Assignment 4

In this Exercise, we have included DistributedCache class to import a file containing words to be skipped while calculating their word count. Also counters are used to report corresponding type of words through the use of ENUM.

EnhancedWordCount Class

```
import java.io.*;
import java.util.*;

import org.apache.hadoop.fs.Path;
import org.apache.hadoop.filecache.DistributedCache;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
```

Enum to keep track counter of words starting with a digit/letter and all words counted

```
enum NatureofWords { STARTS_WITH_DIGIT, STARTS_WITH_LETTER, ALL }

public class EnhancedWordCount extends Configured {
  public static class Map extends MapReduceBase implements Mapper<LongWritable, Text, Text, IntWritable> {
        private final static IntWritable one = new IntWritable(1);
        private Text word = new Text();
```

The Default value of casesensitive Boolean. If not specifically specified through command line, this program differentiates between upper case and lower case letters .

```
        private boolean caseSensitive = true;
        private Set<String> patternsToSkip = new HashSet<String>();

        private long numRecords = 0;
        private String inputFile;
        private BufferedReader fis;

        public void configure(JobConf job) {
                caseSensitive = job.getBoolean("wordcount.case.sensitive", true);
                inputFile = job.get("map.input.file");

                        if (job.getBoolean("wordcount.skip.patterns", false)) {
                                Path[] patternsFiles = new Path[0];
                                try {
                                        patternsFiles = DistributedCache.getLocalCacheFiles(job);
                                } catch (IOException ioe) {
                                    System.err.println("Caught exception getting cached files: " + ioe.toString());
                                }
                        for (Path patternsFile : patternsFiles) {
                                 parseSkipFile(patternsFile);
                                 }}
```

We will be mentioning the Boolean value of wordcount.skip.patterns in command line through string “-skip”.

```
         private void parseSkipFile(Path patternsFile) {
                try {
                        fis = new BufferedReader(new FileReader(patternsFile.toString()));
                        String pattern = null;
                        while ((pattern = fis.readLine()) != null) {
                         patternsToSkip.add(pattern);}
                } catch (IOException ioe) {
                 System.err.println("Caught exception parsing the cached file '"+patternsFile+"':"+ ioe.toString());
  }}

  public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output, Reporter reporter)
  throws IOException {
         String line = (caseSensitive) ? value.toString() : value.toString().toLowerCase();
        StringTokenizer tokenizer = new StringTokenizer(line);
        while (tokenizer.hasMoreTokens()) {
                String token = tokenizer.nextToken();
                if (patternsToSkip.contains(token))
                        System.out.println("Word Skipped");
                else{
                        word.set(token);
            output.collect(word, one);
          }
    //reporter.incrCounter(Counters.INPUT_WORDS, 1);
  }}}

 public static class Reduce extends MapReduceBase implements Reducer<Text, IntWritable, Text, IntWritable> {
        public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text, IntWritable> output,
Reporter reporter) throws IOException {
                int sum = 0;

                String token = key.toString();
                if (StringUtils.startsWithDigit(token)){
                        reporter.incrCounter(NatureofWords.STARTS_WITH_DIGIT, 1);
  }
                else if (StringUtils.startsWithLetter(token)){
                        reporter.incrCounter(NatureofWords.STARTS_WITH_LETTER, 1);
  }
                reporter.incrCounter(NatureofWords.ALL, 1);

                while (values.hasNext()) {
                        sum += values.next().get();
  }
                output.collect(key, new IntWritable(sum));
 }
}

 public static void main (String[] args) throws Exception {
        JobConf conf = new JobConf(EnhancedWordCount.class);
        conf.setJobName("enhancedwordcount");
```

Paths entered in DistibutedCache are taken out. These paths are fed in parseSkipFile(). This function reads individual words from the files referred by extracted paths and added to patternsToSkip array.

1) If Boolean "caseSensitive" is true, string value is taken as it is,if false, string value is taken as lower case.
2) If words extracted are present in patternsToSkip array, they are not sent to coutput.collect

Counters for defined Enum are increased from output of StringUtil class defined.

```
conf.setOutputKeyClass(Text.class);
conf.setOutputValueClass(IntWritable.class);
conf.setMapperClass(Map.class);
conf.setCombinerClass(Reduce.class);
conf.setReducerClass(Reduce.class);

conf.setInputFormat(TextInputFormat.class);
conf.setOutputFormat(TextOutputFormat.class);

List<String> other_args = new ArrayList<String>();
        for (int i=0; i < args.length; ++i) {
                if ("-skip".equals(args[i])) {
                        DistributedCache.addCacheFile(new Path(args[++i]).toUri(), conf);
                        conf.setBoolean("wordcount.skip.patterns", true);
                }else if("-case".equals(args[i])){
                        conf.setBoolean("wordcount.case.sensitive", false );}
                        {
                                other_args.add(args[i]);}}

FileInputFormat.setInputPaths(conf, new Path(other_args.get(0)));
FileOutputFormat.setOutputPath(conf, new Path(other_args.get(1)));

JobClient.runJob(conf);}}
```

The above for loop takes inputs as command line arguments

1) If string "-skip" is encountered, next string of the command line is taken as address of the file which contains words to be skipped. The file associated with this string is added to DistributedCache class.
2) If string "-case" is encountered, the Boolean of wordcount.case.sensitive is set as false and program doesnot differentiates between upper and lower case letters.

