Triveni Y   Date: 3-6-2021

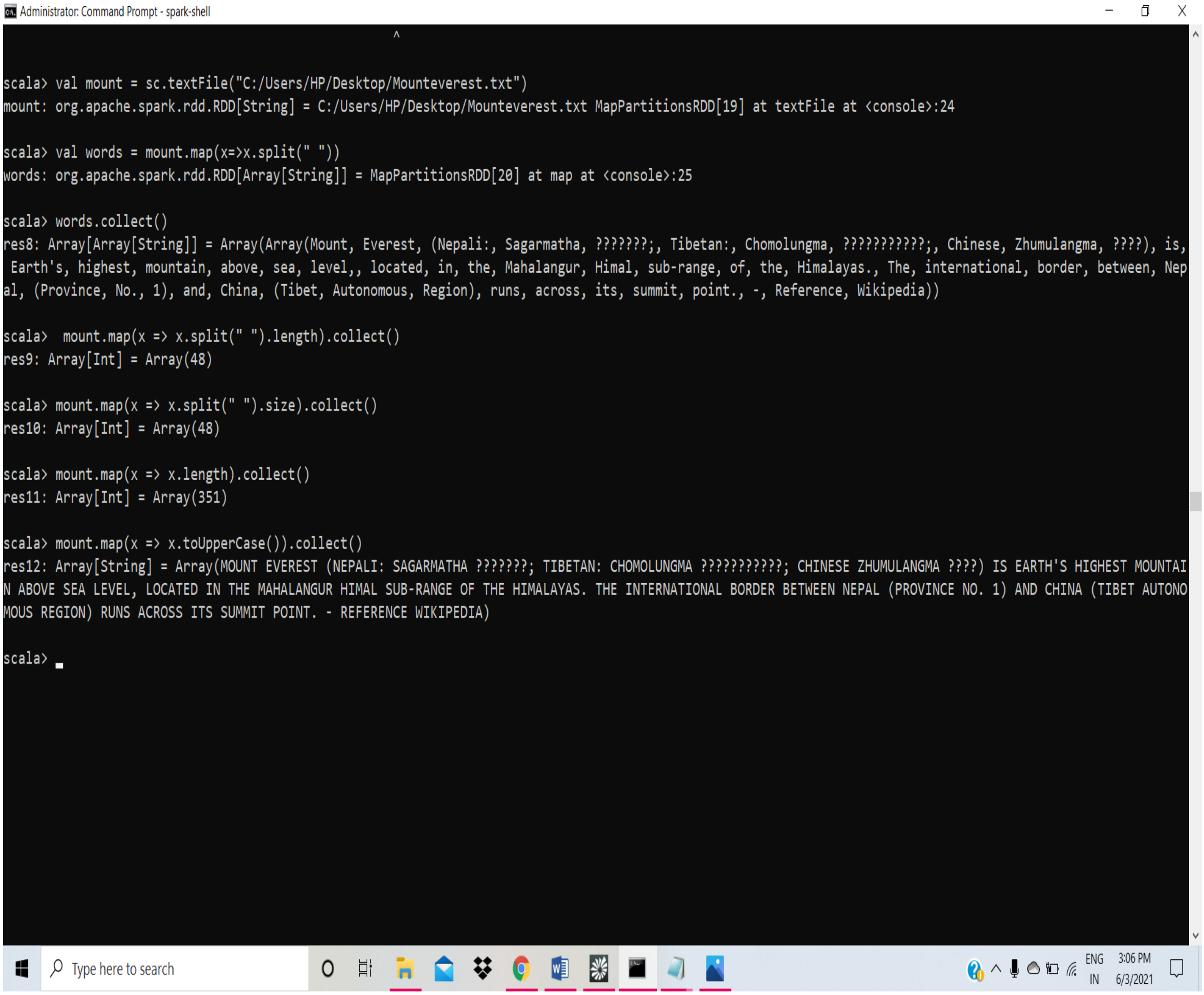
1BM19CS411

BDA LAB SCALA PROGRAMS

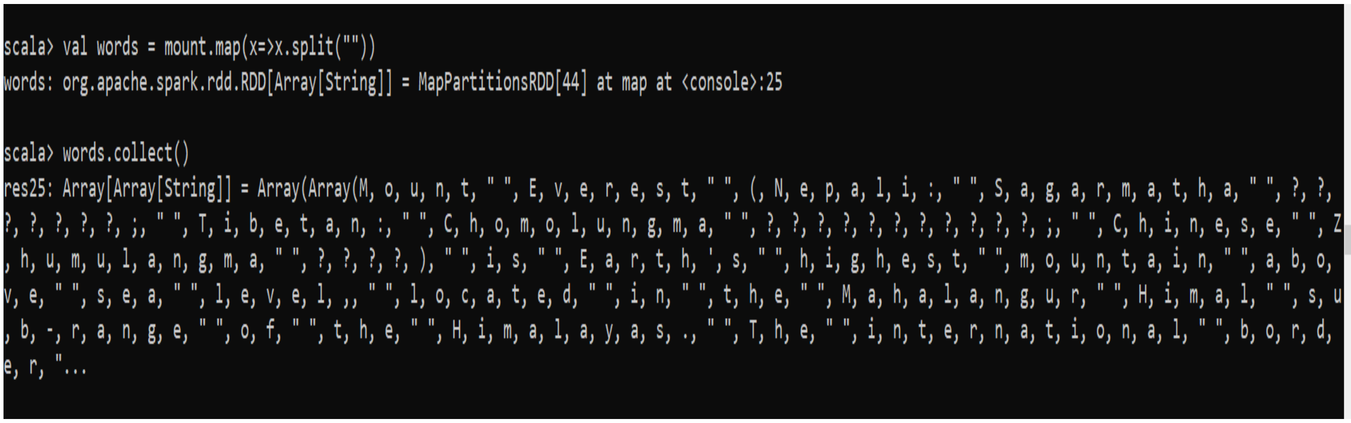
Execute any four transformations and four actions in spark shell

