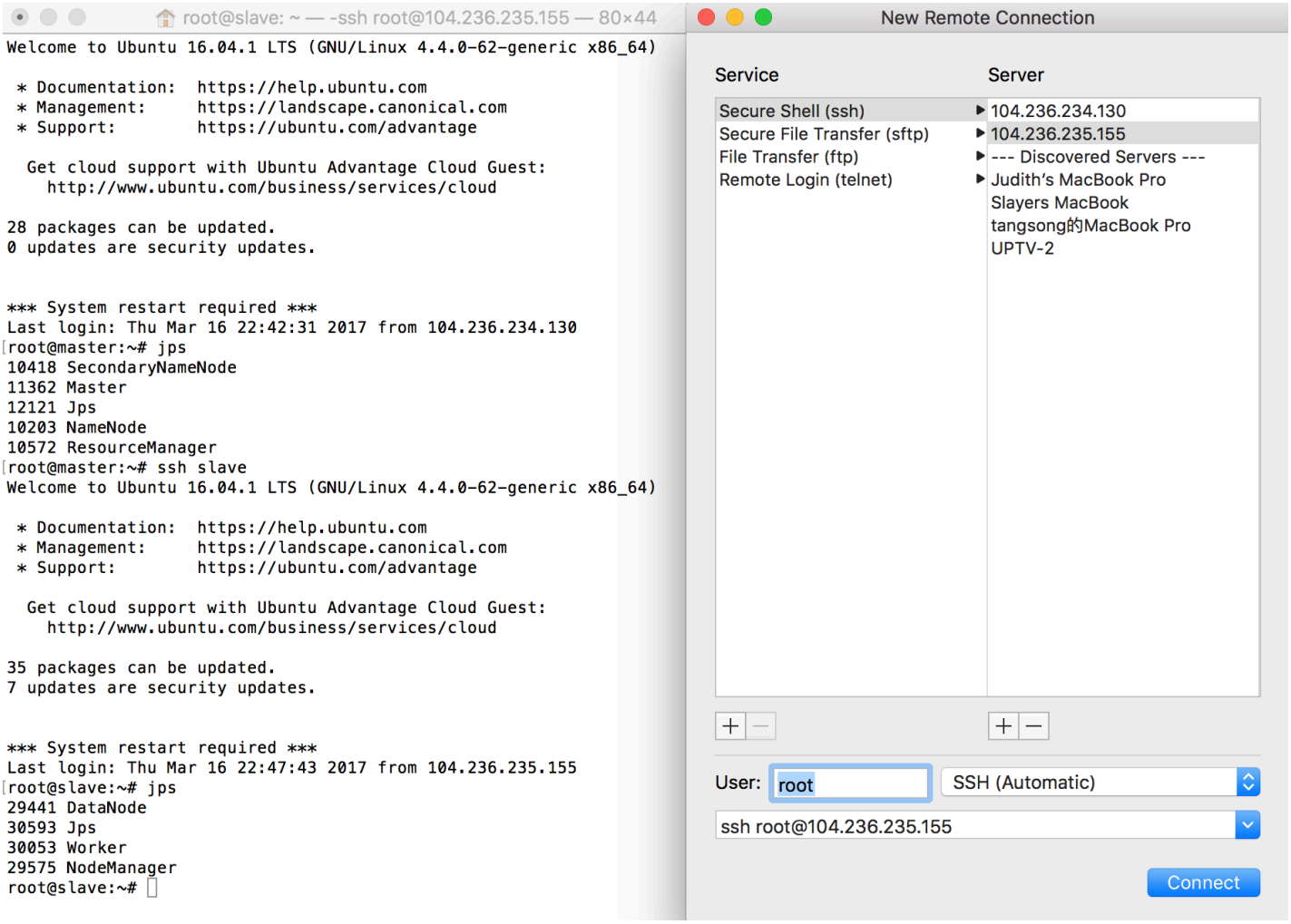
# INFSCI 2750: Cloud Computing
# Mini Project 2
# README

