**Assignment:**

Create a multi-threaded application that will read a text file and does a word count. Use any programming language of your choice.

Program Input : <http://www.gutenberg.org/files/2600/2600-0.txt>

         Read the above document in the program and split the document automatically into several parts, to

compute the below in parallel -

For each part :

        Find total words in the part

       Compute list of words and how many times they appear in the part

       Identify the most used word and how many times they appear in the part

Output: Print Summary

   Aggregate results from all parts and display a summary of Total Word count and top 5 Most used words and how many times they appear in the document

Upload the code to github.

Will do today night

Uploaded.

The below are the url

Extra Credit

- Theoretically explain strategies on how the execution time could be improved and other scaling techniques. How would you re-architect your program if the input size is not ~ 3MB, but 1 TB?

**Brief Explanation :**

---In case of 1 TB file size data (1024 GB) will apply the same log for splitting into chunks of 1024 parts and each of 1 GB and Also try to utilise the maximum resources available in the system ( Processors , Thread).

Alternatively, Move the file into HDFS via Java jscp code and as per hdfs-site.xml , the moment file moved to hdfs it will be splited automatically into block of 256/128 mb  and will be distributed  across the cluster.

Further,  We can run any of Process like Map Reduce, Or Spark job to achieve the objective.

For improving the execution time proving couple of approaches

<Code level and cost level >

--- Caching id the best way to maintain set of data into the map and In case of reprocessing return from stored cache.

----Also we can maintain, some set of line into cache with result and If same record is going to process return result from cache.

**Incase of Unavailability of HDFS :**

If there is no such availability like HDFS ,

Then best approach is RandomAccessFile API to spilt big file into several chunks (which I did in the same exercise)  and perform the multithread operation over the each individual chunks.

- If there are gaps in requirement, the candidate should fill it creatively.

  Experienced 1 issue occurring as space “ ” .

