**Combiner:**

1. save as CombinerClass.java
2. mkdir -m 755 cust\_combiner

* Creates a directory named cust\_combiner with read, write, and execute permissions for the owner, and read and execute permissions for others. This directory will hold the compiled class files.

1. javac -classpath $(hadoop classpath) -d cust\_combiner CombinerClass.java

* Compiles CombinerClass.java using Hadoop's classpath and places the resulting .class files into the cust\_combiner directory.

1. jar -cvf ${HOME}/scripts/custcombiner.jar -C cust\_combiner/ .

* Creates a JAR file named custcombiner.jar containing the compiled classes from the cust\_combiner directory.

1. hadoop jar ${HOME}/scripts/custcombiner.jar CombinerClass /volcheck /combiner.res1

* Executes the MapReduce job using the JAR file. CombinerClass is the name of the main class to run, /volcheck is the input path on HDFS, and /combiner.res1 is the output path on HDFS.

‘’’ Map(breask into K,V) -> shuffle/sort (all K,Vs) -> Reduce (aggregate on entire data)

Map(breask into K,V) -> Combiner (aggregate each Map's data) -> \

shuffle/sort (all K,Vs) -> Reduce (aggregate )

scenario 1:

M1 -> (aa,1),(bb,1),(aa,1)

M2 -> (aa,1), (bb,1),(cc,1)

R -> (aa,3), (bb,2),(cc,1)

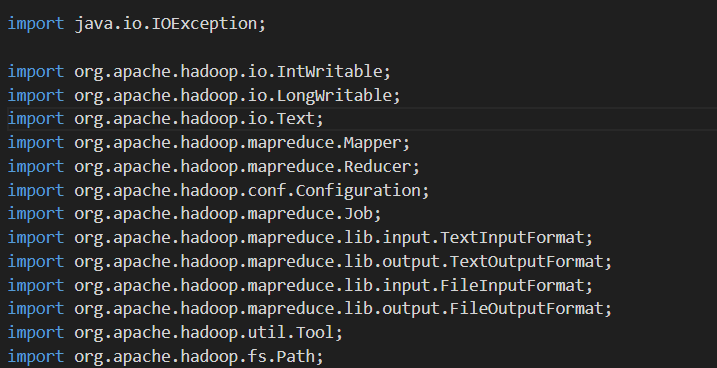
scenario 2:

M1 -> (aa,1),(bb,1),(aa,1) -> C1 -> (aa,2),(bb,1)

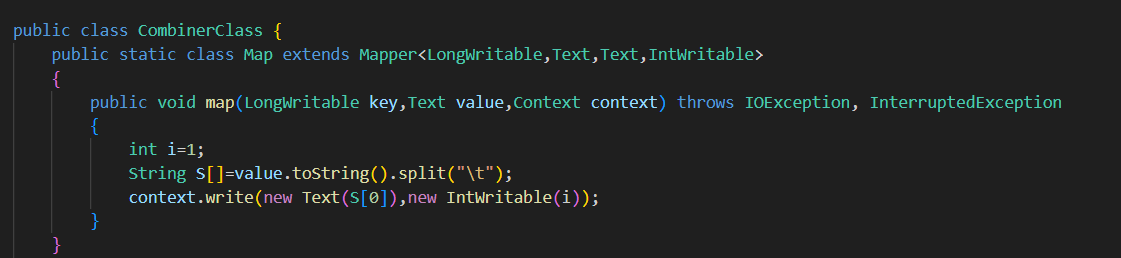
M2 -> (aa,1), (bb,1),(cc,1)-> C2 -> (aa,1), (bb,1),(cc,1)

R -> (aa,3), (bb,2),(cc,1) ‘’’