1. Setting up Spark on YARN in remote Ubuntu server
   Following picture is the screenshot of working system

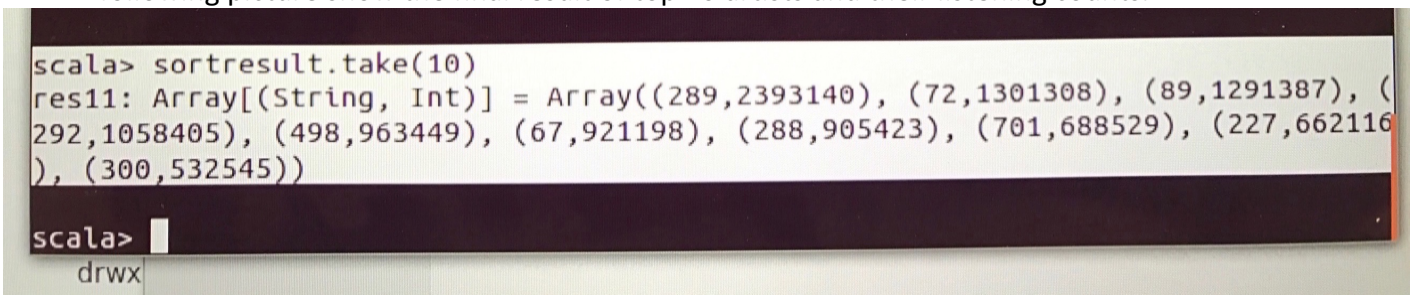



in Master server, there is Master.
In Slave server, there is a Worker

2. Develop spark program
   (1) Using cached RDD operation to analyze user_artists.dat in standalone Spark
   place the data file in the local working directory for example like /opt/spark/spark-2.0.2-bin-hadoop2.7

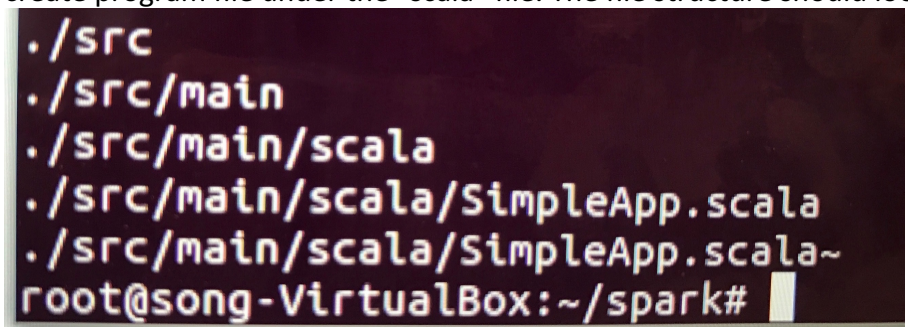
```
#start the spark by spark-shell

#input the data file in the spark RDD by sc
val data=sc.textFile("user_artists.dat").cache()
data.count
it should show 92834

#map the result
val mapresult=data.map(line=>line.split("\\s+"))
mapresult.collect

#map the key-value pair
val mapkey=mapresult.map(arr => (arr(1),arr(2).toInt)).cache()
mackey.collect

#reduce the result by key
val reduceresult=mapkey.reduceByKey(new org.apache.spark.HashPartitioner(2), (x,y)
=> x+y)
reduceresult.collect
#or
val reduceresult=mapkey.reduceByKey(_+_)
#sort the value descending
val sortresult=reduceresult.sortBy(_._2, false)
val final=sortresult.take(10)
```

   the top 10 artists ID are 289, 72, 89, 292, 498, 67, 288, 701, 227, 300
   following picture show the final result of top 10 artists and their listening counts.



(2)how to execute the .scala program.(sbt package tool)

a. make sure that spark and scala are both implemented successfully in root user
b. download sbt(simple buildtool) which is used for package the scala file and produce the .jar file. Modify the ~/.bashrc by adding path variable and install the sbt. Checking the whether sbt is working, by sbt sbt-version.
c. Create a scala file directory, for example “mkdir –p ./spark/src/main/scala” and create program file under the “scala” file. The file structure should look like this