Transformations & Actions

Map

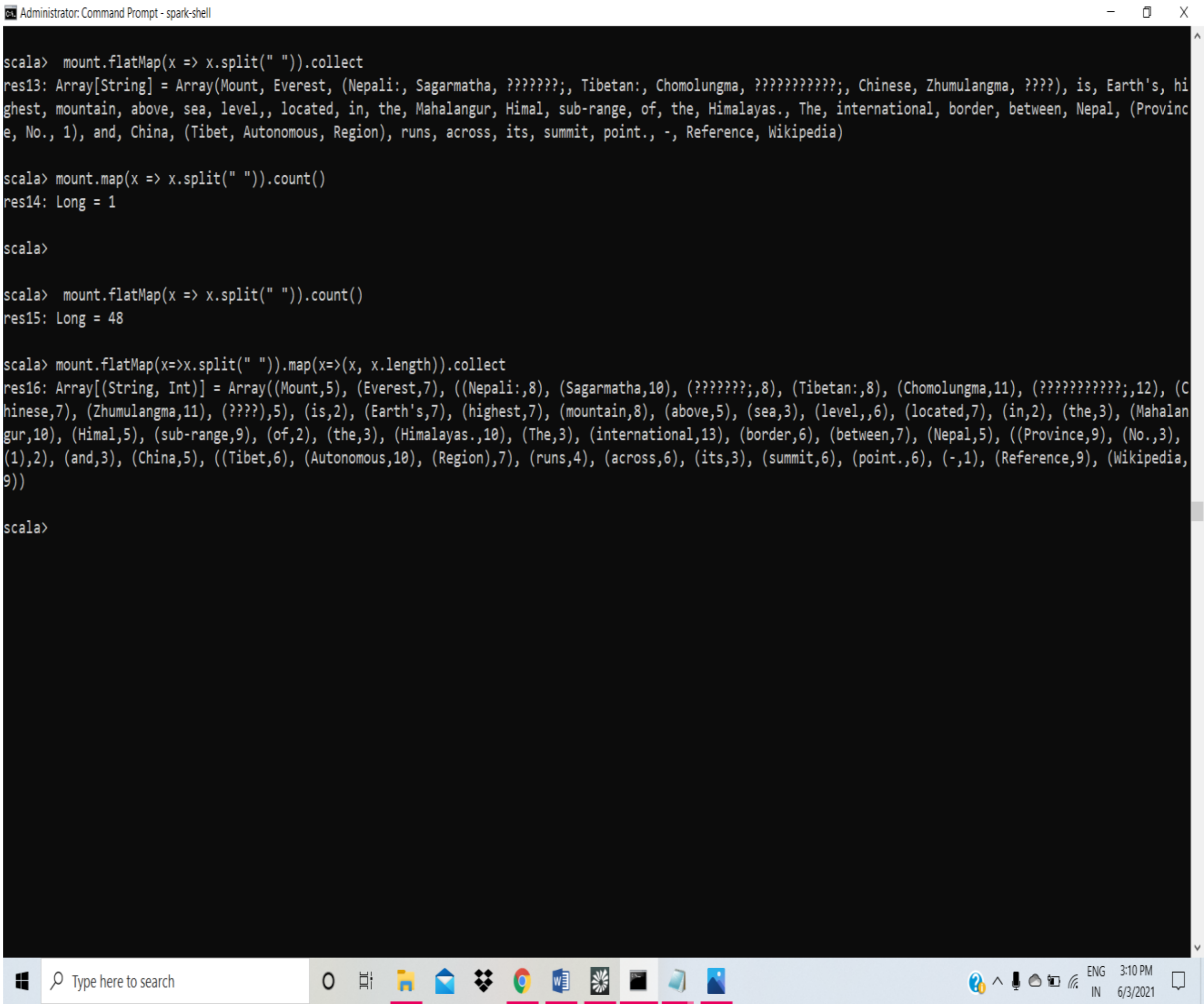


```
                                          ^

scala> val mount = sc.textFile("C:/Users/HP/Desktop/Mounteverest.txt")
mount: org.apache.spark.rdd.RDD[String] = C:/Users/HP/Desktop/Mounteverest.txt MapPartitionsRDD[19] at textFile at <console>:24

scala> val words = mount.map(x=>x.split(" "))
words: org.apache.spark.rdd.RDD[Array[String]] = MapPartitionsRDD[20] at map at <console>:25

scala> words.collect()
res8: Array[Array[String]] = Array(Array(Mount, Everest, (Nepali:, Sagarmatha, ???????;, Tibetan:, Chomolungma, ???????????;, Chinese, Zhumulangma, ????), is,
 Earth's, highest, mountain, above, sea, level,, located, in, the, Mahalangur, Himal, sub-range, of, the, Himalayas., The, international, border, between, Nep
al, (Province, No., 1), and, China, (Tibet, Autonomous, Region), runs, across, its, summit, point., -, Reference, Wikipedia))

scala>  mount.map(x => x.split(" ").length).collect()
res9: Array[Int] = Array(48)

scala> mount.map(x => x.split(" ").size).collect()
res10: Array[Int] = Array(48)

scala> mount.map(x => x.length).collect()
res11: Array[Int] = Array(351)

scala> mount.map(x => x.toUpperCase()).collect()
res12: Array[String] = Array(MOUNT EVEREST (NEPALI: SAGARMATHA ???????; TIBETAN: CHOMOLUNGMA ???????????; CHINESE ZHUMULANGMA ????) IS EARTH'S HIGHEST MOUNTAI
N ABOVE SEA LEVEL, LOCATED IN THE MAHALANGUR HIMAL SUB-RANGE OF THE HIMALAYAS. THE INTERNATIONAL BORDER BETWEEN NEPAL (PROVINCE NO. 1) AND CHINA (TIBET AUTONO
MOUS REGION) RUNS ACROSS ITS SUMMIT POINT. - REFERENCE WIKIPEDIA)

scala> 
```



