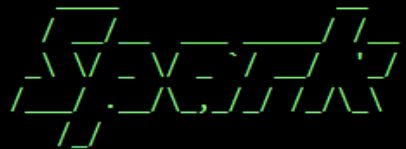


## Module 6: Assignment - 1

1. Find out the count of each word in the 'Shakespeare.txt' dataset

Solution:

```
Spark session available as 'spark'.  
Welcome to
```



```
version 3.0.3  
  
Using Scala version 2.12.10 (OpenJDK 64-Bit Server VM, Java 1.8.0_302)  
Type in expressions to have them evaluated.  
Type :help for more information.  
  
scala> val shakespeare = sc.textFile("/home/bitnami/sparkdata/Shakespeare.txt")  
shakespeare: org.apache.spark.rdd.RDD[String] = /home/bitnami/sparkdata/Shakespe  
are.txt MapPartitionsRDD[1] at textFile at <console>:24
```

Output:

```
scala> shakespeare.flatMap(i => i.split("\\W")).map(i => (i,1)).reduceByKey((x,y) => (x+y)).collect
res2: Array[(String, Int)] = Array((ou,1), (fantasticall,11), (Cloathiers,1), (bone,11), (Prophecies,3),
  (Hats,3), (vailing,1), (Mantua,20), (Fillop,1), (Dollours,1), (gaping,9), (Rancour,2), (Boat,3), (bryer
  ,2), (hem,5), (Friend,165), (forsooth,41), (gainsayes,1), (compulsieue,2), (been,57), (tormento,1), (ston
  ie,2), (fuller,2), (maintaine,37), (pig,4), (accomplished,2), (crying,27), (Sought,1), (Galliard,3), (br
  eath,241), (battering,2), (continuantly,1), (contemptible,2), (Vrine,3), (Ciuet,3), (tongued,1), (fowl,1
  ), (swain,1), (Haruest,13), (Cryes,3), (leafes,1), (Accur,2), (Carpenter,6), (Nathan,1), (afterward,8),
  (Supreame,1), (Stopt,1), (Satisfie,4), (supporter,2), (Herefords,1), (inquisition,1), (Abates,1), (oathe
  s,35), (lightens,3), (stern,2), (Eskales,1)...
```

2. Display the most commonly used words (words with the count over 100 are considered common)

Solution:

```
scala> shakespeare.flatMap(i => i.split("\\W")).map(i => (i,1)).reduceByKey((x,y) => (x+y)).filter(_._2 > 100).sortBy(_._2).collect
```

Output:

```
scala> shakespeare.flatMap(i => i.split("\\W")).map(i => (i,1)).reduceByKey((x,y) => (x+y)).filter(_._2 > 100).sortBy(_._2).collect
res4: Array[(String, Int)] = Array((already,101), (fetch,101), (walke,101), (Stand,101), (Death,101), (Clau,101), (Hamlet,101), (fauour,102), (Gent,102), (louing,102), (Falstaffe,103), (First,103), (enter,103), (appeare,103), (mother,103), (Nurse,103), (Earle,103), (deed,104), (liuing,104), (mercy,104), (losse,104), (offer,104), (behold,104), (Faith,104), (speakes,104), (Brothers,104), (ten,104), (Timon,104), (Ser,105), (sport,105), (Ape,105), (ll,105), (Sil,105), (Earth,105), (Husband,105), (company,105), (childe,105), (strength,105), (blacke,106), (foot,106), (Pol,106), (water,106), (along,106), (read,107), (Cel,107), (Buckingham,107), (yeeld,107), (wonder,107), (speech,108), (Hero,108), (whole,108), (Martius,108), (Cassi,108), (sword,108), (War,109), (person...
```

3. Display the words that are rarely used (words with the count below 30 are considered rare)

Solution:

```
scala> shakespeare.flatMap(i => i.split("\\W")).map(i => (i,1)).reduceByKey((x,y) => (x+y)).filter(_._2 < 30).sortBy(_._2, false).collect
```

Output:

```
scala> shakespeare.flatMap(i => i.split("\\W")).map(i => (i,1)).reduceByKey((x,y) => (x+y)).filter(_._2 < 30).sortBy(_._2, false).collect
res6: Array[(String, Int)] = Array((couer,29), (wouldst,29), (thicke,29), (Among,29), (Whil,29), (bottom
e,29), (Quarrell,29), (Reason,29), (curtesie,29), (Romans,29), (Mir,29), (behind,29), (Robin,29), (Terti
us,29), (manie,29), (Players,29), (Scene,29), (notice,29), (fairest,29), (Courage,29), (loud,29), (Soft,
29), (conduct,29), (boyes,29), (tydings,29), (Cell,29), (Strike,29), (likely,29), (Tree,29), (cap,29), (
ended,29), (title,29), (partly,29), (reuerend,29), (Pride,29), (T,29), (reading,29), (infinite,29), (dre
w,29), (wed,29), (courage,29), (Scepter,29), (dwell,29), (bleeding,29), (friendship,29), (Ed,29), (Deare
,29), (wisely,29), (Hortensio,29), (lately,29), (supply,29), (store,29), (greatnesse,29), (Winter,29), (
reach,29), (Nose,29), (sleepes,29), (supper...
```

4. Display the most commonly used word 5.

```
scala> shakespeare.flatMap(i => i.split("\\W")).map(i => (i,1)).reduceByKey((x,y) => (x+y)).sortBy(_._2, false).take(5)
res10: Array[(String, Int)] = Array(("",224391), (the,21691), (I,20091), (and,16677), (to,15168))
```

Output:

```
scala> shakespeare.flatMap(i => i.split("\\W")).map(i => (i,1)).reduceByKey((x,y) => (x+y)).sortBy(_._2).take(5)
res11: Array[(String, Int)] = Array((ou,1), (Cloathiers,1), (vailing,1), (Fillop,1), (Dollours,1))
```