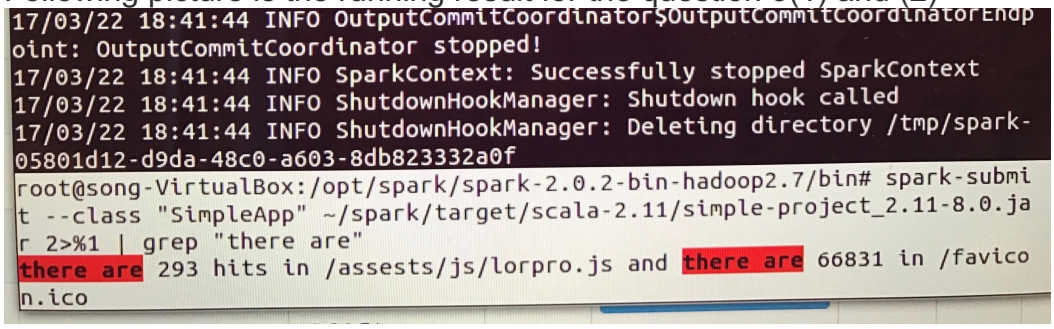

and then use “sbt package” command to package this scala program file into jar file, which is very time consuming , it will take approximate 200 second to finish the package.

And then in the spark file directory, enter “bin” file and command “spark-submit”

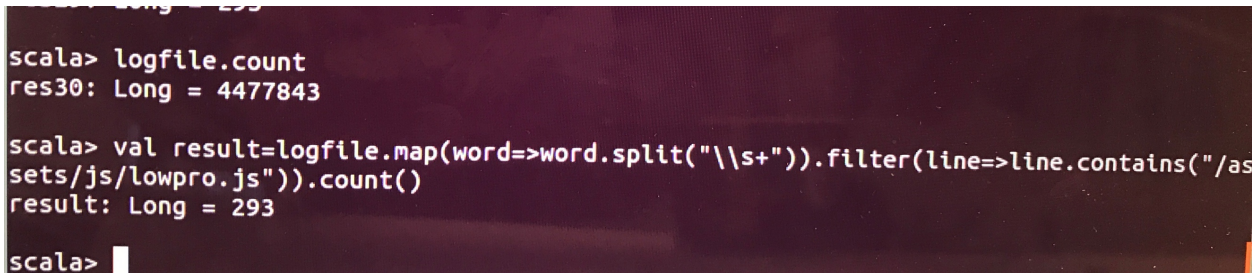
Enter the flowing command in the terminal

“spark-submit --class "SimpleApp" ~/spark/target/scala-2.11/simple-project_2.11-1.0.jar” and it will show the result or all the result will be saved to the new specified output file.

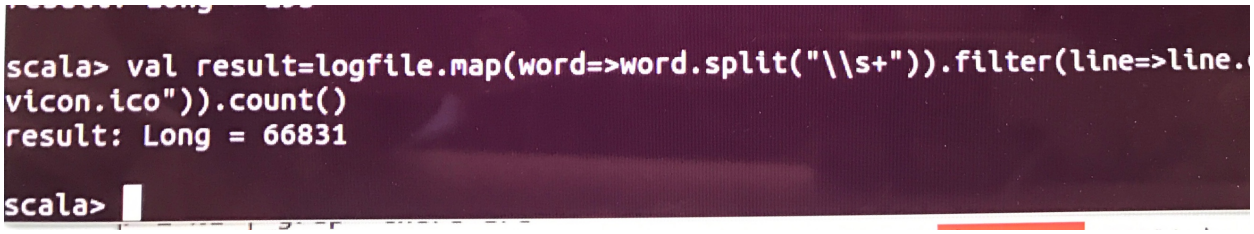
Following picture is the running result for the question 3(1) and (2)



3. Develop Spark Program
   Cached RDD method:
   (1)there are 293 hit for the /assets/js/lowpro.js
   val logfile=sc.textFile("access_log.dat").cache()
   val mapkey=logfile.map(word=>word.split("\\s+"))
   val mapfilter=mapkey.filter(line=>line.contains("/assets/js/lowpro.js")).cache()
   val result=mapfilter.count()




   (2) there are 66831 record for "/favicon.ico"
   val logfile=sc.textFile("access_log.dat").cache()
   val mapkey=logfile.map(word=>word.split("\\s+"))
   val mapfilter=mapkey.filter(line=>line.contains("/favicon.ico"))
   val result=mapfilter.count()

