**Assignment 1 Task 1 – Description of my code**

**Task 1: common word count**

|  |  |
| --- | --- |
| Working directory | WordCount folder Data: Task1\_data folder |
| Input | stopwords.txt, task1-input1.txt, task1-input2.txt |
| Final Output | Task1\_data/output/step4 |

|  |  |  |  |
| --- | --- | --- | --- |
| Steps | Input | Functionality | Output |
| Step 1 | stopwords.txt  task1-input1.txt | TokenizerWCMapper reads in input1.txt, and decomposes the file to tokens and removes the stopwords in the tokens.  Input: file text  Output: <word, 1>  IntSumReducer sums the token count and outputs word count in input1.txt  Input: <word, 1>  Output: <word, count> | output/step1 |
| Step 2 | stopwords.txt  task1-input2.txt | TokenizerWCMapper reads in input2.txt, and decomposes the file to tokens and removes the stopwords in the tokens.  Input: file text  Output: <word, 1>  IntSumReducer sums the token count and outputs word count in input2.txt  Input: <word, 1>  Output: <word, count> | output/step2 |
| Step 3 | output/step1  output/step2 | Mapper1 reads word count from file 1  Mapper2 reads word count from file 2  Input: <word, count>  Output: <word, count>  CountCommonReducer takes the smaller one of the 2 counts from 2 files (if both files have the word). And then outputs common word count.  Input: <word, count>  Output: <word, common word count> | output/step3 |
| Step 4 | output/step3 | SortMapper reads the common word count from step 3 output.  Input: <word, common word count>  Output: <common word count, word>  ReverseComparator is implemented to sort the output from Mapper in descending order based on the count.  SortReducer writes the top 15 sorted output to the file.  Input: <sorted common word count, word>  Output: <top 15 common word count, word> | output/step4 |