1. What are the top 20 phrases (sorted by total score) from the full data set only (again, with the 1960s as the foreground corpus this time). (5 points)

```
may be 0.008838196822560993
new york 0.008817980473762197
have been 0.00879834392608166
has been 0.007900288977719905
united states 0.007760735964659167
had been 0.007515958432386998
can be 0.00697532200759655
per cent 0.006740140326731554
should be 0.005310057922622394
would be 0.00514195257354407
did not 0.005113570149586299
will be 0.005057954394069061
he had 0.004996219855868485
must be 0.0045970677369172016
does not 0.004385124599823419
more than 0.004347771887133784
they are 0.004048348563741787
they were 0.003712297896963449
we have 0.0035556577286179445
at least 0.003509615106198601
```

2. Write down the test statistic for phraseness and informativeness based on Binomial Likelihood Ratio Test (BLRT). (5 points)

The test statistic**:**

$$BLRT(n_1,k_1,n_2,k_2) = 2\log\frac{L(p_1,k_1,n_1)L(p_2,k_2,n_2)}{L(p,k_1,n_1)L(p,k_2,n_2)}$$

Where

- $p_i = k_i / n_i$, $p = (k_1+k_2)/(n_1+n_2)$,
- $L(p,k,n) = p^k(1-p)^{n-k}$

For phraseness, the definition of k1,n1,k2,n2 is :

		comment
k_1	$C(W_1=x \wedge W_2=y)$	how often bigram *x y* occurs in corpus C
n_1	$C(W_1=x)$	how often word *x* occurs in corpus C
k_2	$C(W_1\neq x \wedge W_2=y)$	how often *y* occurs in C after a non-*x*

n_2	$C(W_1 \neq x)$	how often a non-*x* occurs in C

For informativeness:

		comment
k_1	$C(W_1=x \wedge W_2=y)$	how often bigram *x y* occurs in corpus C
n_1	$C(W_1=* \wedge W_2=*)$	how many bigrams in corpus C
k_2	$B(W_1=x \wedge W_2=y)$	how often *x y* occurs in **background corpus**
n_2	$B(W_1=* \wedge W_2=*)$	how many bigrams in background corpus

3. Using the test statistic from previous question, write down the hypothesis test (i.e. a condition involving the test statistic) for determining phraseness and informativeness. (5 points)
When define λ is:

$$\lambda = 2\log\frac{L(p_1,k_1,n_1)L(p_2,k_2,n_2)}{L(p,k_1,n_1)L(p,k_2,n_2)}$$

There is a c, when λ<=c, we reject the null hypothesis.
where null hypothesis H0: x and y are independent.
And H1: X and Y are from one distribution.

4. Provide a toy example where BLRT-based informativeness .
p1= k1/n1 (foreground)
p2= k2/n1 (background)

BLRT-based informativeness = (p1*p2) / (k1+k2)/(n1+n2)

(k1+k2)/(n1+n2) is fixed. So when p1 is small and p2 is large. BLRT-based informativeness will still give high score. while
KL-divergence based informativeness = p1 log p1/p2 , is the distance between p1 and p2. When p1 is small and p2 is large, their distance will be large,thus will get low score.
In this situateion, BLRT-based informativeness will have worse performance.

5,

Explain how you would calculate p(wl ˆθ) from p(wl ˆθdi) using only map and reduce steps. (5 points)

Suppose that we can store the queries in memory.
Map: we output the words that in the query and its probability. key=word, value=probabililty.
Reduce: For the same key, we sum up the log probability of the given key.

And for the number of files, we can just input as a parameter.

6. Since the above task can be performed using only map and reduce steps, do you think it is a good candidate for a Hadoop implementation? Why or why not? (5 points)
Yes, it is a good candidate.

7,

7. Under what precise conditions on a reduce function can it also be used as a combiner? (5 points)
When the reduce function is both commutative and associative, such as addition function. The word count is a good example of using a reduce function as combiner. However, when the conditions cannot be satisfied, a reduce function cannot be used as a combiner.

For example, the mean function. Suppose there are 5 <key,value> pairs from a mapper for

key k: <k,2>,<k,8>, <k,20>, <k,30>, <k,40>. The reduce will receive <k,{2,8,20,30,40 }, and the mean will be 20.

Now, if a combiner were used, and it will be applied on sets (<k,2>, <k,8>) and (<k,20>, <k,30>, <k,40>), then the

reducer would have received <k,{30,5}> and the output would have been different (17.5) ,which is an wrong.

8,

. Explain how you can use org.apache.hadoop.mapreduce.lib.jobcontrol.JobControl class to run multiple Hadoop MapReduce jobs with dependencies between them. Here dependency between a pair of jobs means one job cannot start until the other has finished. As an example for your explanation, consider MapReduce-based Naive Bayes training and testing to be the pair of dependent jobs. (5 points)

Solution:

Job1: MapReduce-based Naive Bayes training

Job2: MapReduce-based Naive Bayes testing

Job2 only can start after job1 finished. Following is the routine about how to use JobControl to control this dependency:

```
JobConf conf = new JobConf(MODEL.class);
//Create job1 and add it to a controller
Job job1=new Job(conf,"Job1");
ControlledJob ctrljob1=new  ControlledJob(conf);
ctrljob1.setJob(job1);
//Create job2 and add it to a controller
Job job2=new Job(conf,"Job2");
ControlledJob ctrljob2=new ControlledJob(conf);
ctrljob2.setJob(job2);

//Set the dependency between the two jobs.job2 must wait for job1 to finish
ctrljob2.addDependingJob(ctrljob1);
```

```
//Create a main controller to control those two jobs.
JobControl jobCtrl=new JobControl("myctrl");
// Add two jobs to main controller
jobCtrl.addJob(ctrljob1);
jobCtrl.addJob(ctrljob2);


Thread  t=new Thread(jobCtrl);
t.start();

while(true){

if(jobCtrl.allFinished()){
System.out.println(jobCtrl.getSuccessfulJobList());
jobCtrl.stop();
break;

}
```