```
package it.polito.bigdata.hadoop.exercise17;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.MultipleInputs;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

/**
 * Driver class.
 */
public class DriverBigData extends Configured implements Tool {

  @Override
  public int run(String[] args) throws Exception {

    Path inputPath1, inputPath2;
    Path outputDir;
    int numberOfReducers;
      int exitCode;

      // Parse the parameters
    numberOfReducers = Integer.parseInt(args[0]);
    inputPath1 = new Path(args[1]);
    inputPath2 = new Path(args[2]);
    outputDir = new Path(args[3]);

    Configuration conf = this.getConf();

    // Define a new job
    Job job = Job.getInstance(conf);

    // Assign a name to the job
    job.setJobName("Exercise 17");

    // Set two input paths and two mapper classes
    MultipleInputs.addInputPath(job, inputPath1, TextInputFormat.class,
MapperType1BigData.class);
    MultipleInputs.addInputPath(job, inputPath2, TextInputFormat.class,
MapperType2BigData.class);

    // Set path of the output folder for this job
    FileOutputFormat.setOutputPath(job, outputDir);

    // Specify the class of the Driver for this job
    job.setJarByClass(DriverBigData.class);


    // Set job output format
    job.setOutputFormatClass(TextOutputFormat.class);


    // Set map output key and value classes
```

```
    job.setMapOutputKeyClass(Text.class);
    job.setMapOutputValueClass(FloatWritable.class);

    // Set reduce class
    job.setReducerClass(ReducerBigData.class);

    // Set reduce output key and value classes
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(FloatWritable.class);

    // Set number of reducers
    job.setNumReduceTasks(numberOfReducers);

    // Execute the job and wait for completion
    if (job.waitForCompletion(true)==true)
      exitCode=0;
    else
      exitCode=1;

    return exitCode;
  }

  /** Main of the driver
   */

  public static void main(String args[]) throws Exception {
      // Exploit the ToolRunner class to "configure" and run the Hadoop application
    int res = ToolRunner.run(new Configuration(), new DriverBigData(), args);

    System.exit(res);
  }
}
```