1.create java project and save with appropriate project name CSEXXX_LogAnalysis

2.Right click on project and create 3 java class , name it as
a. ProcessLogs-main class
b. LogMapper-mapper class
c. LogReducer-reducer class

3.Add Jar files-right click on project ,
a.choose build path option
b.choose configure build path
c.click on add external jars
d.path - Desktop/hadoop-java-jars - select all by pressing ctrl A , click ok

4.Add implementation code in all three java files

5.commands for execution - open new terminal

6.hadoop dfs -mkdir /input - create directory to store all dataset files in HDFS-
  hadoop dfs -ls /input   -displays files inside input directory

7.hadoop dfs -put local file system path /input/ - copies file from local file system to HDFS

8.hadoop jar path-to-jarfile mainclass /path-to-inputfile /output

copy paste path wherever required

9.to check output there are 2 ways :

a.check output from browser --- open firefox browser type in url   localhost:50070- browse filesystem choose output directory and open part-00000 file

b.use cat command

```
   hadoop dfs -ls /output --displays contents of output directory
   hadoop dfs -cat /output/part-00000 - displays contents of this file
```

Hint: In main Class change

class name in JobConf() function
class name in setMapperClass() function
class name in setreducerClass() function
main method change in run() with constructor name

ProcessLogs Main Class:

```
import java.io.*;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.FileSystem;
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 ProcessLogs extends Configured implements Tool {

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

        if(args.length<2)
        {
                System.out.println("Plz Give Input Output Directory Correctly");
                return -1;
        }

        JobConf conf = new JobConf(ProcessLogs.class);
        FileInputFormat.setInputPaths(conf,new Path(args[0]));
        FileOutputFormat.setOutputPath(conf, new Path(args[1]));
        conf.setMapperClass(LogMapper.class);
        conf.setReducerClass(LogReducer.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 ProcessLogs(), args);
                System.exit(exitcode);
        }
}
```

LogMapper Class:

```
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 LogMapper extends MapReduceBase implements
Mapper<LongWritable,Text,Text,IntWritable>
{
	public void map(LongWritable key, Text value,
			OutputCollector<Text, IntWritable> output, Reporter r)
			throws IOException {

		String[] s = value.toString().split(" ");
		String ip = s[0];

		output.collect(new Text(ip), new IntWritable(1));
	}

}
```

LogReducer Class:

```
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 LogReducer extends MapReduceBase implements
Reducer<Text,IntWritable,Text,IntWritable>
{
        public void reduce(Text key, Iterator<IntWritable> values,
                        OutputCollector<Text, IntWritable> output, Reporter r)
                        throws IOException {

        int count=0;
        while(values.hasNext())
        {
                IntWritable i= values.next();
                count+= i.get();
        }
        output.collect(key, new IntWritable(count));

        }

}
```