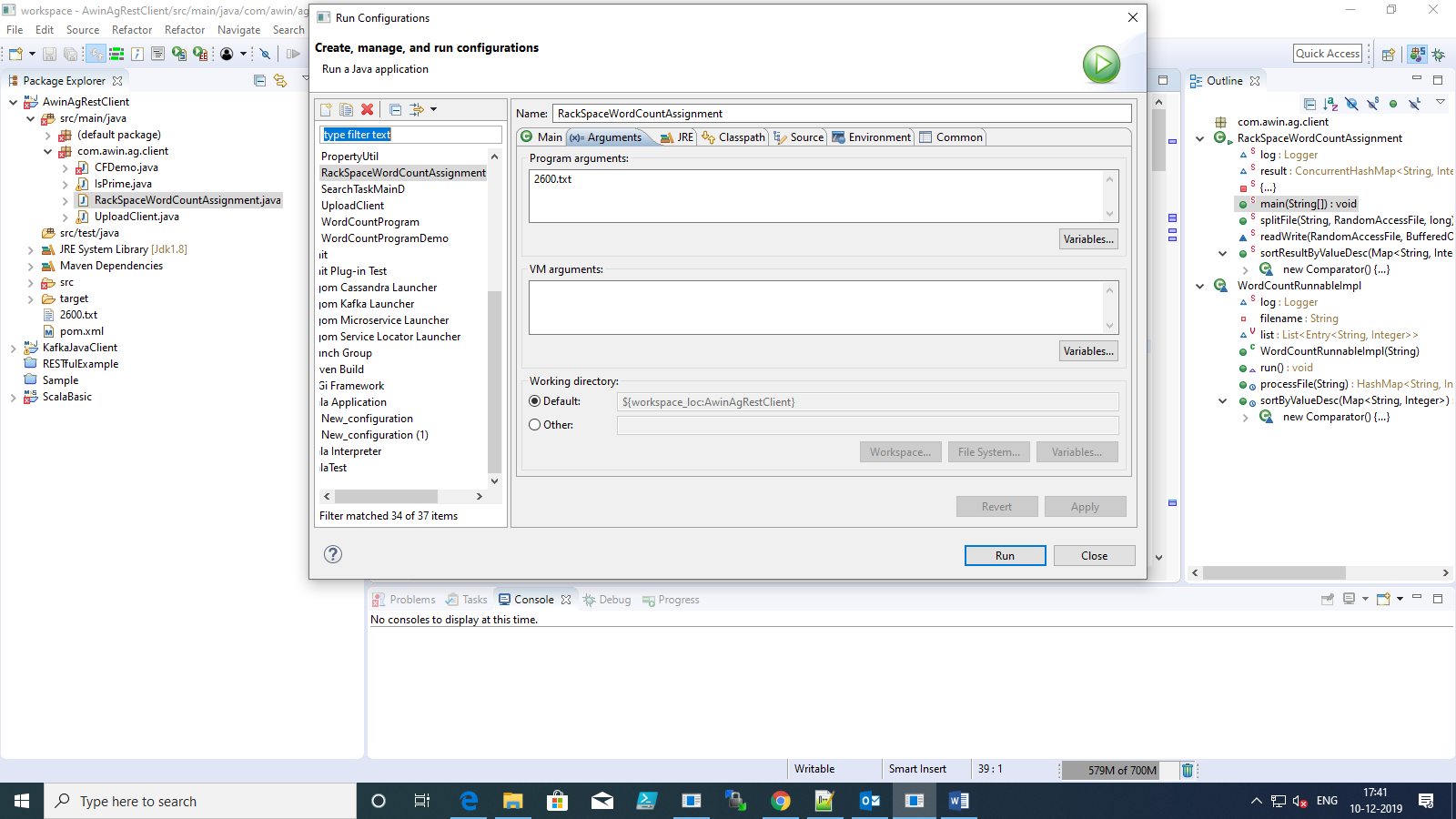
Step to Run the App

1: Copy the file 2600.txt into home folder of project code

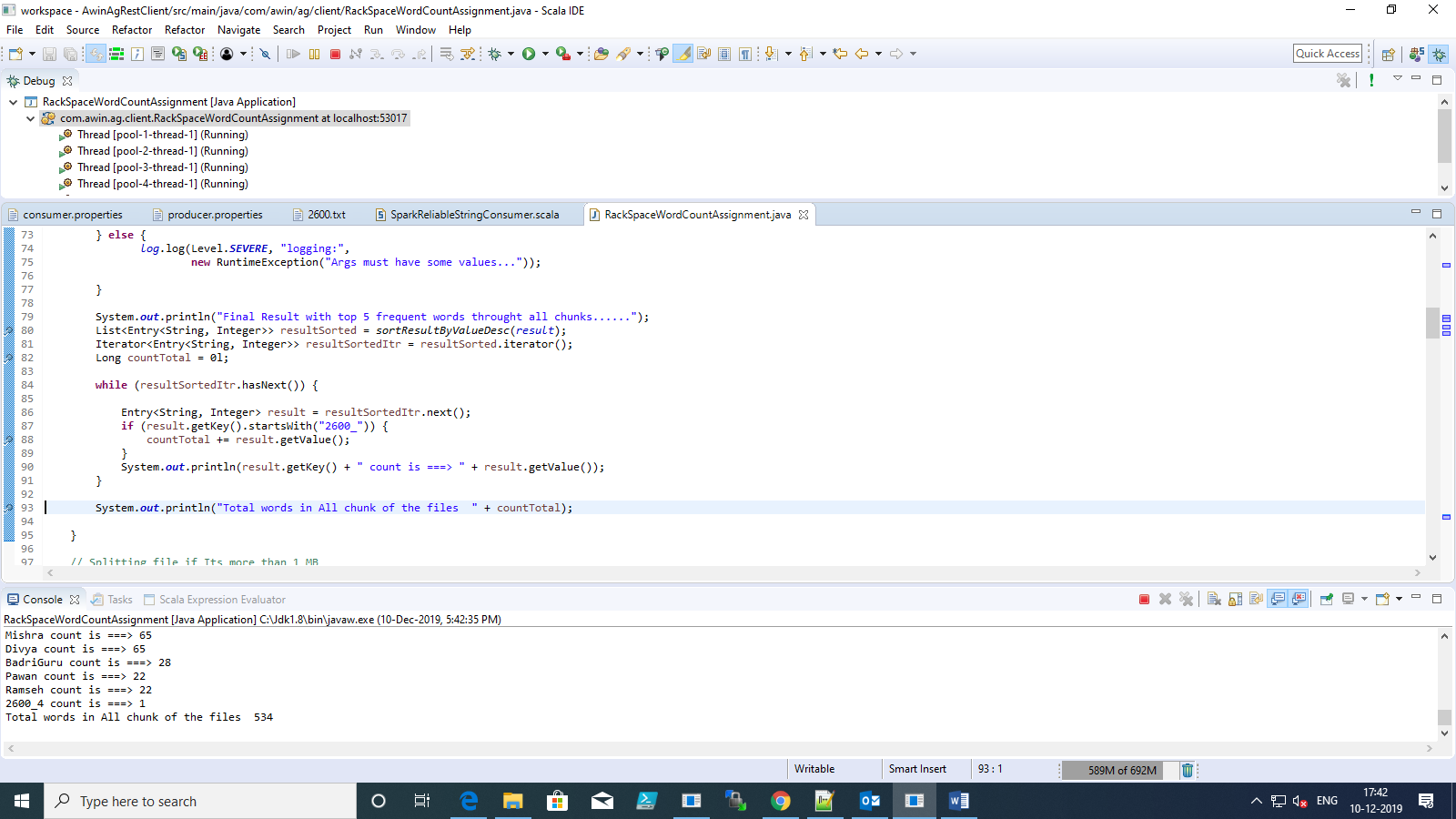
Move java file to some package (if not move into default)

Select file run as ->Run Configurations->arguments paste file name e,g 2600.txt

->Apply->Run



The below result should appear on console



Thanks