Lab 3.

In this lab, we will continue our investigation on the Amazon fine foods dataset we have been using so far (available in the HFDS shared folder of the BigData@Polito cluster: /data/students/bigdata-01QYD/Lab3/Reviews.csv). Up to now, all the analysis we did were on a "per-row" basis, since we analyzed only the text of the reviews alone. Today we will start looking at the connections between the single reviews, and to do this we will need to transpose the original dataset. The transposed dataset contains one line per reviewer. The first field of each line contains always the id of the reviewer (AXXXXXXXXX), followed by the list of all the products reviewed by her/him (BXXXXXXXXX). Suppose that each reviewer has reviewed at most 1000 products.

Here's a sample of the first lines of such transposed dataset:

```
A09539661HB8JHRFVRDC,B002R8UANK,B002R8J7YS,B002R8SLUY
A1008ULQSWI006,B0017OAQIY
A100EBHBG1GF5,B0013T5YO4
A1017Y0SGBINVS,B0009F3SAK
A101F8M8DPFOM9,B005HY2BRO,B000H7MFVI
A102H88HCCJJAB,B0007A8XV6
A102ME7M2YW2P5,B000FKGT8W
A102QP2OSXRVH,B001EQ5SGU,B000EH0RTS
A102TGNH1D915Z,B000RHXKC6,B0002DHNXC,B0002DHNXC,B000XJK7UG,B00008DFK5,B000
SP1CWW,B0009YD7P2,B000SP1CWW,B00008DFK5,B0009YD7P2
A1051WAJL0HJWH,B000W5U5H6
A1052V04GOA7RV,B002GJ9JY6,B001E5E3JY,B008ZRKZSM,B002GJ9JWS
```

For the following exercise, you can use the sample dataset AmazonTransposedDataset_Sample.txt, which is available in the HDFS shared folder /data/students/bigdata-01QYD/Lab3

You can then reproduce the full transposed dataset (computed on the whole Reviews.csv) on your own (Ex. 2 of this lab) and repeat your investigation on it.

Ex 1. "People also like…"

In this exercise, we try to build a very basic version of a recommending system. Your goal is to find the top 100 pairs of products most often reviewed (and so probably bought) together.

In this exercise, we consider two products as reviewed (i.e., bought) together if they appear in the same line of the input transposed file (the input file is /data/students/bigdata-01QYD/Lab3/AmazonTransposedDataset_Sample.txt). We ignore temporal constraints, so even if a decade or a thousand products have passed between the two reviews, we count the pair, as it represents anyway the tastes of a single user.

We suggest you to implement your application by using the template project contained in Lab3_Skeleton.zip.

Lab3_Skeleton.zip contains two classes that can be useful for implementing the solution of this exercise.

- WordCountWritable
  - This class can be used to store a pair (word, count), where word is a String and count is an Integer.
  - The public WordCountWritable(String word, Integer count) constructor is used to create new WordCountWritable objects.
  - String getWord() and Integer getCount() are used to retrieve the value of word and count, respectively.
  - setWord(String value) and setCount(Integer value) are used to set the value of word and count, respectively.
  - This class implements the Comparable interface. Hence, the public int compareTo(WordCountWritable other) can be used to compare objects of this class.
  - This class implements the Writable interface.

- TopKVector<T extends Comparable<T>>
  - This class can be used to store/manage the top-k objects of a set of objects. The type of managed objects is T. T can be an arbitrary class implementing the Comparable interface.
  - The TopKVector(int k) method is the constructor used to create TopKVector objects. Each object of type TopKVector contains a Vector storing the top-k objects among the set of objects of type T that are inserted in it. Initially, this vector is empty (i.e., initially the top-k vector contains no objects).
  - public void updateWithNewElement(T newElement) is used to insert a new object in the internal top-k vector of the TopKVector object on which the method is invoked. newElement is inserted in the top-k vector if and only if it is in the top-k objects. Otherwise, it is discarded.
  - public Vector<T> getLocalTopK() returns a Vector<T> containing the top-k objects associated with the TopKVector object on which this method is invoked.

The following snippet of code shows how to use these two classes.

```
…..
// Create an object that is used to store/manage a top-3 vector
// containing objects of type WordCountWritable
TopKVector<WordCountWritable> top3 = new TopKVector<WordCountWritable>(3);

// Insert 5 objects of type WordCountWritable in top3
// top3 automatically stores only the top-3 objects
top3.updateWithNewElement(new WordCountWritable(new String("p1,p2"), new Integer(4)));
top3.updateWithNewElement(new WordCountWritable(new String("p1,p3"), new
Integer(40)));
top3.updateWithNewElement(new WordCountWritable(new String("p2,p4"), new Integer(3)));
top3.updateWithNewElement(new WordCountWritable(new String("p5,p6"), new Integer(6)));
top3.updateWithNewElement(new WordCountWritable(new String("p15,p16"), new
Integer(1)));
```

```
// Retrieve the top-k objects from top3
Vector<WordCountWritable> top3Objects = top3.getLocalTopK();

// Print the content of the top-3 selected objects on the standard output
for (WordCountWritable value : top3Objects) {
        System.out.println(value.getWord() + " " + value.getCount());
}

// The following is the output of this snippet of code
// p1,p3 40
// p5,p6 6
// p1,p2 4
….
```

Ex 2. Transposing the Reviews.csv dataset

The original review dataset (available in the HFDS shared folder of the BigData@Polito cluster: /data/students/bigdata-01QYD/Lab3/Reviews.csv) lists all the reviews per-row, and is comma-separated. In each line, two of the columns represent the user id and product id.

Write a MapReduce application that transposes the original review dataset (Reviews.csv), listing for each user id all the products he/she has reviewed, as in the format above specified (i.e., the same format of the sample file AmazonTransposedDataset_Sample.txt).

Save the output of this application (i.e. the transposed dataset) to a HDFS folder named ReviewsTransposed in your HDFS home. Then, use it as input for the application you have built in Ex. 1, and save the output in a HDFS folder named MostReviewedTogether. Keep these folders for the future labs.