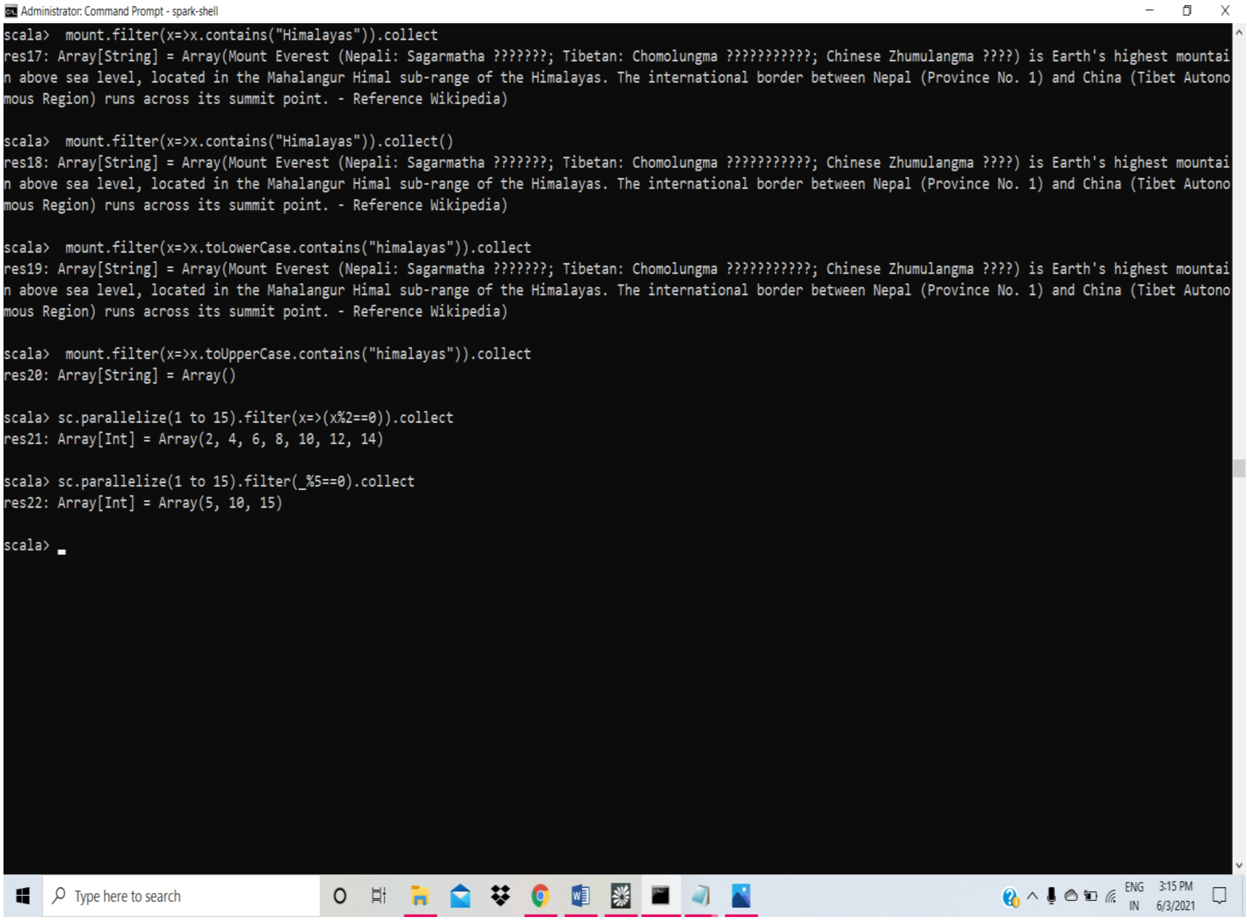
```
scala> val words = mount.map(x=>x.split(""))
words: org.apache.spark.rdd.RDD[Array[String]] = MapPartitionsRDD[44] at map at <console>:25

scala> words.collect()
res25: Array[Array[String]] = Array(Array(M, o, u, n, t, " ", E, v, e, r, e, s, t, " ", (, N, e, p, a, l, i, :, " ", S, a, g, a, r, m, a, t, h, a, " ", ?, ?,
?, ?, ?, ?, ?, ;, " ", T, i, b, e, t, a, n, :, " ", C, h, o, m, o, l, u, n, g, m, a, " ", ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ;, " ", C, h, i, n, e, s, e, " ", Z
, h, u, m, u, l, a, n, g, m, a, " ", ?, ?, ?, ?, ), " ", i, s, " ", E, a, r, t, h, ', s, " ", h, i, g, h, e, s, t, " ", m, o, u, n, t, a, i, n, " ", a, b, o,
v, e, " ", s, e, a, " ", l, e, v, e, l, ,, " ", l, o, c, a, t, e, d, " ", i, n, " ", t, h, e, " ", M, a, h, a, l, a, n, g, u, r, " ", H, i, m, a, l, " ", s, u
, b, -, r, a, n, g, e, " ", o, f, " ", t, h, e, " ", H, i, m, a, l, a, y, a, s, ., " ", T, h, e, " ", i, n, t, e, r, n, a, t, i, o, n, a, l, " ", b, o, r, d,
e, r, "...
```

