**Part 1 (Inverted Index) and Part 2 (Boolean Query)**

It is implemented in the directory: /home/cloudera/Assignment2

There are a few .txt files already present for testing. Please remove the .txt files before you test the code. The program will read any .txt file present in /home/cloudera/Assignment2 directory as input.

To run the code, go to /home/cloudera/Assignment2 directory.

1. Add test .txt files in this path.
2. javac \*.java
3. java InvertedIndex

Note: to perform and operation, use &

to perform or operation, use |

**Part 3 (MapReduce)**

It is implemented in the directory: /home/cloudera/Assignment2MapReduce

The input files are .txt. They must be placed in the hdfs directory named input. The code will read all .txt files present in this directory.

hadoop fs -ls input

Some .txt files are already present for testing. Please remove them before you test.

InvertedIndex.java is used for MapReduce

Test.java is used for Query

.class files of InvertedIndex need to be created inside the directory

/home/cloudera/Assignment2MapReduce/invertedClasses

To run InvertedIndex.java, follow the steps:

1. Navigate to the directory /home/cloudera/Assignment2MapReduce
2. javac -classpath hadoop-core-1.2.1.jar -d invertedClasses InvertedIndex.java

This will create .class files inside invertedClasses directory

1. jar -cvf invertedIndex.jar -C invertedClasses/ .

This will create a JAR called invertedIndex.jar

1. hadoop jar invertedIndex.jar InvertedIndex /user/cloudera/input /user/cloudera/output

Output will be present in directory /user/cloudera/output

1. hadoop fs -cat output/part-r-00000

Output is present in the above file.

.class files of Test need to be created inside the directory

/home/cloudera/Assignment2MapReduce/classes

To run Test.java, follow the steps:

1. Navigate to the directory /home/cloudera/Assignment2MapReduce
2. javac -classpath hadoop-core-1.2.1.jar -d classes Test.java

This will create .class inside /home/cloudera/Assignment2MapReduce/classes

1. jar -cvf test.jar -C classes/ .

Create JAR called test.jar

1. Run the JAR created using

hadoop jar test.jar Test

Note: File part-r-00000 needs to be present in hdfs output directory, else exception will occur.

Big Report: If any query returns an empty search result from Test.java (Part 3), successive queries made will produce wrong output. To stop this, I am terminating the code whenever a query returns empty result. Did not have enough time to solve this. However, the code should produce correct results, only inconvenience is that the Test program needs to be restarted.

Query implemented in Part 2 does not suffer from this.