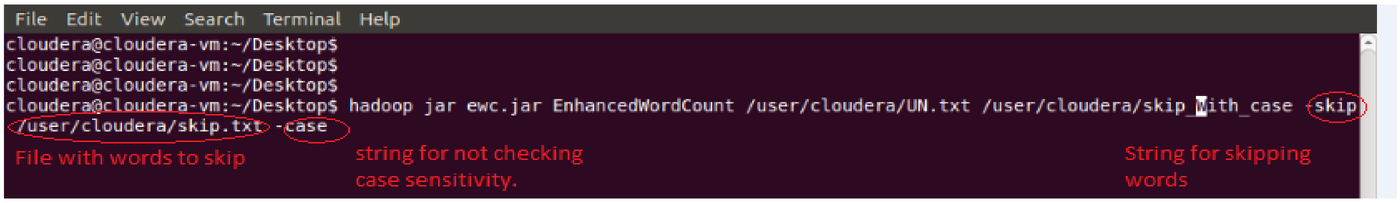
StringUtil Class

```
public class StringUtils {
        public static boolean startsWithDigit(String s){
                if( s == null || s.length() == 0 )
                        return false;

                return Character.isDigit(s.charAt(0));}

        public static boolean startsWithLetter(String s){
                if( s == null || s.length() == 0 )
```

This Class defines two functions startsWithDigit() and StartsWithLetter() which return Boolean.

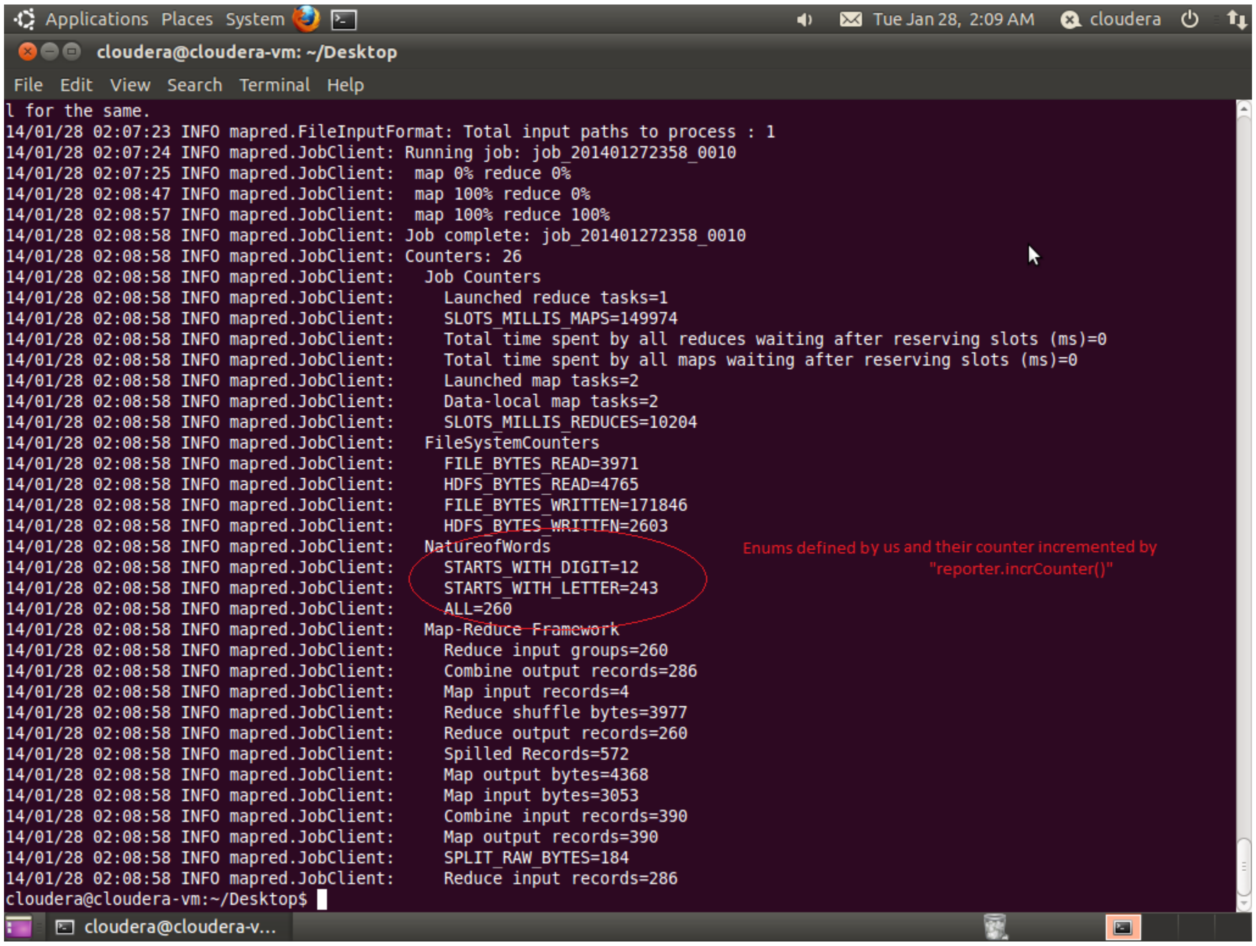
```
                        return false;

                return Character.isLetter(s.charAt(0));
        }}
```