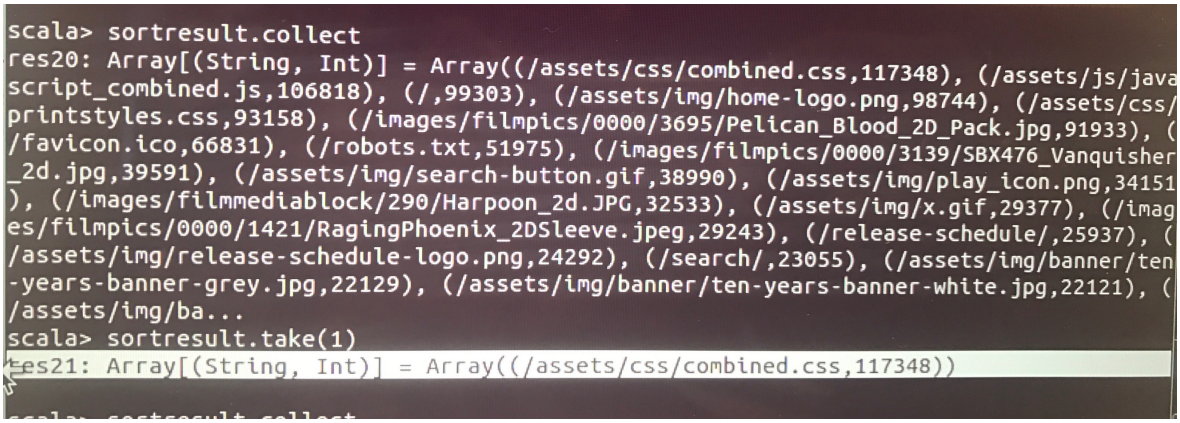


(3) the website path have the most hits number is “/assets/css/combined.css” there are 117348 hits for this site path.

#RDD operation and transition

val logfile=sc.textFile(“access_log.dat”).cache()

reduceresult=logfile.map(line=>line(“\\s+”)).cache().map(arr=>(arr(6),1)).reduceByKey(_+_).sortBy(_._2, false).take(1)

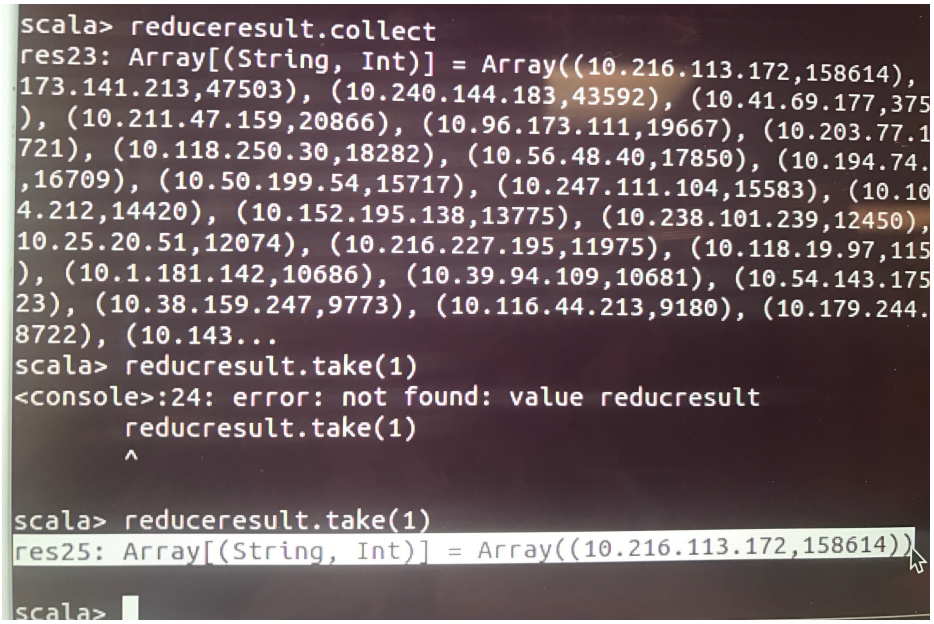



(4) the IP address have the most access website is “10.216.113.172”. there are 158614 access records for this IP address

