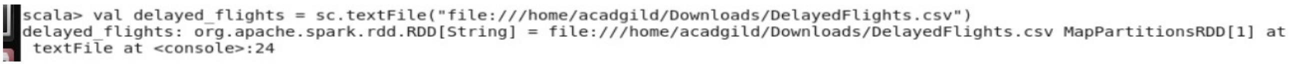
BigData Assignment 8.4

Dataset -
https://drive.google.com/file/d/0B_Qjau8wv1KoWTVDUVFOdzlJNWM/view
Delayed_Flights.csv

Solution -
Loaded the csv file into a rdd text file.



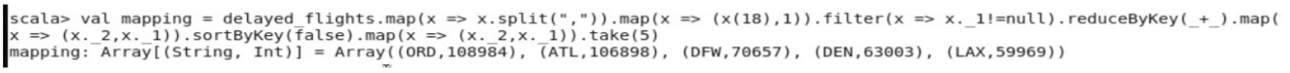
```
scala> val delayed_flights = sc.textFile("file:///home/acadgild/Downloads/DelayedFlights.csv")
delayed_flights: org.apache.spark.rdd.RDD[String] = file:///home/acadgild/Downloads/DelayedFlights.csv MapPartitionsRDD[1] at
 textFile at <console>:24
```

```
val delayed_flights =
sc.textFile("file:///home/acadgild/Downloads/DelayedFlights.csv")
```

1. Find out the top 5 most visited destinations.

Ans – First the data is splitted where ',' is there where each row is mapped to a destination and its count is set initially to 1.And then null is filtered and count is summed up using reduceByKey function and its sorted in descending order by key.

```
val mapping = delayed_flights.map(x => x.split(",")).map(x =>
(x(18),1)).filter(x =>x._1!=null).reduceByKey(_+_).map(x =>
(x._2,x._1)).sortByKey(false).map(x => (x._2,x._1)).take(5)
```

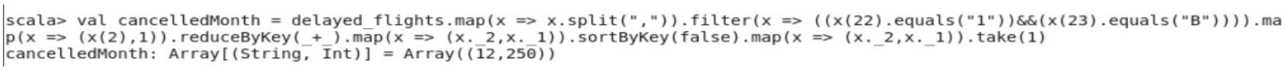


```
scala> val mapping = delayed_flights.map(x => x.split(",")).map(x => (x(18),1)).filter(x => x._1!=null).reduceByKey(_+_).map(
x => (x._2,x._1)).sortByKey(false).map(x => (x._2,x._1)).take(5)
mapping: Array[(String, Int)] = Array((ORD,108984), (ATL,106898), (DFW,70657), (DEN,63003), (LAX,59969))
```

2. Which month has seen the most number of cancellations due to bad weather?

Ans - First the data is spliied where ',' is available and then filtered where fight is cancelled due to weather. Each month is mapped with count is summed up using reduceByKey function and its sorted in descending order by key.

```
val canceled = delayed_flights.map(x => x.split(",")).filter(x =>
((x(22).equals("1"))&& (x(23).equals("B")))).map(x =>
(x(2),1)).reduceByKey(_+_).map(x =>
(x._2,x._1)).sortByKey(false).map(x => (x._2,x._1)).take(1)
```



```
scala> val cancelledMonth = delayed_flights.map(x => x.split(",")).filter(x => ((x(22).equals("1"))&&(x(23).equals("B")))).ma
p(x => (x(2),1)).reduceByKey(_+_).map(x => (x._2,x._1)).sortByKey(false).map(x => (x._2,x._1)).take(1)
cancelledMonth: Array[(String, Int)] = Array((12,250))
```

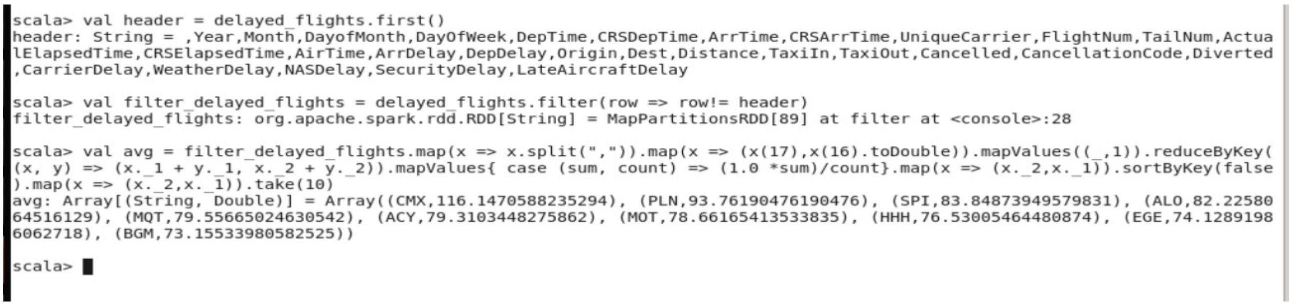
3. Top ten origins with the highest AVG departure delay

