDD. Next use map method on grade2StudentRDD to find tuples whose first element is student name and second element is a tuple whose first element is marks and second element is 1 ( used for counting subjects) and assign to grade2StudentMarksCountByNameRDD. Next, use reduceByKey to calculate a tuple whose first element is total marks and second element is number of subjects, Key is name. Next, use map method on totalMarksCountBySubjectRDD and create tuples whose first element is name second element is average marks calculated by dividing total marks by number of subjects and assign to

averageGrade2StudentMarksCountByNameRDD. Next, filter averageGrade2StudentMarksCountByNameRDD for which average marks is greater than 50 and print the count of tuples

Code is as below:

val grade2StudentRDD = tupleRDD.filter(t=> t.\_3 == "grade-2")

val grade2StudentMarksCountByNameRDD = grade2StudentRDD.map(t=> (t.\_1, (t.\_4, 1)))

val totalGrade2StudentMarksCountByNameRDD = grade2StudentMarksCountByNameRDD.reduceByKey((x,y) => (x.\_1 + y.\_1, x.\_2 + y.\_2))

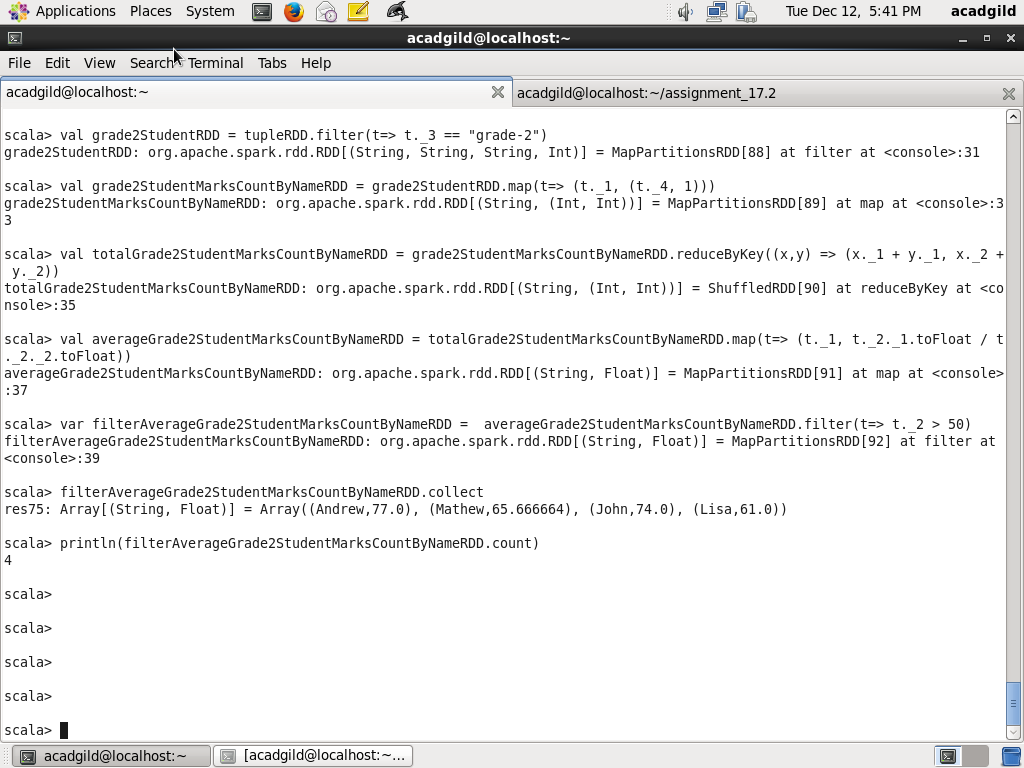
val averageGrade2StudentMarksCountByNameRDD = totalGrade2StudentMarksCountByNameRDD.map(t=> (t.\_1, t.\_2.\_1.toFloat / t.\_2.\_2.toFloat))

var filterAverageGrade2StudentMarksCountByNameRDD = averageGrade2StudentMarksCountByNameRDD.filter(t=> t.\_2 > 50)

filterAverageGrade2StudentMarksCountByNameRDD.collect

println(filterAverageGrade2StudentMarksCountByNameRDD.count)

