```
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
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;

public class WordSequence {

public static class WordSequenceMapper extends Mapper < LongWritable, Text, Text, IntWritable >
{
 private final static IntWritable one = new IntWritable( 1);

 @Override
 public void map( LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
            String lines = value.toString();
             StringTokenizer tokens = new StringTokenizer(lines);
```

```
        List words =  new ArrayList(tokens.countTokens());

        while(tokens.hasMoreTokens()){
                words.add(tokens.nextToken());
        }


for (int i=0; i<words.size() -4; i++){
        StringBuilder sequence = new StringBuilder();

 int counter=i;

 for (int j=0; j<5; j++){



         if(j>0){
                sequence = sequence.append(" ");
                sequence  = sequence.append(words.get(counter));



         }

        else{

                sequence = sequence.append(words.get(counter));
```

```
        }
          counter++;




}
```

```
  Text mapOutput = new Text(sequence.toString().replaceAll("[^a-zA-Z0-9 ]", "")); //Ignore all special characters from input file


  context.write( mapOutput, one);
  sequence = new StringBuilder();


  }
 }
}

public static class WordSequenceReducer extends Reducer < Text, IntWritable, Text, IntWritable >
{
 private IntWritable result = new IntWritable();
 @Override
 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(WordSequence.class);
 FileInputFormat.addInputPath( job, new Path("input"));
 FileOutputFormat.setOutputPath( job, new Path("output"));
 job.setMapperClass( WordSequenceMapper.class);
 job.setCombinerClass( WordSequenceReducer.class);
 job.setReducerClass( WordSequenceReducer.class);
 job.setOutputKeyClass( Text.class);
 job.setOutputValueClass( IntWritable.class);
 System.exit( job.waitForCompletion( true) ? 0 : 1);
 }
}
```