Ans – Header is stored. Then filtered data without header.First the data is splitted where ',' is there where each row is mapped to a origin and its count is set initially to 1.And then null is filtered
and count is summed up using reduceByKey function and its sorted in descending order by key.

```
val header =  delayed_flights.first()
```

```
val filter_delayed_flights = delayed_flights.filter(row => row!=
header)
```

```
val avg = filter_delayed_flights.map(x => x.split(",")).map(x =>
(x(17),x(16).toDouble)).mapValues((_,1)).reduceByKey((x, y) => (x._1
+ y._1, x._2 + y._2)).mapValues{ case (sum, count) => (1.0
*sum)/count}.map(x => (x._2,x._1)).sortByKey(false).map(x =>
(x._2,x._1)).take(10)
```

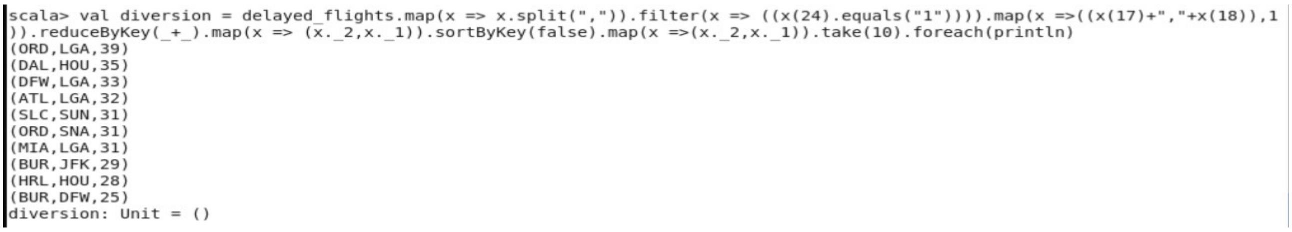


```
scala> val header = delayed_flights.first()
header: String = ,Year,Month,DayofMonth,DayOfWeek,DepTime,CRSDepTime,ArrTime,CRSArrTime,UniqueCarrier,FlightNum,TailNum,Actua
lElapsedTime,CRSElapsedTime,AirTime,ArrDelay,DepDelay,Origin,Dest,Distance,TaxiIn,TaxiOut,Cancelled,CancellationCode,Diverted
,CarrierDelay,WeatherDelay,NASDelay,SecurityDelay,LateAircraftDelay

scala> val filter_delayed_flights = delayed_flights.filter(row => row!= header)
filter_delayed_flights: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[89] at filter at <console>:28

scala> val avg = filter_delayed_flights.map(x => x.split(",")).map(x => (x(17),x(16).toDouble)).mapValues((_,1)).reduceByKey(
(x, y) => (x._1 + y._1, x._2 + y._2)).mapValues{ case (sum, count) => (1.0 *sum)/count}.map(x => (x._2,x._1)).sortByKey(false
).map(x => (x._2,x._1)).take(10)
avg: Array[(String, Double)] = Array((CMX,116.1470588235294), (PLN,93.76190476190476), (SPI,83.84873949579831), (ALO,82.22580
64516129), (MQT,79.55665024630542), (ACY,79.3103448275862), (MOT,78.66165413533835), (HHH,76.53005464480874), (EGE,74.1289198
6062718), (BGM,73.15533980582525))

scala> █
```

4. Which route (origin & destination) has seen the maximum diversion?

Ans - First the data is splitted where ',' is and then filtered there where each row is mapped to a origin and destination and its count is set initially to 1.And then count is summed up using reduceByKey function and its sorted in descending order by key.

```
val diversion = delayed_flights.map(x => x.split(",")).filter(x =>
((x(24).equals("1")))).map(x=>((x(17)+","+x(18)),1)).reduceByKey(_+
_).map(x => (x._2,x._1)).sortByKey(false).map(x
=>(x._2,x._1)).take(10).foreach(println)
```



```
scala> val diversion = delayed_flights.map(x => x.split(",")).filter(x => ((x(24).equals("1")))).map(x =>((x(17)+","+x(18)),1
)).reduceByKey(_+_).map(x => (x._2,x._1)).sortByKey(false).map(x =>(x._2,x._1)).take(10).foreach(println)
(ORD,LGA,39)
(DAL,HOU,35)
(DFW,LGA,33)
(ATL,LGA,32)
(SLC,SUN,31)
(ORD,SNA,31)
(MIA,LGA,31)
(BUR,JFK,29)
(HRL,HOU,28)
(BUR,DFW,25)
diversion: Unit = ()
```