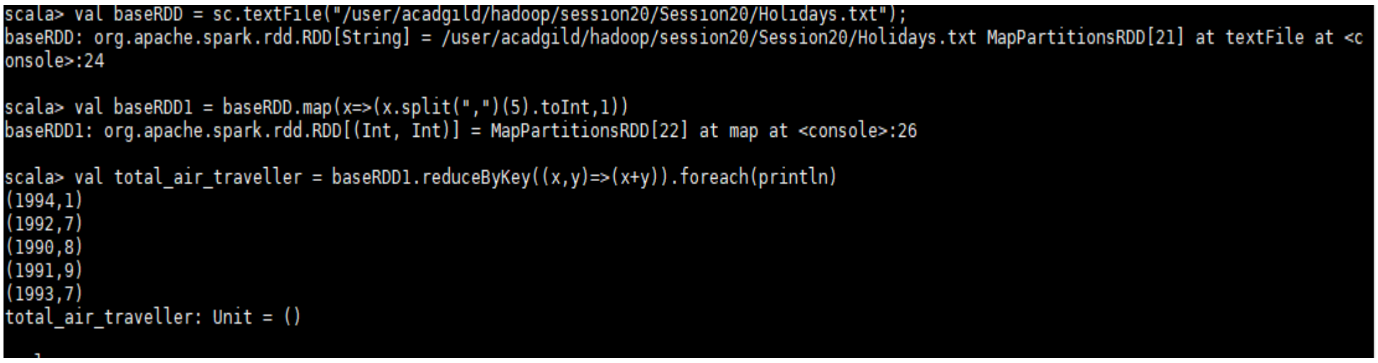
Task 1 - What is the distribution of the total number of air-travelers per year?

Codes used to achieve the above,

1. ***val baseRDD1 = baseRDD.map(x => (x.split(",")(5).toInt,1))***
2. ***val no_air_travelers = baseRDD1.reduceByKey((x,y)=>(x+y)).foreach(println)***

we are creating a tuple RDD baseRDD1 and mapping the key with numerical value 1. After which we are reducing the number of occurrences using reduceByKey and printing the result. Therefore, Total no of air travellers per year is,



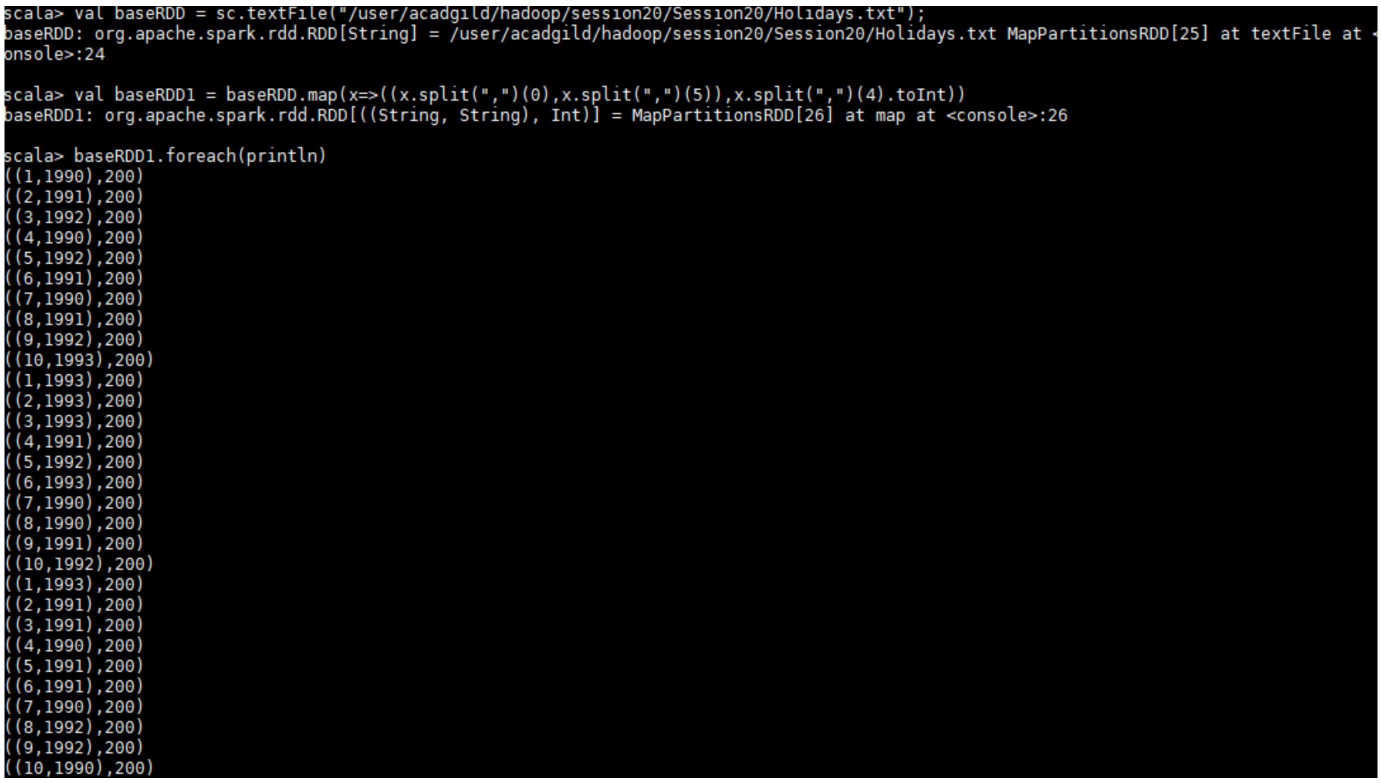
```
scala> val baseRDD = sc.textFile("/user/acadgild/hadoop/session20/Session20/Holidays.txt");
baseRDD: org.apache.spark.rdd.RDD[String] = /user/acadgild/hadoop/session20/Session20/Holidays.txt MapPartitionsRDD[21] at textFile at <c
onsole>:24

scala> val baseRDD1 = baseRDD.map(x=>(x.split(",")(5).toInt,1))
baseRDD1: org.apache.spark.rdd.RDD[(Int, Int)] = MapPartitionsRDD[22] at map at <console>:26

scala> val total_air_traveller = baseRDD1.reduceByKey((x,y)=>(x+y)).foreach(println)
(1994,1)
(1992,7)
(1990,8)
(1991,9)
(1993,7)
total_air_traveller: Unit = ()
```