Running the Program:



```
cloudera@cloudera-vm:~/Desktop$
cloudera@cloudera-vm:~/Desktop$
cloudera@cloudera-vm:~/Desktop$
cloudera@cloudera-vm:~/Desktop$ hadoop jar ewc.jar EnhancedWordCount /user/cloudera/UN.txt /user/cloudera/skip_with_case -skip
/user/cloudera/skip.txt -case
```

File with words to skip
string for not checking case sensitivity.
String for skipping words



```
l for the same.
14/01/28 02:07:23 INFO mapred.FileInputFormat: Total input paths to process : 1
14/01/28 02:07:24 INFO mapred.JobClient: Running job: job_201401272358_0010
14/01/28 02:07:25 INFO mapred.JobClient:  map 0% reduce 0%
14/01/28 02:08:47 INFO mapred.JobClient:  map 100% reduce 0%
14/01/28 02:08:57 INFO mapred.JobClient:  map 100% reduce 100%
14/01/28 02:08:58 INFO mapred.JobClient: Job complete: job_201401272358_0010
14/01/28 02:08:58 INFO mapred.JobClient: Counters: 26
14/01/28 02:08:58 INFO mapred.JobClient:   Job Counters
14/01/28 02:08:58 INFO mapred.JobClient:     Launched reduce tasks=1
14/01/28 02:08:58 INFO mapred.JobClient:     SLOTS_MILLIS_MAPS=149974
14/01/28 02:08:58 INFO mapred.JobClient:     Total time spent by all reduces waiting after reserving slots (ms)=0
14/01/28 02:08:58 INFO mapred.JobClient:     Total time spent by all maps waiting after reserving slots (ms)=0
14/01/28 02:08:58 INFO mapred.JobClient:     Launched map tasks=2
14/01/28 02:08:58 INFO mapred.JobClient:     Data-local map tasks=2
14/01/28 02:08:58 INFO mapred.JobClient:     SLOTS_MILLIS_REDUCES=10204
14/01/28 02:08:58 INFO mapred.JobClient:   FileSystemCounters
14/01/28 02:08:58 INFO mapred.JobClient:     FILE_BYTES_READ=3971
14/01/28 02:08:58 INFO mapred.JobClient:     HDFS_BYTES_READ=4765
14/01/28 02:08:58 INFO mapred.JobClient:     FILE_BYTES_WRITTEN=171846
14/01/28 02:08:58 INFO mapred.JobClient:     HDFS_BYTES_WRITTEN=2603
14/01/28 02:08:58 INFO mapred.JobClient:   NatureofWords
14/01/28 02:08:58 INFO mapred.JobClient:     STARTS_WITH_DIGIT=12
14/01/28 02:08:58 INFO mapred.JobClient:     STARTS_WITH_LETTER=243
14/01/28 02:08:58 INFO mapred.JobClient:     ALL=260
14/01/28 02:08:58 INFO mapred.JobClient:   Map-Reduce Framework
14/01/28 02:08:58 INFO mapred.JobClient:     Reduce input groups=260
14/01/28 02:08:58 INFO mapred.JobClient:     Combine output records=286
14/01/28 02:08:58 INFO mapred.JobClient:     Map input records=4
14/01/28 02:08:58 INFO mapred.JobClient:     Reduce shuffle bytes=3977
14/01/28 02:08:58 INFO mapred.JobClient:     Reduce output records=260
14/01/28 02:08:58 INFO mapred.JobClient:     Spilled Records=572
14/01/28 02:08:58 INFO mapred.JobClient:     Map output bytes=4368
14/01/28 02:08:58 INFO mapred.JobClient:     Map input bytes=3053
14/01/28 02:08:58 INFO mapred.JobClient:     Combine input records=390
14/01/28 02:08:58 INFO mapred.JobClient:     Map output records=390
14/01/28 02:08:58 INFO mapred.JobClient:     SPLIT_RAW_BYTES=184
14/01/28 02:08:58 INFO mapred.JobClient:     Reduce input records=286
cloudera@cloudera-vm:~/Desktop$
```

Applications Places System
Tue Jan 28, 2:09 AM
cloudera
cloudera@cloudera-vm: ~/Desktop
File Edit View Search Terminal Help
Enums defined by us and their counter incremented by "reporter.incrCounter()"
cloudera@cloudera-v...

If you do want upper and lowercase letters to be counted separately, do not type "-case" in command line. Similarly do not include "-skip" and corresponding file path if you do not want words to be skipped.