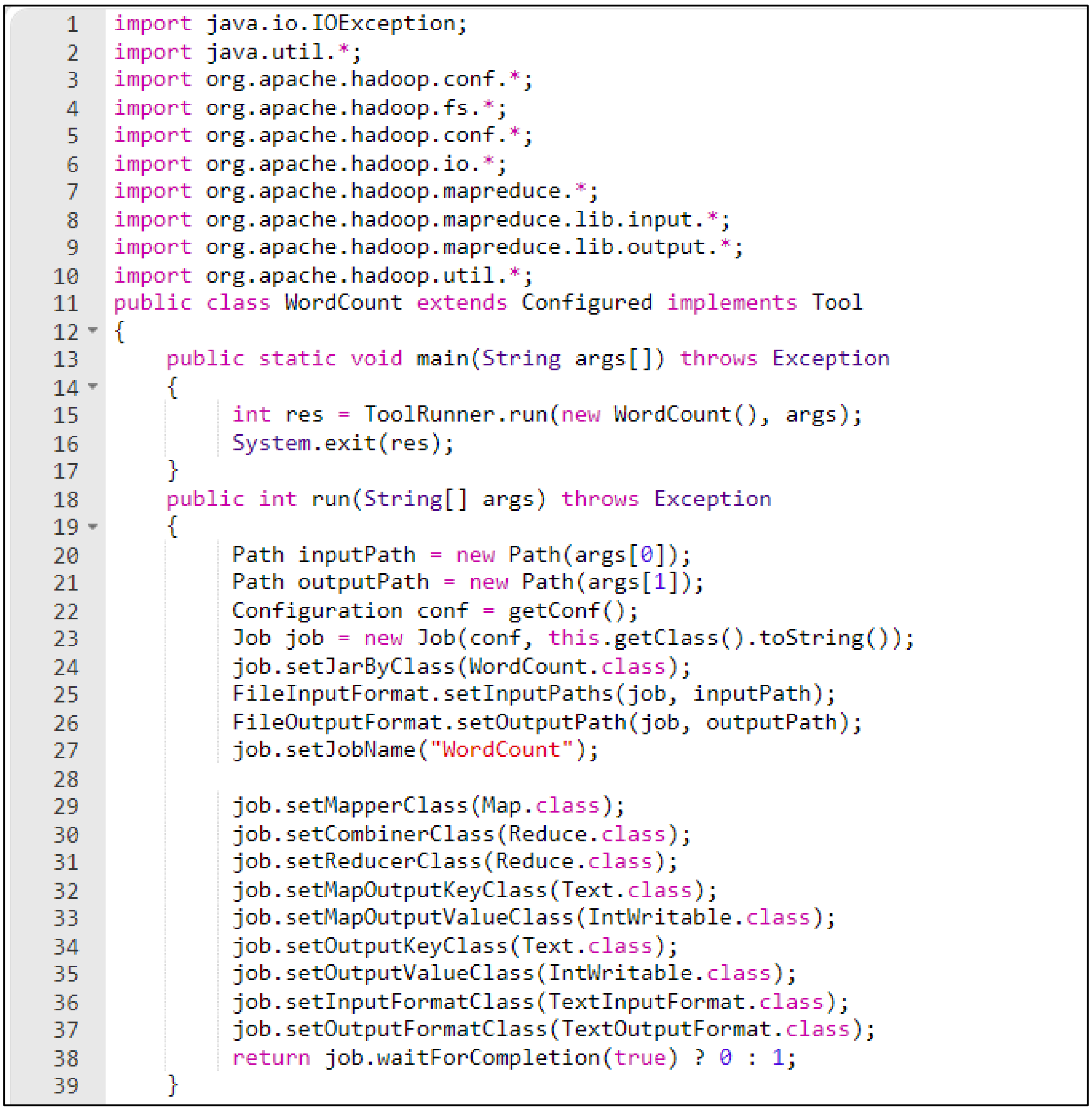
Practical Number: 11

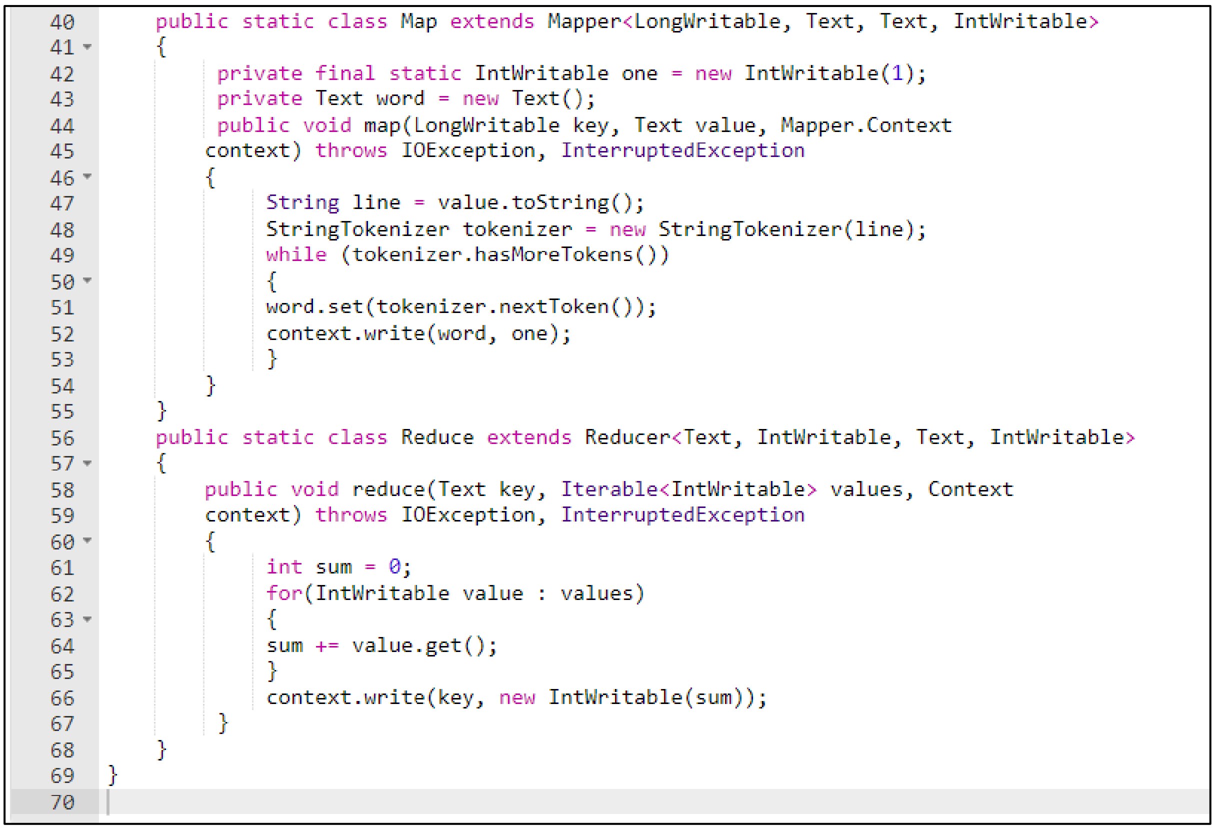
Title: Write a code in JAVA for a simple Wordcount application that counts the number of occurrences of each word in a given input set using the Hadoop MapReduce framework on local-standalone set-up.

Java Code for word count:



```java
import java.io.IOException;
import java.util.*;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.fs.*;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;
import org.apache.hadoop.util.*;
public class WordCount extends Configured implements Tool
{
    public static void main(String args[]) throws Exception
    {
        int res = ToolRunner.run(new WordCount(), args);
        System.exit(res);
    }
    public int run(String[] args) throws Exception
    {
        Path inputPath = new Path(args[0]);
        Path outputPath = new Path(args[1]);
        Configuration conf = getConf();
        Job job = new Job(conf, this.getClass().toString());
        job.setJarByClass(WordCount.class);
        FileInputFormat.setInputPaths(job, inputPath);
        FileOutputFormat.setOutputPath(job, outputPath);
        job.setJobName("WordCount");

        job.setMapperClass(Map.class);
        job.setCombinerClass(Reduce.class);
        job.setReducerClass(Reduce.class);
        job.setMapOutputKeyClass(Text.class);
        job.setMapOutputValueClass(IntWritable.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);
        return job.waitForCompletion(true) ? 0 : 1;
    }
```



```
    public static class Map extends 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, Mapper.Context
        context) throws IOException, InterruptedException
        {
             String line = value.toString();
             StringTokenizer tokenizer = new StringTokenizer(line);
             while (tokenizer.hasMoreTokens())
             {
             word.set(tokenizer.nextToken());
             context.write(word, one);
             }
        }
    }
    public static class Reduce extends Reducer<Text, IntWritable, Text, IntWritable>
    {
        public void reduce(Text key, Iterable<IntWritable> values, Context
        context) throws IOException, InterruptedException
        {
             int sum = 0;
             for(IntWritable value : values)
             {
             sum += value.get();
             }
             context.write(key, new IntWritable(sum));
         }
    }
}
```

Input File:

Pune

Mumbai

Nashik

Pune

Nashik

Kolapur