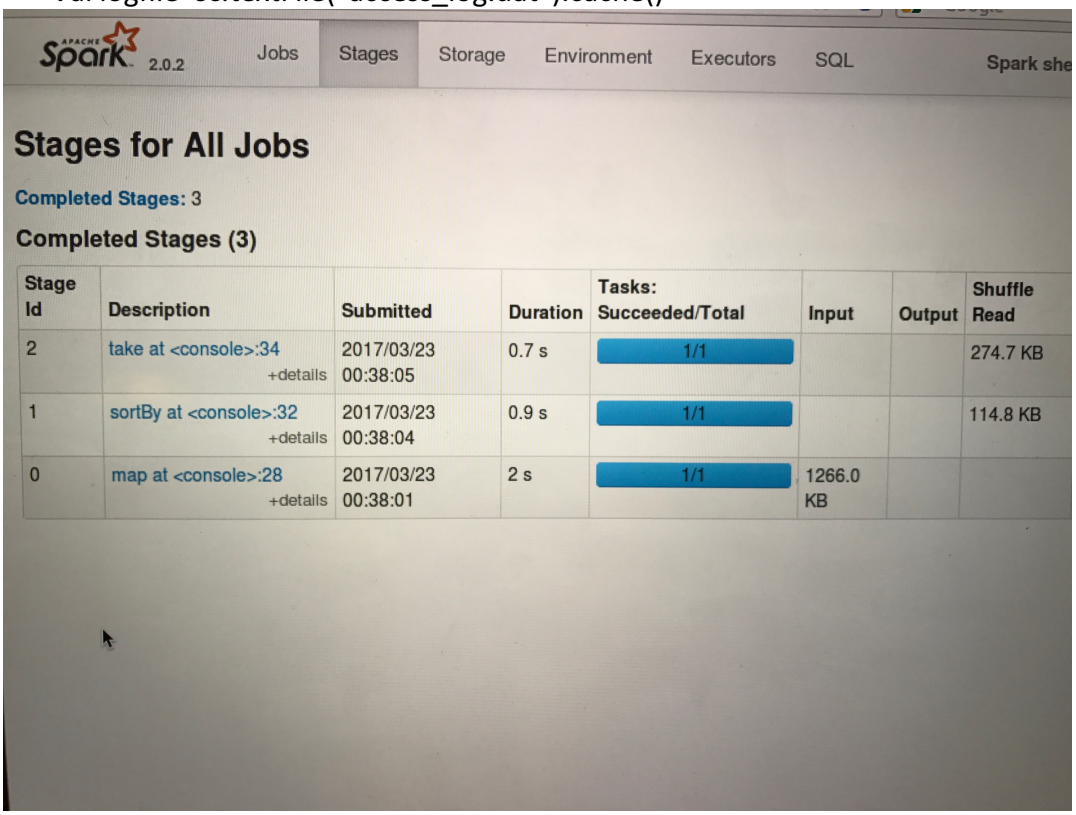
#RDD operation and transition

val logfile=sc.textFile(“access_log.dat”).cache()

val reduceresult=logfile.map(line=>line(“\\s+”)).map(arr=>(arr(0),1)).reduceByKey(_+_).sortBy(_._2, false).take(1)



