ASSIGNMENT NO : 1

WordCount.java

```
package PackageDemo;
import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;

public class WordCount {

    public static void main(String[] args) throws Exception {
        Configuration c = new Configuration();
        String[] files = new GenericOptionsParser(c,
args).getRemainingArgs();
        Path input = new Path(files[0]);
        Path output = new Path(files[1]);
        Job j = new Job(c, "wordcount");
        j.setJarByClass(WordCount.class);
        j.setMapperClass(MapForWordCount.class);
        j.setReducerClass(ReduceForWordCount.class);
        j.setOutputKeyClass(Text.class);
        j.setOutputValueClass(IntWritable.class);
        FileInputFormat.addInputPath(j, input);
        FileOutputFormat.setOutputPath(j, output);
        System.exit(j.waitForCompletion(true) ? 0 : 1);
    }

    public static class MapForWordCount extends Mapper<LongWritable, Text,
Text, IntWritable> {

        public void map(LongWritable key, Text value, Context con) throws
IOException, InterruptedException {
            String line = value.toString();
            String[] words = line.split(",");
            for (String word : words) {
                Text outputKey = new Text(word.toUpperCase().trim());
                IntWritable outputValue = new IntWritable(1);
                con.write(outputKey, outputValue);
            }
        }
    }

    public static class ReduceForWordCount extends Reducer<Text,
IntWritable, Text, IntWritable> {

        public void reduce(Text word, Iterable<IntWritable> values, Context
con) throws IOException, InterruptedException {
            int sum = 0;
            for (IntWritable value : values) {
```

```
                sum += value.get();
            }
            con.write(word, new IntWritable(sum));
        }
    }
}
```

WCMapper.java

```
// Importing libraries
import java.io.IOException;
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 WCMapper extends MapReduceBase implements Mapper<LongWritable,
                                                                Text,
Text, IntWritable> {

      // Map function
      public void map(LongWritable key, Text value, OutputCollector<Text,
                      IntWritable> output, Reporter rep) throws IOException
      {

            String line = value.toString();

            // Splitting the line on spaces
            for (String word : line.split(" "))
            {
                  if (word.length() > 0)
                  {
                        output.collect(new Text(word), new IntWritable(1));
                  }
            }
      }
}
```