Flatmap



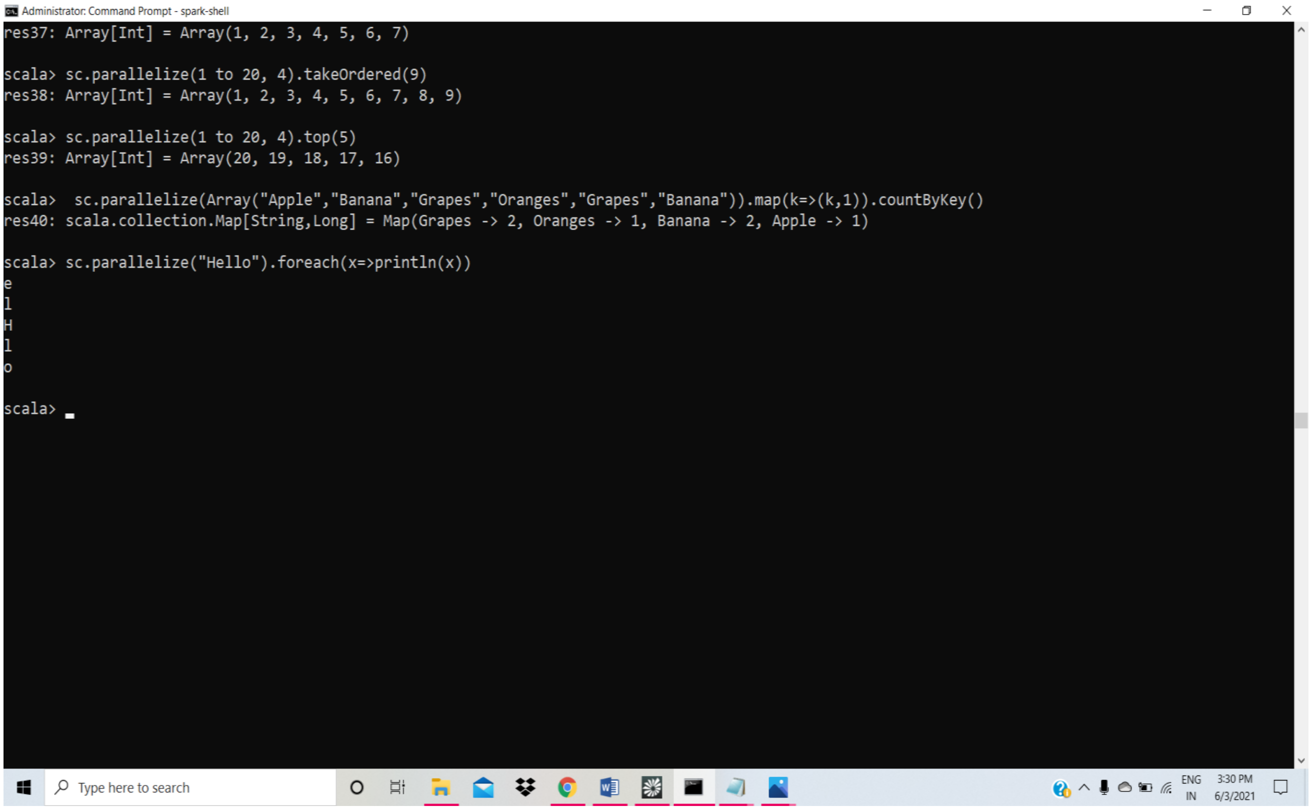
```
scala>  mount.flatMap(x => x.split(" ")).collect
res13: Array[String] = Array(Mount, Everest, (Nepali:, Sagarmatha, ???????;, Tibetan:, Chomolungma, ???????????;, Chinese, Zhumulangma, ????), is, Earth's, hi
ghest, mountain, above, sea, level,, located, in, the, Mahalangur, Himal, sub-range, of, the, Himalayas., The, international, border, between, Nepal, (Provinc
e, No., 1), and, China, (Tibet, Autonomous, Region), runs, across, its, summit, point., -, Reference, Wikipedia)

scala> mount.map(x => x.split(" ")).count()
res14: Long = 1

scala>

scala>  mount.flatMap(x => x.split(" ")).count()
res15: Long = 48

scala> mount.flatMap(x=>x.split(" ")).map(x=>(x, x.length)).collect
res16: Array[(String, Int)] = Array((Mount,5), (Everest,7), ((Nepali:,8), (Sagarmatha,10), (???????;,8), (Tibetan:,8), (Chomolungma,11), (???????????;,12), (C
hinese,7), (Zhumulangma,11), (????),5), (is,2), (Earth's,7), (highest,7), (mountain,8), (above,5), (sea,3), (level,,6), (located,7), (in,2), (the,3), (Mahalan
gur,10), (Himal,5), (sub-range,9), (of,2), (the,3), (Himalayas.,10), (The,3), (international,13), (border,6), (between,7), (Nepal,5), ((Province,9), (No.,3),
(1),2), (and,3), (China,5), ((Tibet,6), (Autonomous,10), (Region),7), (runs,4), (across,6), (its,3), (summit,6), (point.,6), (-,1), (Reference,9), (Wikipedia,
9))

scala>
```

