WC_Mapper.java

```
package org.kunal;

import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
public class WC_Mapper extends MapReduceBase implements
Mapper<LongWritable,Text,Text,IntWritable>{
    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();
    public void map(LongWritable key, Text value,OutputCollector<Text,IntWritable>
output,
                    Reporter reporter) throws IOException{
        String line = value.toString();
        StringTokenizer  tokenizer = new StringTokenizer(line);
        while (tokenizer.hasMoreTokens()){
            word.set(tokenizer.nextToken());
            output.collect(word, one);
        }
    }

}
```

WC_Reducer.java

```
package org.kunal;

import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reducer;
import org.apache.hadoop.mapred.Reporter;

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

WC_Runner.java

```
package org.kunal;

import java.io.IOException;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.mapred.TextInputFormat;
import org.apache.hadoop.mapred.TextOutputFormat;
public class WC_Runner {
    public static void main(String[] args) throws IOException{
        JobConf conf = new JobConf(WC_Runner.class);
        conf.setJobName("WordCount");
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(IntWritable.class);
        conf.setMapperClass(WC_Mapper.class);
        conf.setCombinerClass(WC_Reducer.class);
        conf.setReducerClass(WC_Reducer.class);
        conf.setInputFormat(TextInputFormat.class);
        conf.setOutputFormat(TextOutputFormat.class);
        FileInputFormat.setInputPaths(conf,new Path(args[0]));
        FileOutputFormat.setOutputPath(conf,new Path(args[1]));
        JobClient.runJob(conf);
    }
}
```

input.txt

```
The quick brown fox jumps over the lazy dog. The quick brown fox jumps over the lazy dog
again and again. Dog dog cat mouse bird fish snake lizard frog turtle rabbit. A long time
ago in a galaxy far far away. May the force be with you. To be or not to be that is the
question. One fish two fish red fish blue fish. Green eggs and ham. The cat in the hat.
Oh the places you'll go. When you go there you'll know.

Hi my name is Claude. Hello world hello universe. MapReduce is used for processing large
data sets with parallel distributed algorithms on clusters. The map step processes data
and the reduce step aggregates results. Hadoop is an open source software framework for
reliable scalable distributed computing. It provides a runtime for MapReduce algorithms
to work with large data sets.

This is some example text for a word counting application. This text contains many common
words and phrases. It includes book titles movie quotes and tech terminology. With this
text you can test counting the frequency of words. You can analyze which words appear
most often. You can explore the distribution of word lengths. There are many
possibilities for gaining insights from the word frequencies in this corpus of sample
text data. I've tried to make the text relatively varied and substantial in length to
allow for meaningful analysis and testing of your MapReduce word counting implementation
on Hadoop. Let me know if you need any other sample text generated!
```

```
Output

A          1
Claude.    1
Dog        1
Green      1
Hadoop     1
Hadoop.    1
Hello      1
Hi         1
I've       1
It         2
Let        1
MapReduce  3
May        1
Oh         1
One        1
The        4
There      1
This       2
To         1
When       1
With       1
You        2
a          3
again      1
again.     1
aggregates 1
ago        1
algorithms 2
allow      1
an         1
analysis   1
analyze    1
and        7
any        1
appear     1
application.          1
are        1
away.      1
be         3
bird       1
blue       1
book       1
brown      2
can        3
cat        2
clusters.  1
common     1
computing. 1
contains   1
corpus     1
counting   3
data       3
data.      1
distributed           2
distribution          1
dog        2
dog.       1
eggs       1
example    1
```

```
explore    1
far        2
fish       4
fish.      1
for        6
force      1
fox        2
framework  1
frequencies          1
frequency  1
frog       1
from       1
gaining    1
galaxy     1
generated! 1
go         1
go.        1
ham.       1
hat.       1
hello      1
if         1
implementation       1
in         4
includes   1
insights   1
is         5
jumps      2
know       1
know.      1
large      2
lazy       2
length     1
lengths.   1
lizard     1
long       1
make       1
many       2
map        1
me         1
meaningful 1
most       1
mouse      1
movie      1
my         1
name       1
need       1
not        1
of         4
often.     1
on         2
open       1
or         1
other      1
over       2
parallel   1
phrases.   1
places     1
possibilities        1
processes  1
processing 1
provides   1
```

```
question.  1
quick      2
quotes     1
rabbit.    1
red        1
reduce     1
relatively 1
reliable   1
results.   1
runtime    1
sample     2
scalable   1
sets       1
sets.      1
snake      1
software   1
some       1
source     1
step       2
substantial          1
tech       1
terminology.         1
test       1
testing    1
text       6
that       1
the        11
there      1
this       2
time       1
titles     1
to         4
tried      1
turtle     1
two        1
universe.  1
used       1
varied     1
which      1
with       3
word       4
words      2
words.     1
work       1
world      1
you        3
you'll     2
you.       1
your       1
```