```
Import java.io.IOException;
Import java.util.StringTokenizer;

Import org.apache.hadoop.conf.Configuration;
Import org.apache.hadoop.fs.Path;
Import org.apache.hadoop.io.IntWritable;
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;

Public class WordCount {

 Public static class TokenizerMapper
     Extends Mapper<Object, Text, Text, IntWritable>{

  Private final static IntWritable one = new IntWritable(1);
  Private Text word = new Text();

  Public void map(Object key, Text value, Context context
            ) throws IOException, InterruptedException {
    StringTokenizer itr = new StringTokenizer(value.toString());
    While (itr.hasMoreTokens()) {
     Word.set(itr.nextToken());
```

```
    Context.write(word, one);
   }
  }
}

Public static class IntSumReducer
   Extends Reducer<Text,IntWritable,Text,IntWritable> {
  Private IntWritable result = new IntWritable();

  Public void reduce(Text key, Iterable<IntWritable> values,
                Context context
                ) throws IOException, InterruptedException {
   Int sum = 0;
   For (IntWritable val : values) {
    Sum += val.get();
   }
   Result.set(sum);
   Context.write(key, result);
  }
}

Public static void main(String[] args) throws Exception {
  Configuration conf = new Configuration();
  Job job = Job.getInstance(conf, “word count”);
  Job.setJarByClass(WordCount.class);
  Job.setMapperClass(TokenizerMapper.class);
```

```
    Job.setCombinerClass(IntSumReducer.class);
    Job.setReducerClass(IntSumReducer.class);
    Job.setOutputKeyClass(Text.class);
    Job.setOutputValueClass(IntWritable.class);
    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```