Task 2 - What is the total air distance covered by each user per year?

Codes used to achieve the above,

1. ***val baseRDD2 = baseRDD.map(x => ((x.split(",")(0),x.split(",")(5)),x.split(",")(4).toInt))***
2. ***val distance_user = baseRDD2.reduceByKey((x,y) => (x + y)).foreach(println)***

We are creating a tuple rdd “baseRDD2” and mapping the key and value. Here the userID, year acts as key and the travel distance is value.



```
scala> val baseRDD = sc.textFile("/user/acadgild/hadoop/session20/Session20/Holidays.txt");
baseRDD: org.apache.spark.rdd.RDD[String] = /user/acadgild/hadoop/session20/Session20/Holidays.txt MapPartitionsRDD[25] at textFile at <
onsole>:24

scala> val baseRDD1 = baseRDD.map(x=>((x.split(",")(0),x.split(",")(5)),x.split(",")(4).toInt))
baseRDD1: org.apache.spark.rdd.RDD[((String, String), Int)] = MapPartitionsRDD[26] at map at <console>:26

scala> baseRDD1.foreach(println)
((1,1990),200)
((2,1991),200)
((3,1992),200)
((4,1990),200)
((5,1992),200)
((6,1991),200)
((7,1990),200)
((8,1991),200)
((9,1992),200)
((10,1993),200)
((1,1993),200)
((2,1993),200)
((3,1993),200)
((4,1991),200)
((5,1992),200)
((6,1993),200)
((7,1990),200)
((8,1990),200)
((9,1991),200)
((10,1992),200)
((1,1993),200)
((2,1991),200)
((3,1991),200)
((4,1990),200)
((5,1991),200)
((6,1991),200)
((7,1990),200)
((8,1992),200)
((9,1992),200)
((10,1990),200)
```

Task 3 - Which user has travelled the largest distance till date?

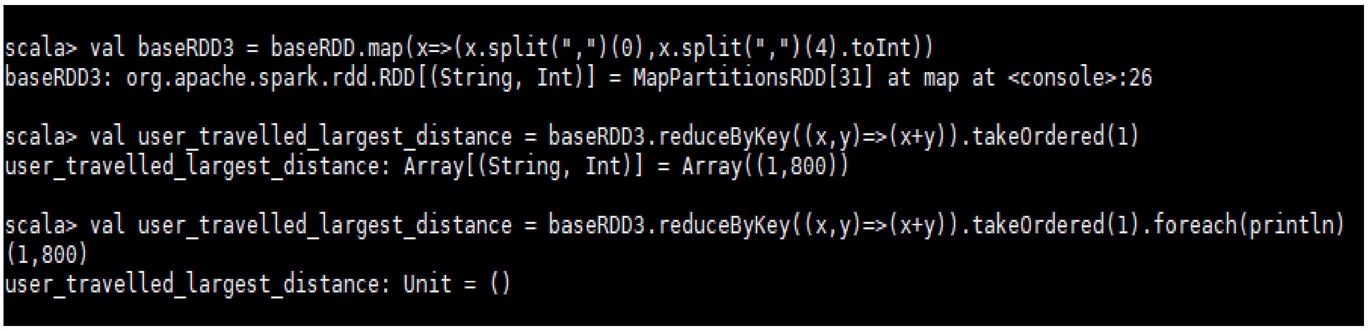
Codes used below,

1. ***val baseRDD3 = baseRDD.map(x=> (x.split(",")(0),x.split(",")(4).toInt))***
2. ***val largest_dist = baseRDD3.reduceByKey((x,y)=>(x+y)).takeOrdered(1)***

The tuple rdd "baseRDD3" is created to map the key and value from the baseRDD. Here the userID and is key and the travel distance is value,

In the 2nd step, we are reducing the number of occurrences using reduceByKey and using the takeOrdered function to get the result,

largest_dist: Array[(String, Int)] = Array((1,800))



```
scala> val baseRDD3 = baseRDD.map(x=>(x.split(",")(0),x.split(",")(4).toInt))
baseRDD3: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[31] at map at <console>:26

scala> val user_travelled_largest_distance = baseRDD3.reduceByKey((x,y)=>(x+y)).takeOrdered(1)
user_travelled_largest_distance: Array[(String, Int)] = Array((1,800))

scala> val user_travelled_largest_distance = baseRDD3.reduceByKey((x,y)=>(x+y)).takeOrdered(1).foreach(println)
(1,800)
user_travelled_largest_distance: Unit = ()
```