WCReducer.java

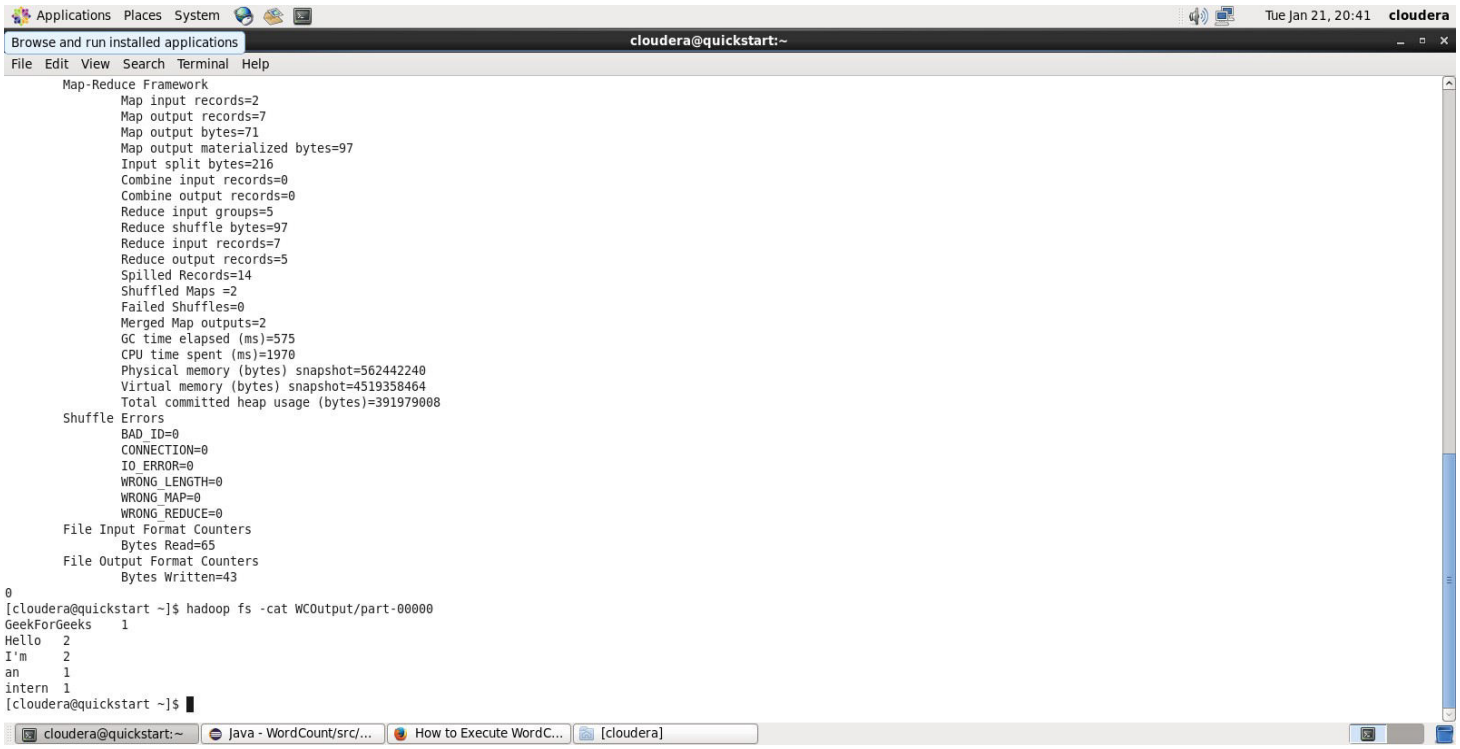
```
// Importing libraries
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reducer;
import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

public class WCReducer extends MapReduceBase implements Reducer<Text,
                                                    IntWritable, Text,
IntWritable> {

      // Reduce function
      public void reduce(Text key, Iterator<IntWritable> value,
                      OutputCollector<Text, IntWritable> output,
```

```
                                        Reporter rep) throws IOException
    {

        int count = 0;

        // Counting the frequency of each words
        while (value.hasNext())
        {
            IntWritable i = value.next();
            count += i.get();
        }

        output.collect(key, new IntWritable(count));
    }
}
```

WCDriver.java

```
// Importing libraries
import java.io.IOException;
import org.apache.hadoop.conf.Configured;
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.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class WCDriver extends Configured implements Tool {

    public int run(String args[]) throws IOException
    {
        if (args.length < 2)
        {
            System.out.println("Please give valid inputs");
            return -1;
        }

        JobConf conf = new JobConf(WCDriver.class);
        FileInputFormat.setInputPaths(conf, new Path(args[0]));
        FileOutputFormat.setOutputPath(conf, new Path(args[1]));
        conf.setMapperClass(WCMapper.class);
        conf.setReducerClass(WCReducer.class);
        conf.setMapOutputKeyClass(Text.class);
        conf.setMapOutputValueClass(IntWritable.class);
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(IntWritable.class);
        JobClient.runJob(conf);
        return 0;
    }

    public static void main(String args[]) throws Exception
    {
        int exitCode = ToolRunner.run(new WCDriver(), args);
        System.out.println(exitCode);
    }
}
```

OUTPUT


Applications Places System
Tue Jan 21, 20:41 cloudera
Browse and run installed applications
cloudera@quickstart:~
File Edit View Search Terminal Help
Map-Reduce Framework
Map input records=2
Map output records=7
Map output bytes=71
Map output materialized bytes=97
Input split bytes=216
Combine input records=0
Combine output records=0
Reduce input groups=5
Reduce shuffle bytes=97
Reduce input records=7
Reduce output records=5
Spilled Records=14
Shuffled Maps =2
Failed Shuffles=0
Merged Map outputs=2
GC time elapsed (ms)=575
CPU time spent (ms)=1970
Physical memory (bytes) snapshot=562442240
Virtual memory (bytes) snapshot=4519358464
Total committed heap usage (bytes)=391979008
Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0
File Input Format Counters
Bytes Read=65
File Output Format Counters
Bytes Written=43
0
[cloudera@quickstart ~]$ hadoop fs -cat WCOutput/part-00000
GeekForGeeks 1
Hello 2
I'm 2
an 1
intern 1
[cloudera@quickstart ~]$
cloudera@quickstart:~
Java - WordCount/src/...
How to Execute WordC...
[cloudera]