4. Time Performance, running with cache RDD
   Val logfile=sc.textFile("access_log.dat").cache()



Spark 2.0.2 Jobs Stages Storage Environment Executors SQL Spark she

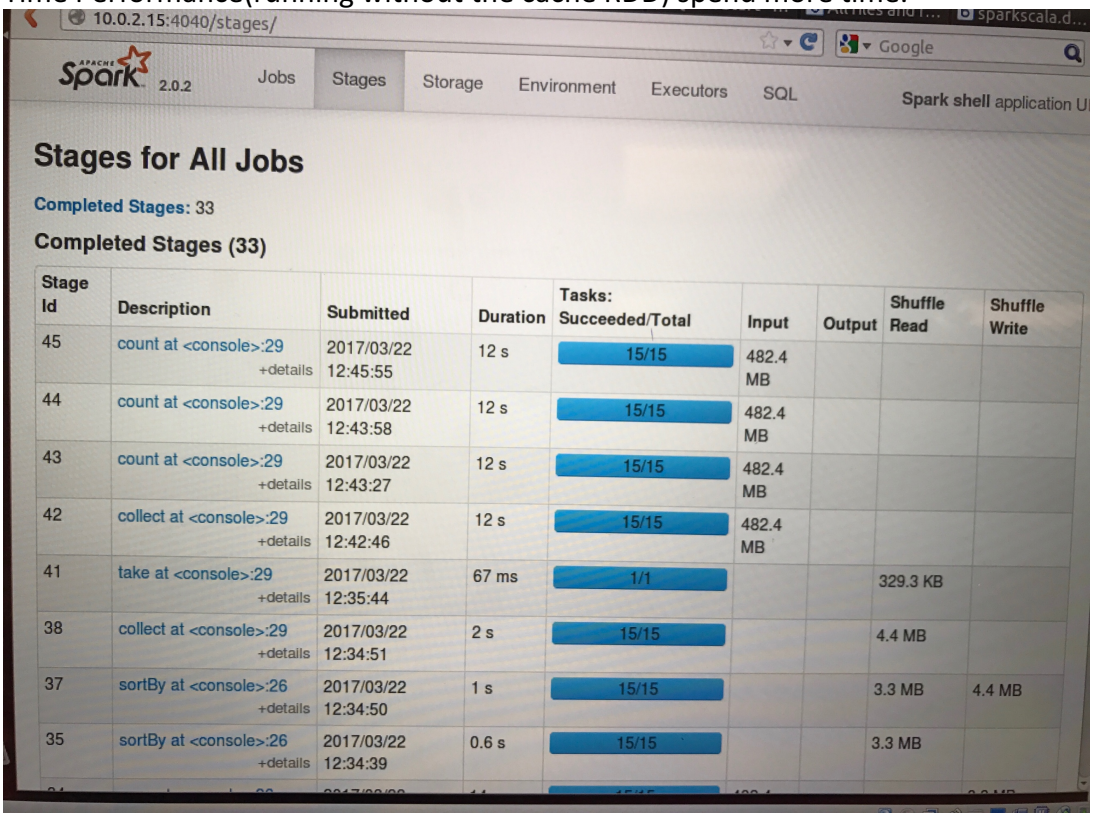
## Stages for All Jobs

**Completed Stages:** 3

### Completed Stages (3)

| Stage Id | Description | Submitted | Duration | Tasks: Succeeded/Total | Input | Output | Shuffle Read |
|---|---|---|---|---|---|---|---|
| 2 | take at <console>:34 +details | 2017/03/23 00:38:05 | 0.7 s | 1/1 | | | 274.7 KB |
| 1 | sortBy at <console>:32 +details | 2017/03/23 00:38:04 | 0.9 s | 1/1 | | | 114.8 KB |
| 0 | map at <console>:28 +details | 2017/03/23 00:38:01 | 2 s | 1/1 | 1266.0 KB | | |

Time Performance(running without the cache RDD) spend more time.



10.0.2.15:4040/stages/

Spark 2.0.2 Jobs Stages Storage Environment Executors SQL Spark shell application U

## Stages for All Jobs

**Completed Stages:** 33

### Completed Stages (33)

| Stage Id | Description | Submitted | Duration | Tasks: Succeeded/Total | Input | Output | Shuffle Read | Shuffle Write |
|---|---|---|---|---|---|---|---|---|
| 45 | count at <console>:29 +details | 2017/03/22 12:45:55 | 12 s | 15/15 | 482.4 MB | | | |
| 44 | count at <console>:29 +details | 2017/03/22 12:43:58 | 12 s | 15/15 | 482.4 MB | | | |
| 43 | count at <console>:29 +details | 2017/03/22 12:43:27 | 12 s | 15/15 | 482.4 MB | | | |
| 42 | collect at <console>:29 +details | 2017/03/22 12:42:46 | 12 s | 15/15 | 482.4 MB | | | |
| 41 | take at <console>:29 +details | 2017/03/22 12:35:44 | 67 ms | 1/1 | | | 329.3 KB | |
| 38 | collect at <console>:29 +details | 2017/03/22 12:34:51 | 2 s | 15/15 | | | 4.4 MB | |
| 37 | sortBy at <console>:26 +details | 2017/03/22 12:34:50 | 1 s | 15/15 | | | 3.3 MB | 4.4 MB |
| 35 | sortBy at <console>:26 +details | 2017/03/22 12:34:39 | 0.6 s | 15/15 | | | 3.3 MB | |