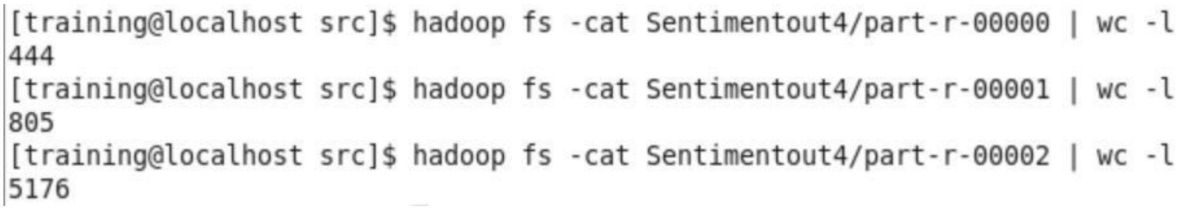
(Hao Zhang, WustlKey: h.zhang633, ID: 452003), (Hanming Li, WustlKey: lihanming, ID: 451802)

1. (a) The sumReducer can be used as a combiner because its operation is associated and commutative. But some reducer cannot be used as combiner, such as the reducer to get the variance of each key.
   (b) The code of add combiner is: job.setCombinerClass(SumReducer.class);
   The code of rename the job is: job.setJobName("Word Count Driver with Combiner");
   No, the result of operate this job with combiner to the test input is same as the original WordCount computation.
   (c) The total number of counters of Word Count Driver and Word Count Driver with Combiner are same, are both 299379.
   The number of bytes read, number of bytes written, map output records, reduce input groups are same for the Word Count Driver and the Count Driver with Combiner.
   Combine input records: 0 – 964453 = -964453
   Combine output records: 0 – 56268 = -56268
   Reduce input records: 964453 – 56268 = 908185.
   So 908185 key-value pairs are combined by the combiner.
   (d) Using my laptop, the CPU time spent by the original wordcount is 11890 while the one with the combiner is 12280. The physical memory used by the original wordcount is 980013056, by the one with combiner is 954490880.
   We can see that the time used by the wordcount with the combiner is even longer than the original one while the saved memory is not very big, so it is not a good idea to use the combiner here.
   But for some real big data, to do the mapreduce job with the mapper directly will take very huge memory, which will be difficult to be handle, for this kind of situations, the combiner is necessary, which will cost a little bit more time but save a lot of momery during the running of the job.

2. (c)



```
[training@localhost src]$ hadoop fs -cat Sentimentout4/part-r-00000 | wc -l
444
[training@localhost src]$ hadoop fs -cat Sentimentout4/part-r-00001 | wc -l
805
[training@localhost src]$ hadoop fs -cat Sentimentout4/part-r-00002 | wc -l
5176
```

As the output result shows, there are 444 positive words and 805 negative words used in the poems, so:
The sentiment score s = (444-805)/(444+805) = -0.289
The positive score p = 444/(444+805) = 0.355
Because the sentiment score is minus, so we can tell that the Shakespeare's poems suggest negative emotion.

(d) This is not a good way to measure the emotion in the poems, because in the real sentences, there are many phrase and fixed matches of words, so it cannot show the real meaning that the author want to express to just check the sentiment of each single words. And some words have many meanings, they may express different emotions when used at different places, so it is not so convincing to simply divide the words into positive and negative words.

So I think the mapper should can identify some phrases that have special emotions instead of just use the single words as keys. And the words should be judged according to the words in front of it and after it first, then determine its sentiment.