Filter



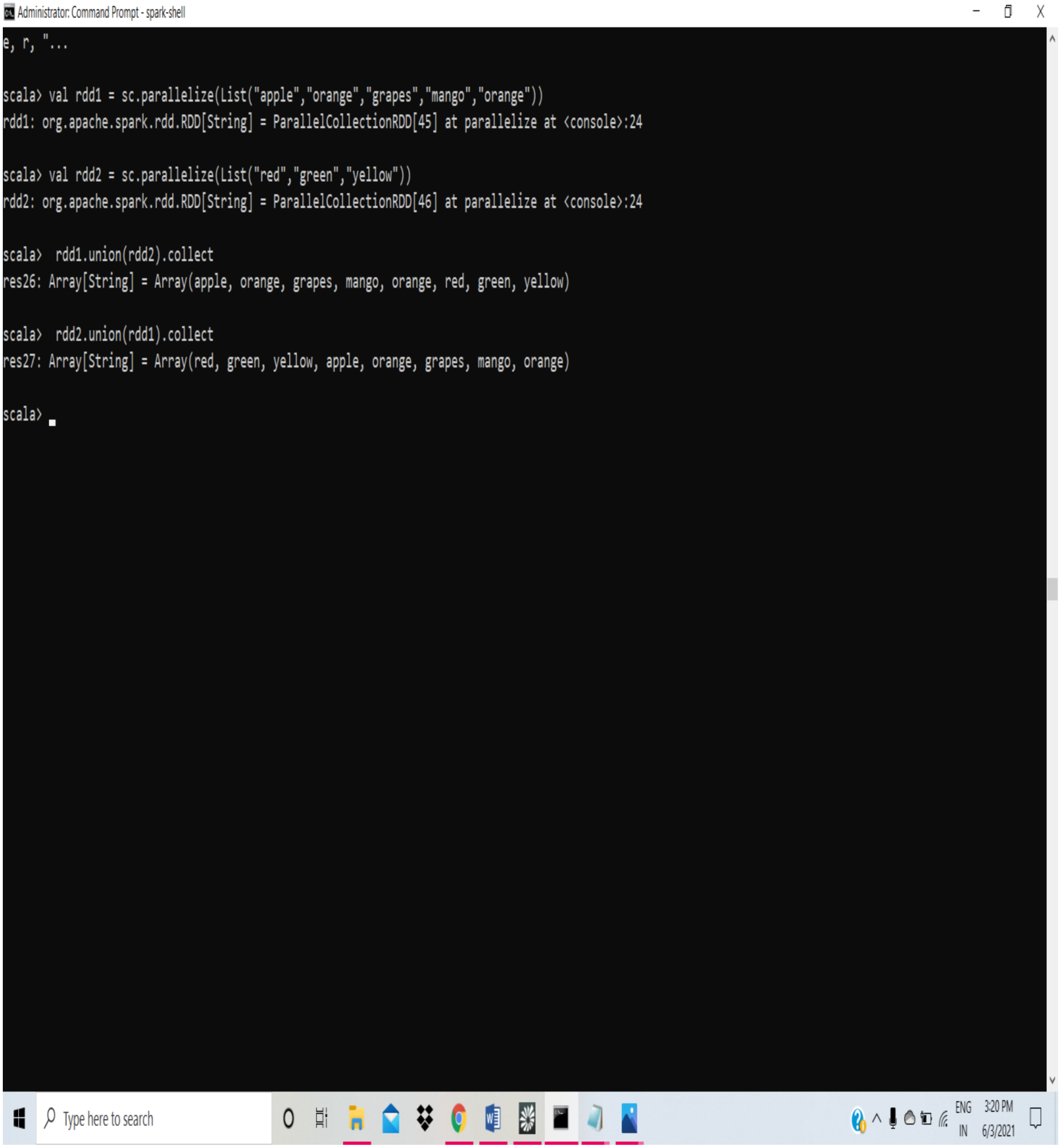
```
scala>  mount.filter(x=>x.contains("Himalayas")).collect
res17: Array[String] = Array(Mount Everest (Nepali: Sagarmatha ???????; Tibetan: Chomolungma ???????????; Chinese Zhumulangma ????) is Earth's highest mountai
n above sea level, located in the Mahalangur Himal sub-range of the Himalayas. The international border between Nepal (Province No. 1) and China (Tibet Autono
mous Region) runs across its summit point. - Reference Wikipedia)

scala>  mount.filter(x=>x.contains("Himalayas")).collect()
res18: Array[String] = Array(Mount Everest (Nepali: Sagarmatha ???????; Tibetan: Chomolungma ???????????; Chinese Zhumulangma ????) is Earth's highest mountai
n above sea level, located in the Mahalangur Himal sub-range of the Himalayas. The international border between Nepal (Province No. 1) and China (Tibet Autono
mous Region) runs across its summit point. - Reference Wikipedia)

scala>  mount.filter(x=>x.toLowerCase.contains("himalayas")).collect
res19: Array[String] = Array(Mount Everest (Nepali: Sagarmatha ???????; Tibetan: Chomolungma ???????????; Chinese Zhumulangma ????) is Earth's highest mountai
n above sea level, located in the Mahalangur Himal sub-range of the Himalayas. The international border between Nepal (Province No. 1) and China (Tibet Autono
mous Region) runs across its summit point. - Reference Wikipedia)

scala>  mount.filter(x=>x.toUpperCase.contains("himalayas")).collect
res20: Array[String] = Array()

scala> sc.parallelize(1 to 15).filter(x=>(x%2==0)).collect
res21: Array[Int] = Array(2, 4, 6, 8, 10, 12, 14)

scala> sc.parallelize(1 to 15).filter(_%5==0).collect
res22: Array[Int] = Array(5, 10, 15)

scala>
```