Screenshot is as below:



Task 3.1 Find if there is any student name for which average of marks across grades is equal to average of marks per grade

Step1: Calculate the average of marks of each student name across grades

Use the code below to calculate the average marks per student across grades. In this final result, map is used to create a key by concatenating name, ---, average marks with value being the average marks

val studentMarksCountByNameAcrossGradesRDD = tupleRDD.map(t=> (t.\_1, (t.\_4, 1)))

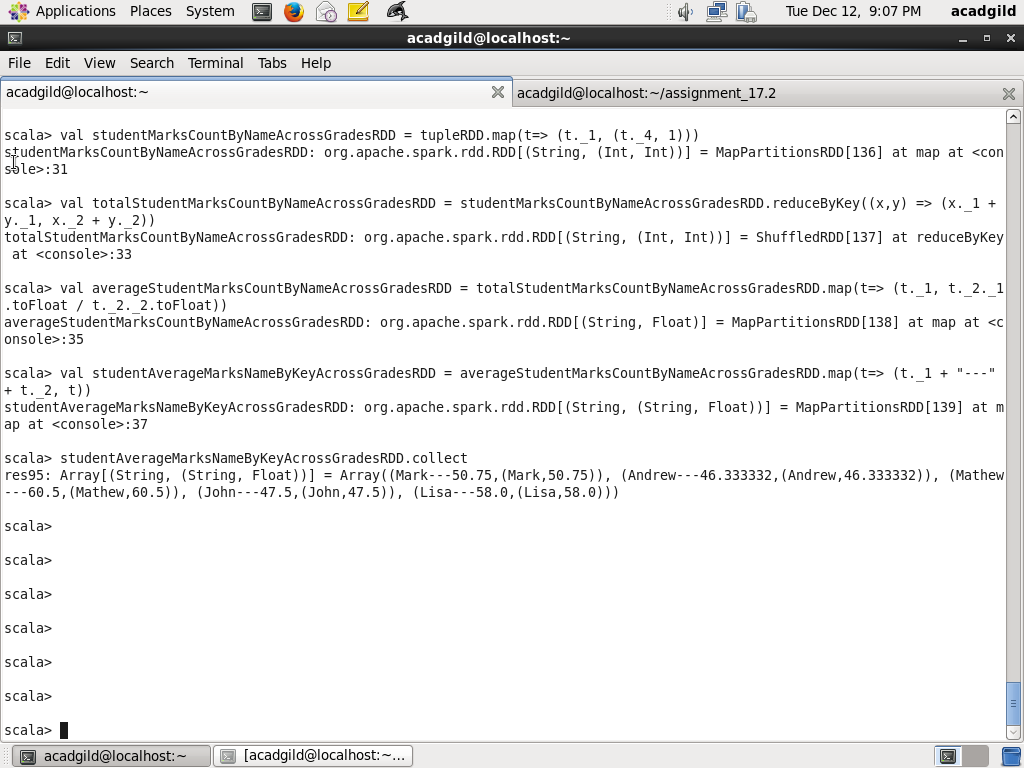
val totalStudentMarksCountByNameAcrossGradesRDD = studentMarksCountByNameAcrossGradesRDD.reduceByKey((x,y) => (x.\_1 + y.\_1, x.\_2 + y.\_2))

val averageStudentMarksCountByNameAcrossGradesRDD = totalStudentMarksCountByNameAcrossGradesRDD.map(t=> (t.\_1, t.\_2.\_1.toFloat / t.\_2.\_2.toFloat))

val studentAverageMarksNameByKeyAcrossGradesRDD = averageStudentMarksCountByNameAcrossGradesRDD.map(t=> (t.\_1 + "---" + t.\_2, t))

studentAverageMarksNameByKeyAcrossGradesRDD.collect

Screenshot is as below:



