```java
package maxLog;

import java.io.IOException;

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.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;


import maxLog.LogMapper;
import maxLog.LogReducer;

public class LogMain {
    public static void main(String[] args) throws IllegalArgumentException, IOException, ClassNotFoundException, InterruptedException {
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "Maximum Log");
        job.setJarByClass(LogMain.class);
        job.setMapperClass(LogMapper.class);
        job.setReducerClass(LogReducer.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

        try {
            FileSystem fs = FileSystem.get(URI.create("hdfs://localhost:9000" + args[1]), conf);
            String fname = fs.listStatus(new Path(args[1]))[1].getPath().toString();
            FSDataInputStream in = null;
            in = fs.open(new Path(fname));
//            File Obj = new File("/home/hduser/Documents/Assignment2/part-r-00000");
            Scanner Reader = new Scanner(in);
            String key = "";
            int max = 0;

```

```
39              while (Reader.hasNextLine()) {
40                  String data = Reader.nextLine();
41                  String[] vals = data.split("");
42  //                  System.out.println(Arrays.toString(vals));
43                  int count = Integer.parseInt(vals[1]);
44                  if (count > max) {
45                      key = vals[0];
46                      max = count;
47                  }
48              }
49              System.out.println(
    "-----------------------------------------");
50              System.out.println("IP address used maximum no of
    times:- ");
51              System.out.println(key + " -> " + max);
52              System.out.println(
    "-----------------------------------------");
53              Reader.close();
54          }
55          catch (FileNotFoundException e) {
56              System.out.println("An error has occurred.");
57              e.printStackTrace();
58          }
59      }
60  }
61
```

```java
package maxLog;

import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Mapper.Context;
import org.apache.hadoop.io.Text;

import java.io.IOException;
import java.util.StringTokenizer;

import org.apache.hadoop.io.IntWritable;

public class LogMapper extends Mapper<Object, Text, Text, IntWritable>{
    public void map(Object key, Text value, Context context) throws IOException, InterruptedException {
        StringTokenizer itr = new StringTokenizer(value.toString());
        while (itr.hasMoreTokens()) {
            String token = itr.nextToken();
            if (check(token)) {
                context.write(new Text(token), new IntWritable(1));
            }
//            context.write(new Text(token), new IntWritable(1));
        }
    }

    private boolean check(String token) {

        boolean dot = false;
        while (!token.isEmpty()) {
            char first = token.charAt(0);
            if (first == '.') {
                dot = true;
            }
            if (!Character.isDigit(first) && first != '.') {
                return false;
            }
            token = token.substring(1);
        }
        return dot;
    }

```

```
40
41 }
42
```

```java
package maxLog;

import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.Reducer.Context;
import org.apache.hadoop.io.Text;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;

public class LogReducer extends Reducer<Text, IntWritable, Text
, IntWritable>{
    public void reduce(Text key, Iterable<IntWritable> values,
Context context) throws IOException, InterruptedException {
        int sum = 0;
        for (IntWritable val : values) {
            sum += val.get();
        }

        context.write(key, new IntWritable(sum));
    }
}

```