For each



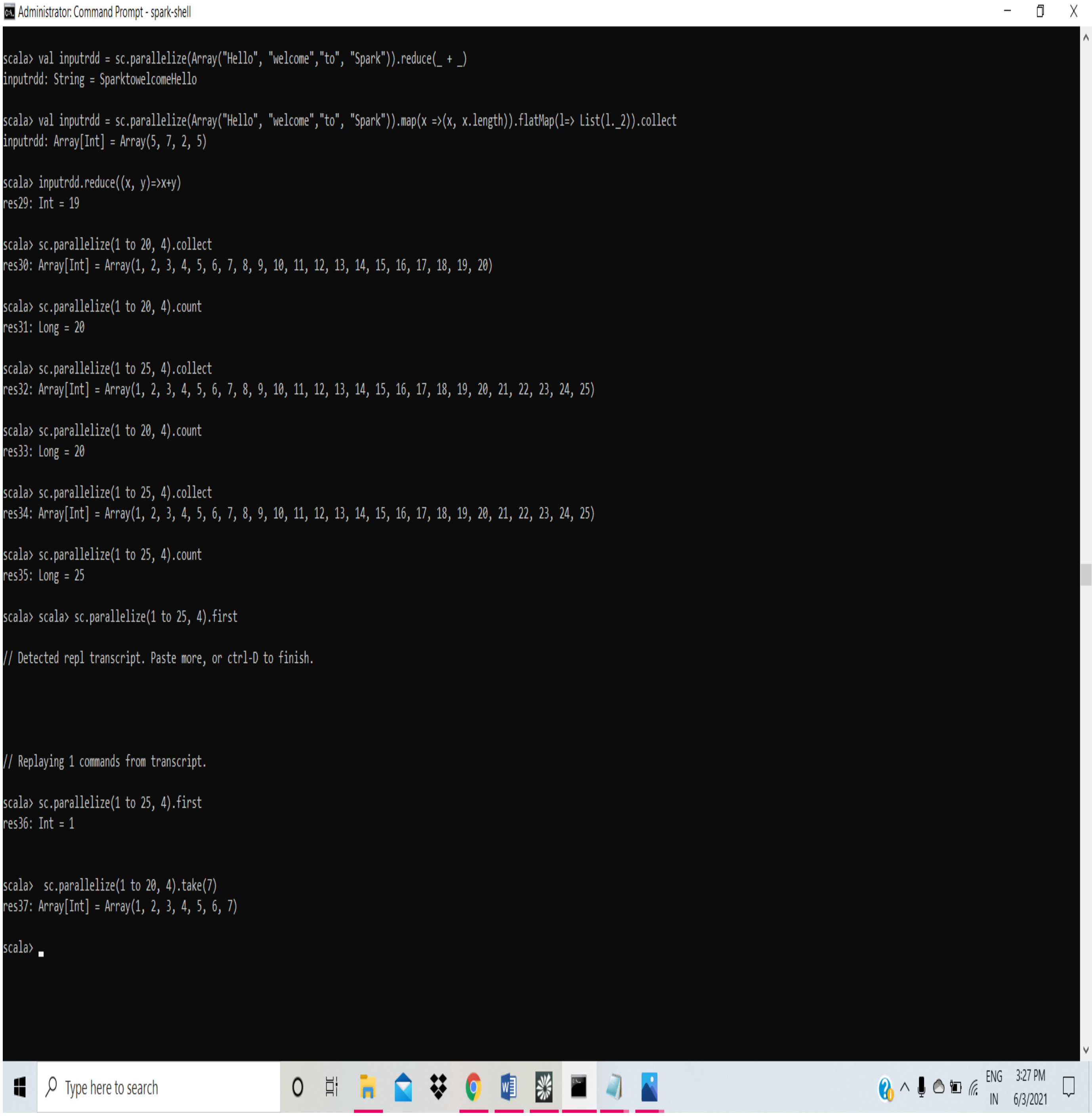
```
res37: Array[Int] = Array(1, 2, 3, 4, 5, 6, 7)

scala> sc.parallelize(1 to 20, 4).takeOrdered(9)
res38: Array[Int] = Array(1, 2, 3, 4, 5, 6, 7, 8, 9)

scala> sc.parallelize(1 to 20, 4).top(5)
res39: Array[Int] = Array(20, 19, 18, 17, 16)

scala>  sc.parallelize(Array("Apple","Banana","Grapes","Oranges","Grapes","Banana")).map(k=>(k,1)).countByKey()
res40: scala.collection.Map[String,Long] = Map(Grapes -> 2, Oranges -> 1, Banana -> 2, Apple -> 1)

scala> sc.parallelize("Hello").foreach(x=>println(x))
e
l
H
l
o

scala>
```

Union



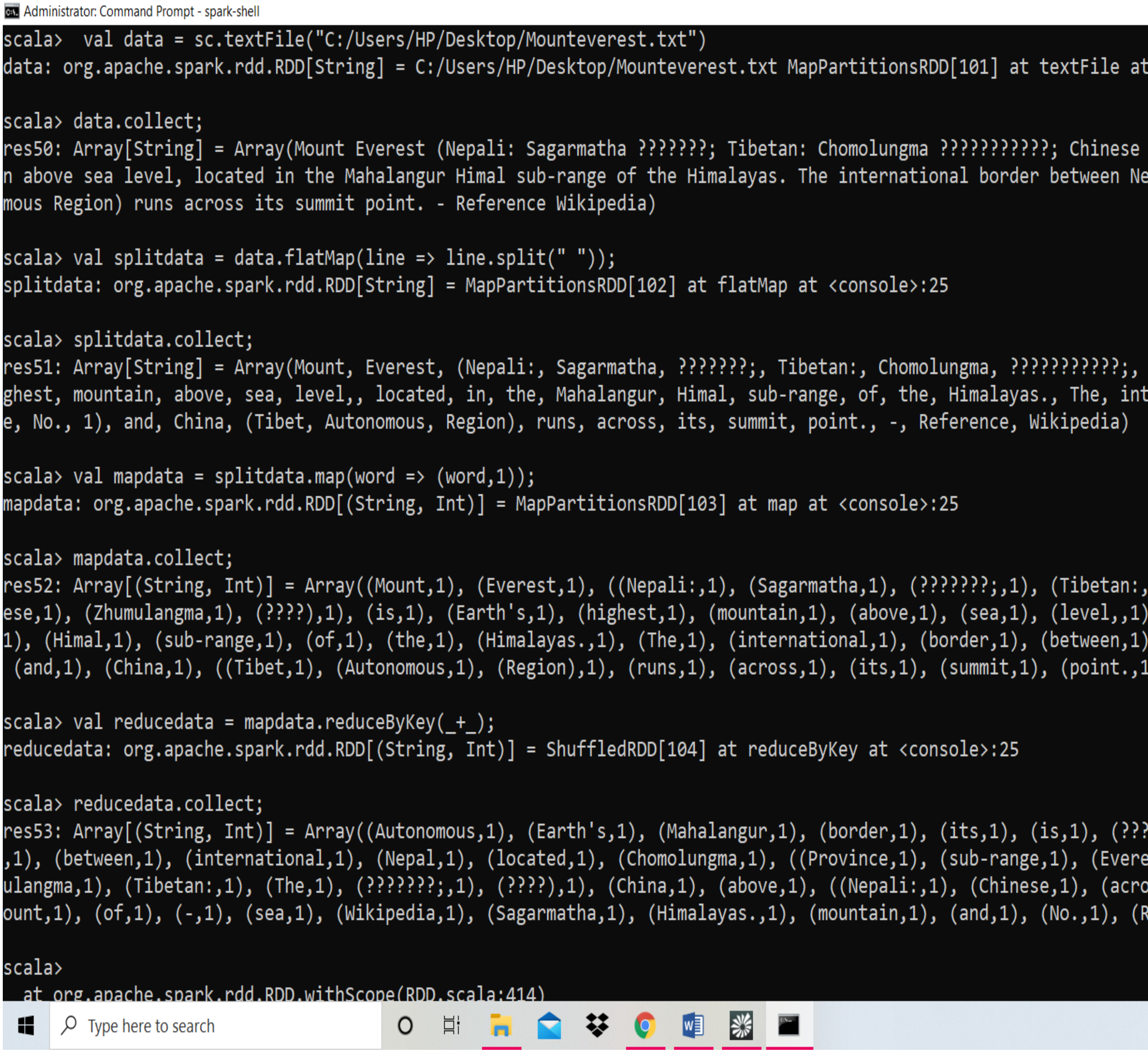
```
e, r, "...

scala> val rdd1 = sc.parallelize(List("apple","orange","grapes","mango","orange"))
rdd1: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[45] at parallelize at <console>:24

scala> val rdd2 = sc.parallelize(List("red","green","yellow"))
rdd2: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[46] at parallelize at <console>:24

scala>  rdd1.union(rdd2).collect
res26: Array[String] = Array(apple, orange, grapes, mango, orange, red, green, yellow)

scala>  rdd2.union(rdd1).collect
res27: Array[String] = Array(red, green, yellow, apple, orange, grapes, mango, orange)

scala>
```

