Part 1:
Here's what did to

- I have a macbook, so I downloaded virtualbox mac version from here https://www.virtualbox.org/wiki/Downloads
- Downloaded and extracted Couldera CDH virtualbox image from http://www.cloudera.com/downloads/quickstart_vms/5-8.html it has hadoop v2.6.0
- Installed virtualbox and then used file > import appliance and pointed out to the CDH image file i extracted in the previous step. My machine has 8GB of ram, so i choose to give the virtual machine 4GB so that i can still work on my host machine if needed.
- Once the gui of the CDH VM opened, I logged in using username: cloudera password: cloudera.
- Opening web browser, there was a "getting started" tutorial (local url: http://quickstart.cloudera/) that i really enjoyed following, it shows me many components of the CDH and how they all work together to achieve several examples there. (Hue Web UI to quickly access everything, Sqoop for sql-to-hadoop feeding, Impala which is a SQL-like hadoop query engine, hive db table-structure over hadoop, Spark, Flume: real time ingestion framework)
- I googled for "hadoop wordcount example", then followed all the steps to setup eclipse, i also referred to the document attached in sakai "Hadoop setup documentation" for the "eclipse project setup" section, since everything else was already taken care of by CDH. it is worth mentioning that jar files paths mentioned there are different in CDH case, In CDH i found all needed jars inside /usr/lib/hadoop and /usr/lib/hadoop/lib
- I followed the setup documentation for exporting the JAR file. I ran into a problem where other subclasses defined inside the main class are not being found. Doing a google search, this article helped: http://stackoverflow.com/questions/21373550/class-not-found-exception-in-mapreduce-wordcount-job

  And the solution was to add
  job.setJarByClass(WordCountMaven.class);
  In the main class that submits the job.
- I wanted to test it on a big text file, so i head out to http://www.gutenberg.org/ downloaded a free book in text format.
- Moved it over to the virtual machine via "Shared folder" functionality of virtualbox.
- Copied that book to hadoop file system via (as CDH intro taught me):
  sudo -u hdfs hadoop fs -mkdir /user/hive/warehouse/2701
  sudo -u hdfs hadoop fs -copyFromLocal /home/cloudera/2701.txt.utf-8 /user/hive/warehouse/2701/
- Executed the MapReduce job using the following command:
  sudo hadoop jar wc2.jar WordCount /user/hive/warehouse/2701 /user/hive/warehouse/2701_wc
- Verified the results via:
  hadoop fs -ls /user/hive/warehouse/2701_wc # which lists files inside that folder

And cat the actual resulted contents via:
hadoop fs -cat /user/hive/warehouse/2701_wc/part-r-00000 | less

- Everything was working fine, but then I wanted to use maven so my life would be easier in such a way that i don't have to manually modifying eclipse build path whenever i need to add a new dependency, as well as find a quicker way to build jar files. So i followed this link:

  http://blog.cloudera.com/blog/2012/08/developing-cdh-applications-with-maven-and-eclipse/
- Now i can run unit tests, and generate a jar file using one simple command:
  mvn clean verify

Q2. Predict Product using Pairs method:

1. Java code (attached)
2. Show input, output and batch file to execute your program at command line in Hadoop.

<table>
<tr><td>input</td><td>Two files:
Cust1.txt:
B12 C31 D76 A12 B76 B12 D76 C31 A10

Cust2.txt:
B12 D76
C31 D76 B12 A12 C31 D76 B12
A12 D76 A12 D76</td><td>Copy to hfs:
sudo -u hdfs hadoop fs -copyFromLocal cust*.txt /user/hive/warehouse/custHistData/</td></tr>
<tr><td>Run command line:</td><td colspan="2">• mvn clean package
• sudo hadoop jar target/RelativeFreqPairs-1.0-SNAPSHOT.jar edu.mum.bigdata.mo.RelativeFreqPairsDriver /user/hive/warehouse/custHistData /user/hive/warehouse/chd_pairs/

OR just use the following bash script start.sh</td></tr>
<tr><td>Bash Script</td><td colspan="2">#!/bin/bash

JAR_FILE="target/RelativeFreqPairs-1.0-SNAPSHOT.jar"
CLASS_NAME="edu.mum.bigdata.mo.RelativeFreqPairsDriver"
HADOOP_OUTPUT_FOLDER="/user/hive/warehouse/chd_pairs_`date +'%Y%m%d%H%M%S'`"