Step2: Calculate average marks each student name per grade

Use the code below to calculate the average marks per student across grades. In this final result, map is used to create a key by concatenating name, ---, average marks with value being the average marks

val studentMarksCountByNameAndGradeRDD = tupleRDD.map(t=> ((t.\_1, t.\_3), (t.\_4, 1)))

val totalStudentMarksCountByNameAndGradesRDD = studentMarksCountByNameAndGradeRDD.reduceByKey((x,y) => (x.\_1 + y.\_1, x.\_2 + y.\_2))

val averageStudentMarksCountByNameAndGradeRDD = totalStudentMarksCountByNameAndGradesRDD.map(t=> (t.\_1.\_1, t.\_2.\_1.toFloat / t.\_2.\_2.toFloat))

val studentAverageMarksNameByKeyForAGradeRDD = averageStudentMarksCountByNameAndGradeRDD.map(t=> (t.\_1 + "---" + t.\_2, t))

studentAverageMarksNameByKeyForAGradeRDD.collect

Step3: Use set intersection to find common student obtained from step1 and step2

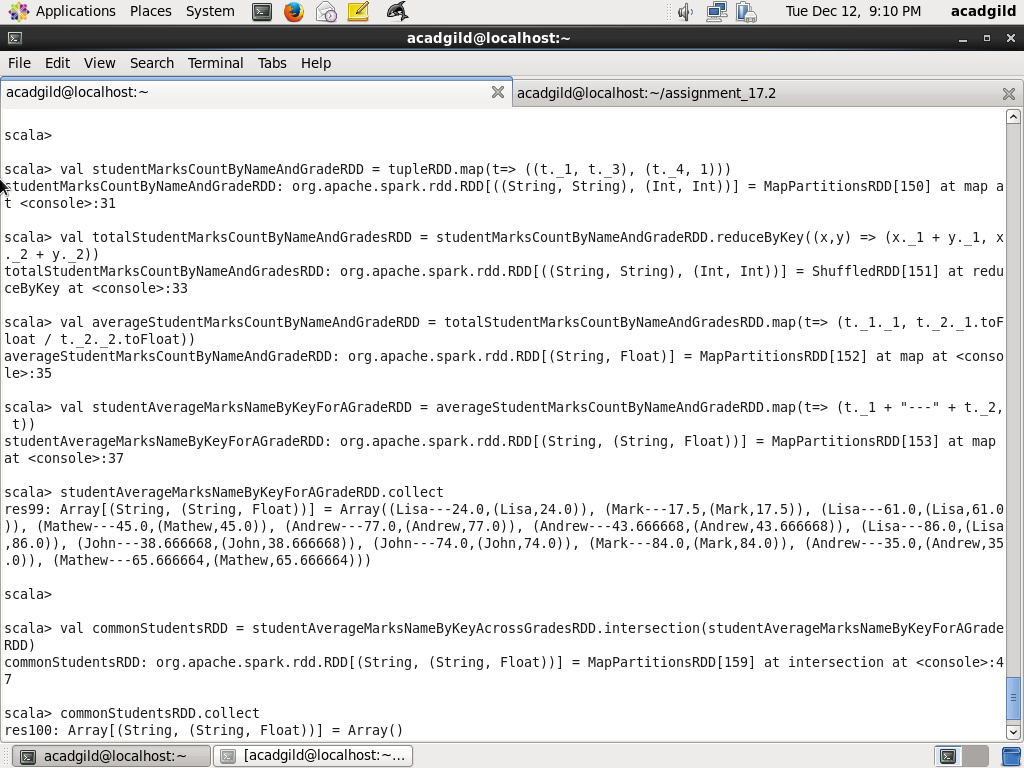
Her use set intersection on studentAverageMarksNameByKeyAcrossGradesRDD, studentAverageMarksNameByKeyForAGradeRDD. Print the common students