Collect count and first,take



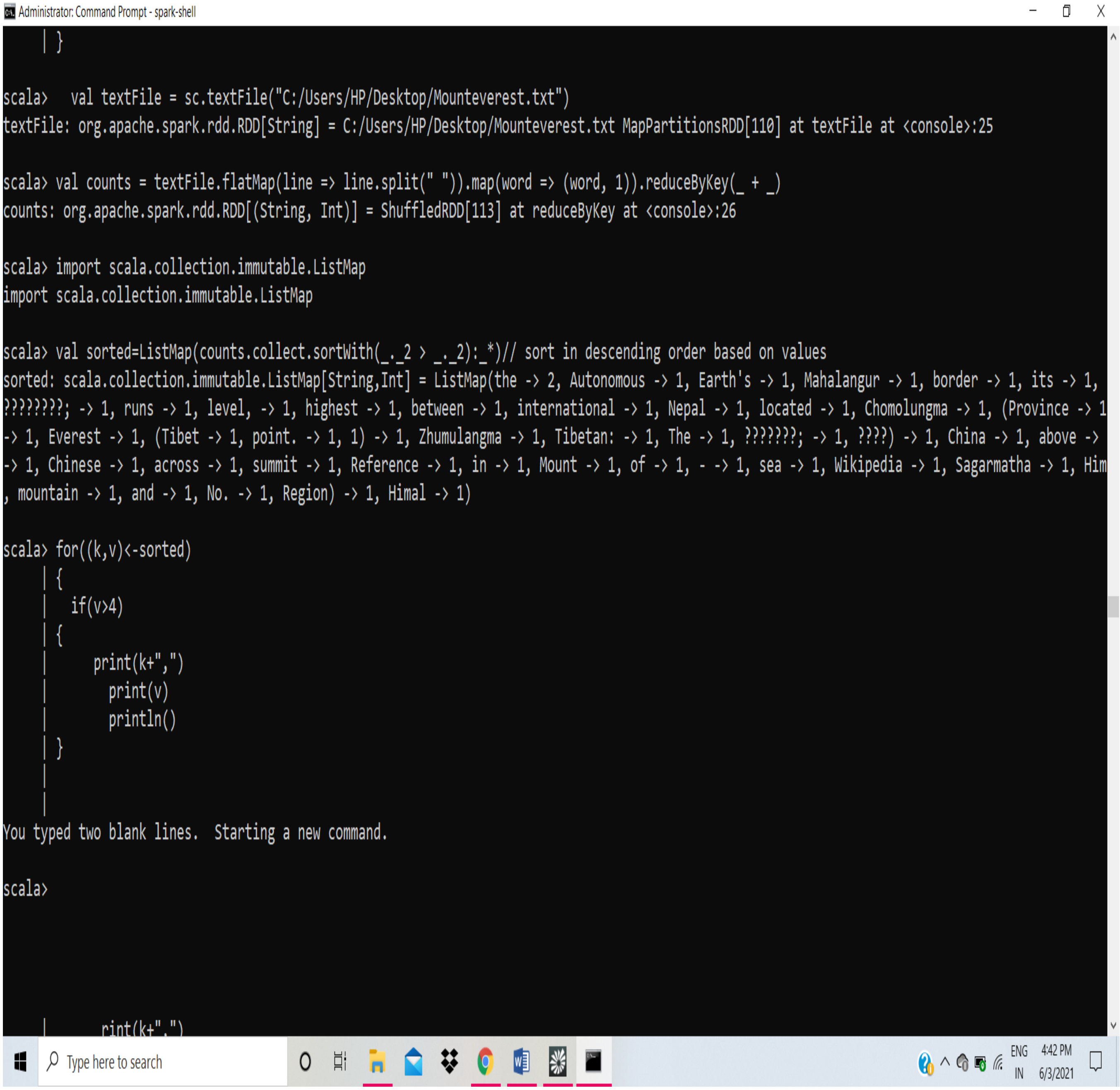
```
scala> val inputrdd = sc.parallelize(Array("Hello", "welcome","to", "Spark")).reduce(_ + _)
inputrdd: String = SparktowelcomeHello

scala> val inputrdd = sc.parallelize(Array("Hello", "welcome","to", "Spark")).map(x =>(x, x.length)).flatMap(l=> List(l._2)).collect
inputrdd: Array[Int] = Array(5, 7, 2, 5)

scala> inputrdd.reduce((x, y)=>x+y)
res29: Int = 19

scala> sc.parallelize(1 to 20, 4).collect
res30: Array[Int] = Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20)

scala> sc.parallelize(1 to 20, 4).count
res31: Long = 20

scala> sc.parallelize(1 to 25, 4).collect
res32: Array[Int] = Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25)

scala> sc.parallelize(1 to 20, 4).count
res33: Long = 20

scala> sc.parallelize(1 to 25, 4).collect
res34: Array[Int] = Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25)

scala> sc.parallelize(1 to 25, 4).count
res35: Long = 25

scala> scala> sc.parallelize(1 to 25, 4).first

// Detected repl transcript. Paste more, or ctrl-D to finish.




// Replaying 1 commands from transcript.

scala> sc.parallelize(1 to 25, 4).first
res36: Int = 1


scala>  sc.parallelize(1 to 20, 4).take(7)
res37: Array[Int] = Array(1, 2, 3, 4, 5, 6, 7)

scala>
```

