Read Me Document

1) The versions of python , spark and scala are as below
   Python Version : 2.7 ,
   Spark version : 2.3.1 ,
   Scala : 2.11

2) Apriori algorithm has been implemented as a part of SON algorithm. The key concept of apriori is anti-monotonicity of the support measure. It assumes that
   - All subsets of a frequent itemset must be frequent
   - Similarly, for any infrequent itemset, all its supersets must be infrequent too
3) Support of a partition is calculated as support*(length of basket / total number of baskets)
4) After all the partitions are merged, the minimum support is checked again.
5) Problem 2:

Support Threshold	**Execution Time**
500	29.175962925 seconds
1000	7.65758991241 seconds

Problem 3:

Support Threshold	**Execution Time**
100000	397.576990843 seconds
120000	332.904181004 seconds

6) The need of using high threshold value is to reduce the number of computations. As the threshold value is high, the number of valid candidate set generally reduces and hence the computation for a larger set reduces. The bottleneck for my execution will be the case when all despite of large threshold value the number of candidate set is very high, that is the almost all members have count value greater than minimum support.
7) The command to execute the program is as below

   spark-submit parth_vaghani_SON.py "/input/file/path/yelp_reviews_small.txt" "<threshold value>" "/path/to/output/file/opsmall.txt"

   example:
   spark-submit final.py "/Users/parth/Desktop/USC/Data Mining/Assignment3/inf553_assignment3/Data/yelp_reviews_small.txt" "1000" "/Users/parth/Desktop/USC/Data Mining/Assignment3/opsmall.txt"