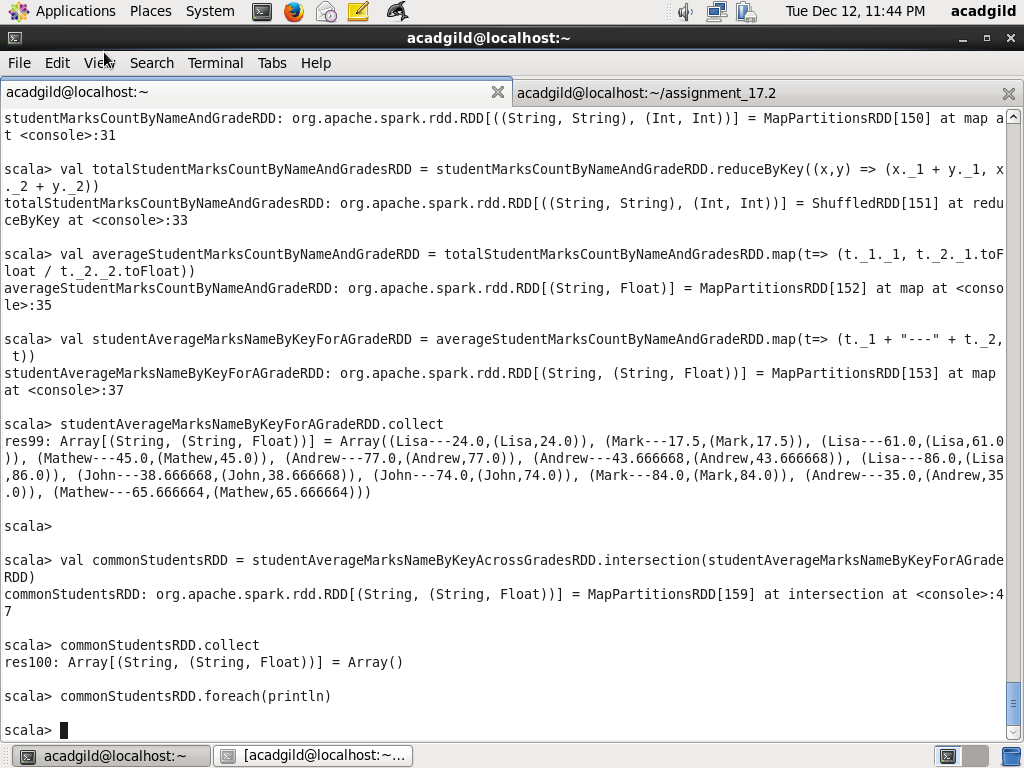
val commonStudentsRDD = studentAverageMarksNameByKeyAcrossGradesRDD.intersection(studentAverageMarksNameByKeyForAGradeRDD)

commonStudentsRDD.collect

commonStudentsRDD.foreach(println)

Screenshot is as below:





Note that there is no student whose average score across grades is same as average score per grade

Mathew(Across Grades) - (45 + 55 + 55 + 87)/4 = 60.5

Mathew (grade\_2) = 65.66

Methew (grade\_3) = 45

Mark (Across Grades) - (23+76+ 92 +12) / 4 = 50.75

Mark (grade\_1) = (76+92)/2 = 84

Mark (grade\_2) = (23+12)/2 = 17.50

John (Across Grades) - (14+74+67+35) / 4 = 47.59

John ( grade\_1) - (14+67+35)/3 = 38.66

John (grade\_2) - 74

Lisa (Across Grades) - (24+86+24+98)/4 = 58

Lisa( grade\_1) - 24

Lisa( grade\_2) - (24+98)/2 = 61

Lisa (grade\_3) - 86

Andrew (Across Grades) - (34+26+74 +23 +44 +77)/6 = 46.33

Andrew (grade\_1) - (34+74+23)/3 = 43.66

Andrew (grade\_2) - 77

Andrew (grade\_3) - (26+44)/2 = 35. I have manually verified

Mathew(Across Grades) - (45 + 55 + 55 + 87)/4 = 60.5

Mathew (grade\_2) = 65.66

Methew (grade\_3) = 45

Mark (Across Grades) - (23+76+ 92 +12) / 4 = 50.75

Mark (grade\_1) = (76+92)/2 = 84

Mark (grade\_2) = (23+12)/2 = 17.50

John (Across Grades) - (14+74+67+35) / 4 = 47.59

John ( grade\_1) - (14+67+35)/3 = 38.66

John (grade\_2) - 74

Lisa (Across Grades) - (24+86+24+98)/4 = 58

Lisa( grade\_1) - 24

Lisa( grade\_2) - (24+98)/2 = 61

Lisa (grade\_3) - 86

Andrew (Across Grades) - (34+26+74 +23 +44 +77)/6 = 46.33

Andrew (grade\_1) - (34+74+23)/3 = 43.66

Andrew (grade\_2) - 77

Andrew (grade\_3) - (26+44)/2 = 35