CSE-E5430 **Autumn 2015**
Scalable Cloud Computing
Home Assignment 1 - Deadline: 2nd of November 2015 at 12:15 (strict deadline!)

Submit your assignment on MyCourses assignment page. Upload a zip file named as "[student number].zip", packaging all your answer files inside the directory "`assignment-1`" (Your submission should have same file structure as the template package). A template package as a .zip file for the home assignment 1 is available on MyCourses Assignment 1 page. Download it, unpack the files, and modify them to contain your answers. When you are done, pack the files to a zip file. Then upload the package to the MyCourses via assignment submission page. **Submission will be closed after the deadline.**

The home exercises are personal, **no group work allowed!** There are two rounds of home exercises of 10 points each. To pass the home exercises ≥ 10 points are needed and ≥ 16 points gives a $+1$ to the exam grade (no effect to exam grades 0 or 5).

The home assignments require you to have a working Apache Hadoop 2.7.1 installation, please see the MyCourses Hadoop Setup page on how to setup/install Hadoop 2.7.1.

1. a) The directory "`question-1`" contains the "`WordCount.java`" word count example also used in the Tutorials and Lectures. Please run the WordCount example on the file "`vanrikki-stool.txt`" and place all the commands and their outputs used to compile, package, prepare and run the WordCount into the file "`question-1-a-log.txt`" (we are expecting similar output as in Lecture 3, slides 18-25). Also copy the output directory generated by the Hadoop WordCount job from HDFS to subdirectory "`output-stool-a`" of the "`question-1`" directory. Write a short explanation of steps to "`short-explanation-1-a.txt`". (2p)

   b) Do a modification to the "`WordCount.java`" word count example and save and return it as new Java file "`TopCount.java`". The modification to the map-reduce should output the words which appear at least 100 times in the input text. Use combiner to minimize number of key-value pairs generated from each node. Please run your TopCount job on the file "`vanrikki-stool.txt`"

and place all the commands and their outputs used to compile, package, prepare and run the WordCount into the file "`question-1-b-log.txt`" (Again we are expecting similar output as in Lecture 3, slides 18-25). Also copy the output directory generated by the Hadoop TopCount job from HDFS to subdirectory "`output-stool-b`" of the "`question-1`" directory. Write a short explanation of steps to "`short-explanation-1-b.txt`". (3p)

2. a) In this question use the subdirectory "`question-2`". It contains an input file "`author_book_tuple.txt`", where each row is a tuple formatted as JSON which contains an author and a book name. The file in question contains a subset of "`https://openlibrary.org`" data.

   Your task is to create a Hadoop program in "`CombineBooks.java`", provided in the "`question-2`" directory. The program should do the following: Given the input author-book tuples, map-reduce program should procude a JSON object which contains all the books from same author in a JSON array, i.e.

   ```
   {"author": "Tobias Wells", "books":[{"book":"A die in the country"},{"book": "Dinky died"}]}
   ```

   Use combiner to minimize number of key-value pairs generated from each node.

   In addition to the "`CombineBooks.java`" file, also return a run log similar to the ones created in question 1 to the file "`question-2-a-log.txt`". Also copy the output directory generated by the Hadoop CombineBooks job from HDFS to subdirectory "`output-a`" of the "`question-2`" directory. Write a short explanation of steps to "`short-explanation-2-a.txt`". (3p)

   b) In this task use the subdirectory "`question-2`". Your task is to modify the Hadoop program in "`CombineBooks.java`" and return is as "`QueryAuthor.java`". The program should do the following: Given the input author-book tuples, map-reduce program should procude a JSON object which contains all the books from only the **queried author** in a JSON array, i.e.

   ```
   {"author": "Tobias Wells", "books":[{"book":"A die in the country"},{"book": "Dinky died"}]}
   ```

   Use combiner to minimize number of key-value pairs generated from each node.

In addition to the "`QueryAuthor.java`" file, also return a run log for author "J. K. Rowling", similar to the ones created in question 1 to the file "`question-2-b-log.txt`". Also copy the output directory generated by the Hadoop QueryAuthor job for J. K. Rowling from HDFS to subdirectory "`J. K. Rowling-output-b`" of the "`question-2`" directory. Write a short explanation of steps to "`short-explanation-2-b.txt`". (2p)

PS: For compiling your code and running the Hadoop jobs you can get help from content of `~/WordCount/run_commands` file on Hadoop 2.7.1 virtual machine. (available on `www.cs.hut.fi/~ahmedh1/hadoop-ubuntu.ova`)