Week 5 Assessment Task

Part A - Spark RDD with text (8 marks)

Detecting popular and trending topics from the news articles is an important task for public opinion monitoring. In Part A your task is to perform text data analysis over a dataset of Australian news from ABC (Australian Broadcasting Corporation) using Spark RDD. You can compare your solution with that in Week 5 using MRJob.

The dataset you are going to use contains data of news headlines published over several years. In this text file, each line is a **headline of a news article**, in format of "*date,* term1 term2 ... ... ". The lines are sorted by the date, and the terms are separated by the space character. A sample file is like below:

```
20030219,council chief executive fails to secure position
20030219,council welcomes ambulance levy decision
20030219,council welcomes insurance breakthrough
20030219,fed opp to re introduce national insurance
20040501,cowboys survive eels comeback
20040501,cowboys withstand eels fightback
20040502,castro vows cuban socialism to survive bush
20200401,coronanomics things learnt about how coronavirus economy
20200401,coronavirus at home test kits selling in the chinese community
20200401,coronavirus campbell remess streams bear making classes
20201015,coronavirus pacific economy foriegn aid china
20201016,china builds pig apartment blocks to guard against swine flu
```

When you click the panel on the right you'll get a connection to a server that has, in your home directory, a text file called "abcnews.txt", containing some sample text (feel free to open the file and explore its contents). The entire dataset can be downloaded from https://www.kaggle.com/therohk/million-headlines.

Your task is to **find the top-3 most frequent terms for each year**. That is, for each year, select 3 terms that appeared in the most articles of that year, which represent the hot topics. If some words appear in the same number of articles, sort them in ascending order alphabetically.

Please ignore the "stop words" which are frequent but meaningless for this task, including: "to", "a", "an", "the", "for", "in", "on", "of", "at", "over", "with", "after", "and", "from", "new", "us", "by", "as", "man", "up", "says", "in", "out", "is", "be", "are", "not", "pm", "am", "off", "more", "less", "no", "how".

In your output, sort the results by years. For each year (in one line), sort the top-3 terms first by their article frequencies and then by the terms in alphabetical order. For example, given the above data set, the output should be (using Spark RDD):

```
2003    council insurance welcomes
2004    cowboys eels survive
2020    coronavirus china economy
```

Write a Python program that uses Spark RDD to do this. A file called "rdd.py" has been created for you - you just need to fill in the details. **Note that the efficiency** (the time complexity) of your method will be considered for marking.

To debug your code, you can first test everything in pyspark, and then write the codes in "rdd.py". To test your program, you first need to create your default directory in Hadoop, and then copy abcnews.txt to it:

```
$ hdfs dfs -mkdir -p /user/user
$ hdfs dfs -put abcnews.txt
```

You can run your program on Spark by running the following command:

```
$ spark-submit rdd.py
```

Please save your results in the '**result-rdd**' folder in HDFS.

Part B - Spark RDD with CSV (4 marks)

In Part B your task is to answer a question about the data in a CSV file, first using Spark RDDs, and then using Spark DataFrames and SQL. By using both to answer the same question about the same file you can more readily see how the two techniques compare.

When you click the panel on the right you'll get a connection to a server that has, in your home directory, the CSV file "orders.csv". It's one that you've seen before. Here are the fields in the file:

```
OrderDate (date)
ISBN (string)
Title (string)
Category (string)
PriceEach (decimal)
Quantity (integer)
FirstName (string)
LastName (string)
City (string)
```

Your task is to **find the number of books ordered each day,** sorted by the number of books **descending**, then order date **ascending**.

Your results should appear as the following:

```
2009-04-03,10
2009-04-02,8
2009-04-01,7
2009-04-04,6
2009-03-31,5
2009-04-05,4
2009-04-08,4
```

First (4 marks)

Write a Python program that uses Spark RDDs to do this. A file called "rdd.py" has been created for you - you just need to fill in the details. You should be able to modify programs that you have already seen in this week's content. To sort the RDD results, you can use `SortBy`, and here is an example of it.

Hint:

```
>>> tmp = [('a', 3), ('b', 2), ('a', 1), ('d', 4), ('2', 5)]
>>> sc.parallelize(tmp).sortBy(lambda x: (x[0],x[1])).collect()
```

Output:

```
[('2', 5), ('a', 1), ('a', 3), ('b', 2), ('d', 4)]
```

To test your program you first need to create your default directory in Hadoop, and copy orders.csv to it:

```
$ hdfs dfs -mkdir -p /user/user
$ hdfs dfs -put orders.csv
```

You can test your program by running the following command:

```
$ spark-submit rdd.py
```

Please save your results in the '**result-rdd**' folder in HDFS.