Worcount program using spark shell



```
scala>  val data = sc.textFile("C:/Users/HP/Desktop/Mounteverest.txt")
data: org.apache.spark.rdd.RDD[String] = C:/Users/HP/Desktop/Mounteverest.txt MapPartitionsRDD[101] at textFile at

scala> data.collect;
res50: Array[String] = Array(Mount Everest (Nepali: Sagarmatha ???????; Tibetan: Chomolungma ???????????; Chinese
n above sea level, located in the Mahalangur Himal sub-range of the Himalayas. The international border between Ne
mous Region) runs across its summit point. - Reference Wikipedia)

scala> val splitdata = data.flatMap(line => line.split(" "));
splitdata: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[102] at flatMap at <console>:25

scala> splitdata.collect;
res51: Array[String] = Array(Mount, Everest, (Nepali:, Sagarmatha, ???????;, Tibetan:, Chomolungma, ???????????;,
ghest, mountain, above, sea, level,, located, in, the, Mahalangur, Himal, sub-range, of, the, Himalayas., The, int
e, No., 1), and, China, (Tibet, Autonomous, Region), runs, across, its, summit, point., -, Reference, Wikipedia)

scala> val mapdata = splitdata.map(word => (word,1));
mapdata: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[103] at map at <console>:25

scala> mapdata.collect;
res52: Array[(String, Int)] = Array((Mount,1), (Everest,1), ((Nepali:,1), (Sagarmatha,1), (???????;,1), (Tibetan:,
ese,1), (Zhumulangma,1), (????),1), (is,1), (Earth's,1), (highest,1), (mountain,1), (above,1), (sea,1), (level,,1)
1), (Himal,1), (sub-range,1), (of,1), (the,1), (Himalayas.,1), (The,1), (international,1), (border,1), (between,1)
 (and,1), (China,1), ((Tibet,1), (Autonomous,1), (Region),1), (runs,1), (across,1), (its,1), (summit,1), (point.,1

scala> val reducedata = mapdata.reduceByKey(_+_);
reducedata: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[104] at reduceByKey at <console>:25

scala> reducedata.collect;
res53: Array[(String, Int)] = Array((Autonomous,1), (Earth's,1), (Mahalangur,1), (border,1), (its,1), (is,1), (???
,1), (between,1), (international,1), (Nepal,1), (located,1), (Chomolungma,1), ((Province,1), (sub-range,1), (Evere
ulangma,1), (Tibetan:,1), (The,1), (???????;,1), (????),1), (China,1), (above,1), ((Nepali:,1), (Chinese,1), (acro
ount,1), (of,1), (-,1), (sea,1), (Wikipedia,1), (Sagarmatha,1), (Himalayas.,1), (mountain,1), (and,1), (No.,1), (R

scala>
  at org.apache.spark.rdd.RDD.withScope(RDD.scala:414)
```

Using RDD and Flatmap count how many times each word appears in a file and write out a list of words whose count is strictly greater than 4 using Spark.



```
     | }

scala>   val textFile = sc.textFile("C:/Users/HP/Desktop/Mounteverest.txt")
textFile: org.apache.spark.rdd.RDD[String] = C:/Users/HP/Desktop/Mounteverest.txt MapPartitionsRDD[110] at textFile at <console>:25

scala> val counts = textFile.flatMap(line => line.split(" ")).map(word => (word, 1)).reduceByKey(_ + _)
counts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[113] at reduceByKey at <console>:26

scala> import scala.collection.immutable.ListMap
import scala.collection.immutable.ListMap

scala> val sorted=ListMap(counts.collect.sortWith(_._2 > _._2):_*)// sort in descending order based on values
sorted: scala.collection.immutable.ListMap[String,Int] = ListMap(the -> 2, Autonomous -> 1, Earth's -> 1, Mahalangur -> 1, border -> 1, its -> 1,
????????; -> 1, runs -> 1, level, -> 1, highest -> 1, between -> 1, international -> 1, Nepal -> 1, located -> 1, Chomolungma -> 1, (Province -> 1
-> 1, Everest -> 1, (Tibet -> 1, point. -> 1, 1) -> 1, Zhumulangma -> 1, Tibetan: -> 1, The -> 1, ???????; -> 1, ????) -> 1, China -> 1, above ->
-> 1, Chinese -> 1, across -> 1, summit -> 1, Reference -> 1, in -> 1, Mount -> 1, of -> 1, - -> 1, sea -> 1, Wikipedia -> 1, Sagarmatha -> 1, Him
, mountain -> 1, and -> 1, No. -> 1, Region) -> 1, Himal -> 1)

scala> for((k,v)<-sorted)
     | {
     |   if(v>4)
     | {
     |      print(k+",")
     |        print(v)
     |        println()
     | }
     |
     |
You typed two blank lines.  Starting a new command.

scala>





     |      rint(k+",")
```