INPUT_FILES="cust*.txt"
HADOOP_INPUT_DEST="/user/hive/warehouse/custHistData"

mvn clean package
sudo -u hdfs hadoop fs -copyFromLocal $INPUT_FILES $HADOOP_INPUT_DEST
sudo hadoop jar $JAR_FILE $CLASS_NAME $HADOOP_INPUT_DEST $HADOOP_OUTPUT_FOLDER

echo "Output written to $HADOOP_OUTPUT_FOLDER"
echo "Output was: "
hadoop fs -cat $HADOOP_OUTPUT_FOLDER/part-r-00000</td></tr>
</table>

output:	hadoop fs -cat /user/hive/warehouse/chd_pairs/part-r-00000 | less

Q3. Predict Product using Stripes method:

1. Java code (attached)
2. Show input, output and batch file to execute your program at command line in Hadoop.

input	Two files: Cust1.txt: B12 C31 D76 A12 B76 B12 D76 C31 A10 Cust2.txt: B12 D76 C31 D76 B12 A12 C31 D76 B12 A12 D76 A12 D76	Copy to hfs: sudo -u hdfs hadoop fs -copyFromLocal cust*.txt /user/hive/warehouse/custHistData/
Run command line:	• mvn clean package • sudo hadoop jar target/RelativeFreqStripes-1.0-SNAPSHOT.jar edu.mum.bigdata.mo.RelativeFreqStripesDriver /user/hive/warehouse/custHistData /user/hive/warehouse/chd_stripes/	
Bash Script	#!/bin/bash JAR_FILE="target/RelativeFreqStripes-1.0-SNAPSHOT.jar" CLASS_NAME="edu.mum.bigdata.mo.RelativeFreqStripesDriver" HADOOP_OUTPUT_FOLDER="/user/hive/warehouse/chd_stripes_`date +'%Y%m%d%H%M%S'`" INPUT_FILES="cust*.txt" HADOOP_INPUT_DEST="/user/hive/warehouse/custHistData" mvn clean package sudo -u hdfs hadoop fs -copyFromLocal $INPUT_FILES $HADOOP_INPUT_DEST	

	sudo hadoop jar $JAR_FILE $CLASS_NAME $HADOOP_INPUT_DEST $HADOOP_OUTPUT_FOLDER echo "Output written to $HADOOP_OUTPUT_FOLDER" echo "Output was: " hadoop fs -cat $HADOOP_OUTPUT_FOLDER/part-r-00000
output:	hadoop fs -cat /user/hive/warehouse/chd_pairs/part-r-00000 | less

Q4. Predict Product using Hybrid method:

3. Java code (attached)
4. Show input, output and batch file to execute your program at command line in Hadoop.

input	Two files: Cust1.txt: B12 C31 D76 A12 B76 B12 D76 C31 A10 Cust2.txt: B12 D76 C31 D76 B12 A12 C31 D76 B12 A12 D76 A12 D76	Copy to hfs: sudo -u hdfs hadoop fs -copyFromLocal cust*.txt /user/hive/warehouse/custHistData/
Run command line:	• mvn clean package • sudo hadoop jar target/RelativeFreqHybrid-1.0-SNAPSHOT.jar edu.mum.bigdata.mo.RelFreqHybridDriver /user/hive/warehouse/custHistData/ /user/hive/warehouse/chd_hybrid/	
Bash script	#!/bin/bash JAR_FILE="target/RelativeFreqHybrid-1.0-SNAPSHOT.jar" CLASS_NAME="edu.mum.bigdata.mo.RelFreqHybridDriver" HADOOP_OUTPUT_FOLDER="/user/hive/warehouse/chd_hybrid_`date +'%Y%m%d%H%M%S'`"	

	INPUT_FILES="cust*.txt" HADOOP_INPUT_DEST="/user/hive/warehouse/custHistData" mvn clean package sudo -u hdfs hadoop fs -copyFromLocal $INPUT_FILES $HADOOP_INPUT_DEST sudo hadoop jar $JAR_FILE $CLASS_NAME $HADOOP_INPUT_DEST $HADOOP_OUTPUT_FOLDER echo "Output written to $HADOOP_OUTPUT_FOLDER" echo "Output was: " hadoop fs -cat $HADOOP_OUTPUT_FOLDER/part-r-00000
output:	hadoop fs -cat /user/hive/warehouse/chd_hybrid/part-r-00000 | less