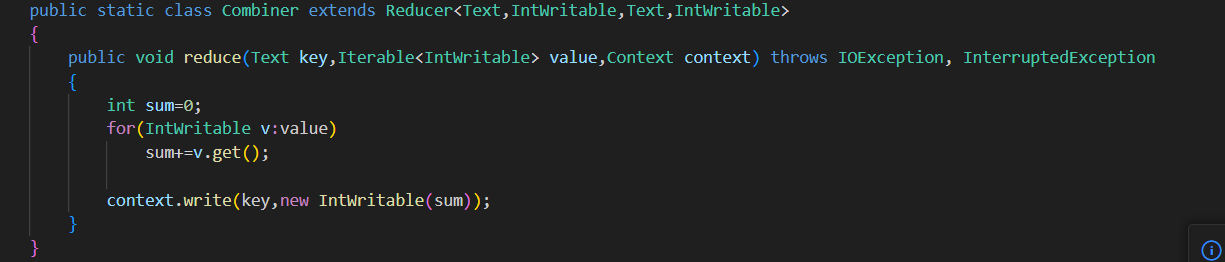
1. Save as CombinerClass.java



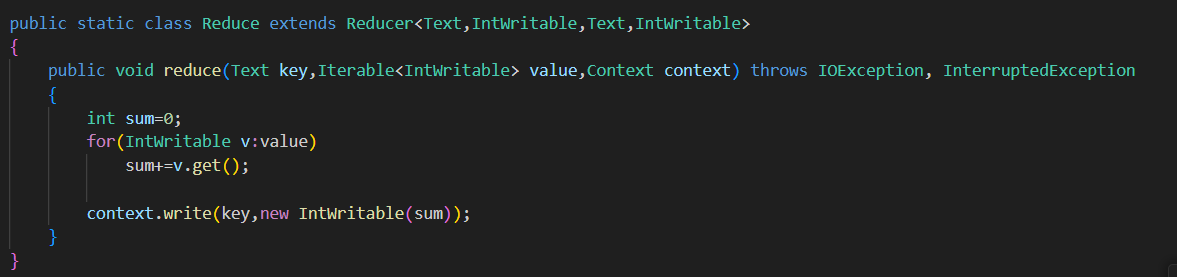
* These imports include Hadoop's classes for I/O operations, MapReduce framework, and configuration management.



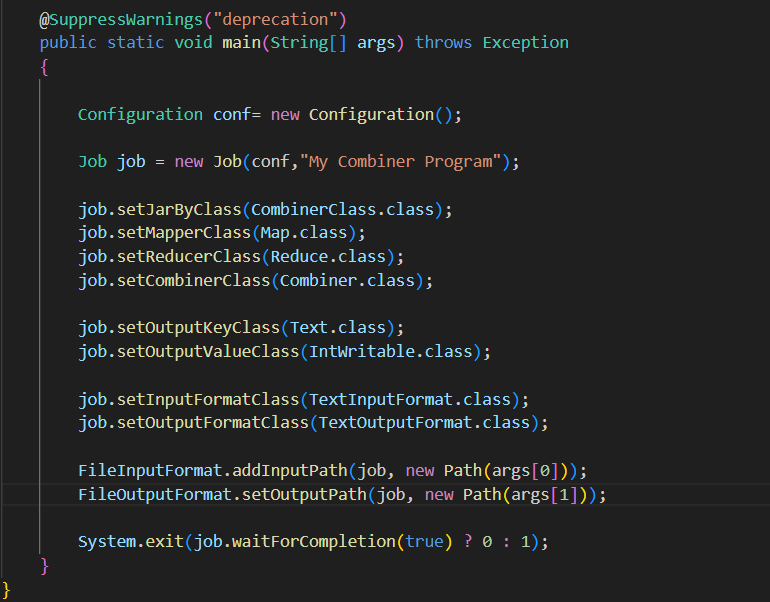
* This mapper reads each line, splits it by tab, and outputs a key-value pair where the key is the first part of the split string and the value is 1



* This combiner performs a local aggregation (summing up values) before the data is shuffled to the reducer. This helps in reducing the amount of data sent over the network.



* This reducer also sums up the values for each key, similar to the combiner, but operates on data from all mappers after the combiner step



* Set Job Name: "My Combiner Program".
* Set Mapper, Combiner, and Reducer Classes: Specifies which classes handle each phase of the MapReduce job.
* Specify Input/Output Paths: Uses command-line arguments to define paths.
* Exit: Exits with 0 if the job completes successfully, otherwise 1
* Implements a MapReduce job with a combiner to efficiently aggregate map outputs before reducing them.
* Compile the Java code, package it into a JAR, and run the MapReduce job on Hadoop. The combiner helps in reducing the volume of data shuffled between map and reduce phases, improving performance.