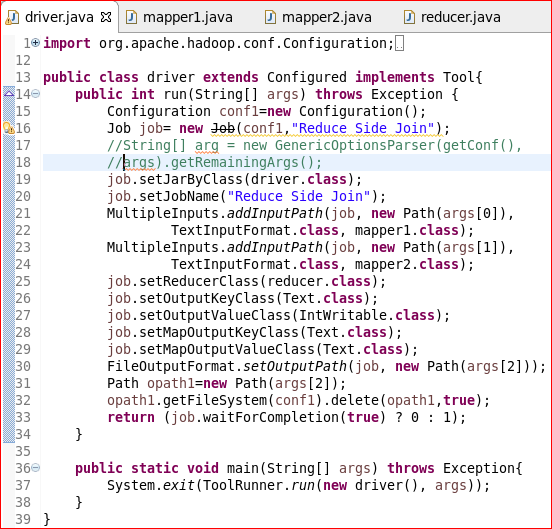
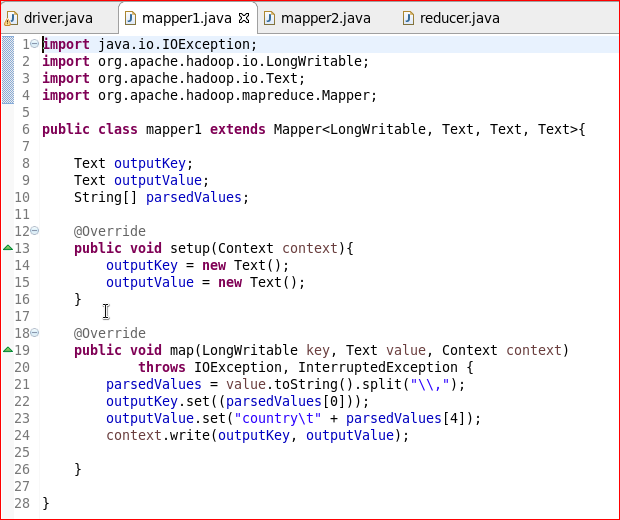
Problem statement:

Execute Session 16 - Assignment 1 (Machine and Sensor data analysis) using Reduce side join and compare which join is faster in normal conditions.

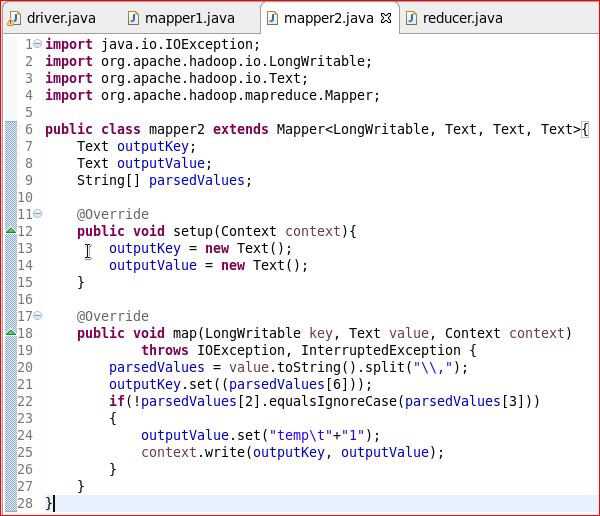
Task class:



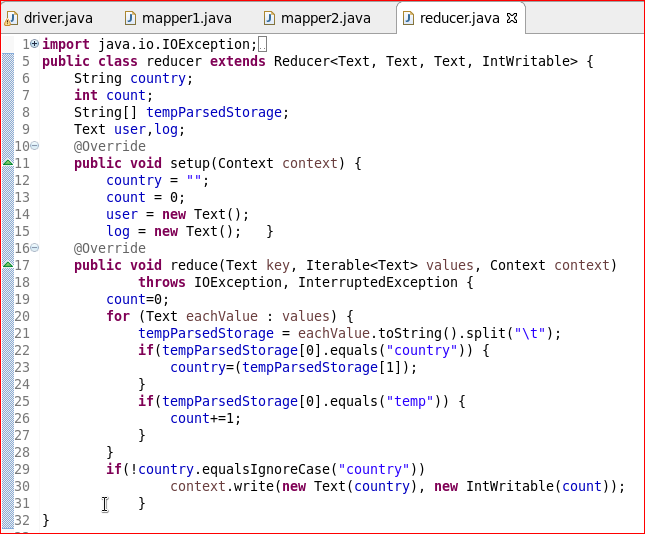
Mapper class 1:



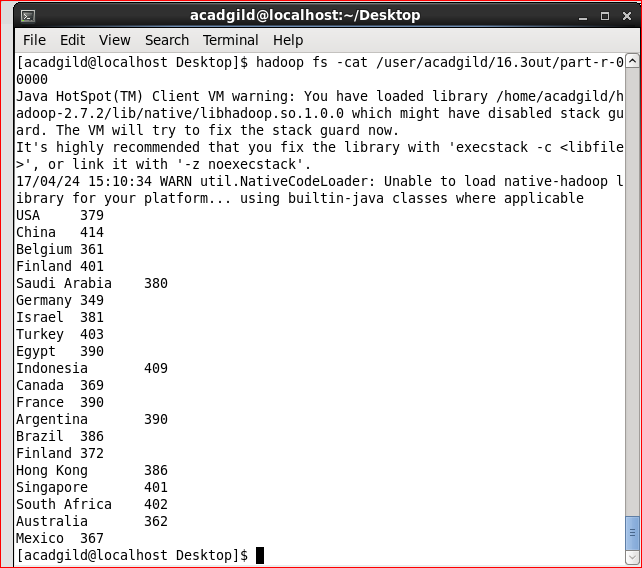
Mapper class 2:



Reducer class:



Output:



Comparison between joins:

**Map Side join**  
A map-side join between large inputs works by performing the join before the data reaches the map function. For this to work, though, the inputs to each map must be partitioned and sorted in a particular way. Each input data set must be divided into the same number of partitions, and it must be sorted by the same key (the join key) in each source. All the records for a particular key must reside in the same partition. This may sound like a strict requirement (and it is), but it actually fits the description of the output of a MapReduce job.

A map-side join can be used to join the outputs of several jobs that had the same number of reducers, the same keys, and output files that are not splittable which means the output files should not be bigger than the HDFS block size. Using the org.apache.hadoop.mapred.join.CompositeInputFormat class we can achieve this.

**Reduce Side join**  
Reduce-Side joins are simpler than Map-Side joins since the input datasets need not to be structured. But it is less efficient as both datasets have to go through the MapReduce shuffle phase. the records with the same key are brought together in the reducer. We can also use the Secondary Sort technique to control the order of the records.