Conrado Uraga
CS 437 Project 4- Map Reduce

For this project, we were to implement a map reduce to average the movies in a text file and output the results. Thus this was fairly straightforward. In the code, we have a map and a reducer. The map takes care of the parsing and maps the values with a key being the movieID and the rating being the value. In the map function, we must avoid the first line of the text file so we simply have a continue loop if it matches UserId, MovieId, etc and then an else statement that handles the parsing. In the reduce method, we take the key and iterate all the values. While iterating, we must take the sum of the values and have a counter that will keep track of how many values we iterated. We then take the average and write the collection as: key (movie id) and value (average of movie ratings).

I decided not to do the sorting implementation due to being busy, which is a shame since it seems interesting to do.

Analysis of the large file: I increased the clusters in a factor of 2. Although it took the same amount of time (2 minutes), the output files increased as well as the size of the file decreased. This is because we're splitting the jobs. However, when you increase the clusters to 16, then we have some empty files as well. The elapsed time is still the same (2 minutes), but it seems like having 16 clusters is overdoing it for this sort of job.

Which folder to look at: I gave you permission to access the fall-2014-437-mapreduce bucket. This bucket holds all the results and thus you should only need to look at this. "OutputResults" has the results of the tutorial. "SmallResults" has the output of the small Netflix file. "LargeResults" then have the results of the large file. These have multiple kinds (i.e. LargeResults_4, LargeResults_6, etc) since we had to run increase the size of the clusters.