**Find the mean max temperature for every month**

**MeanMaxDriver.class**

**package** meanmax;  
  
**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;  
  
**public** **class** MeanMaxDriver {  
 **public** **static** **void** main(String[] args) **throws** Exception {  
 **if** (args.length != 2) {  
 System.err.println("Please Enter the input and output parameters");  
 System.exit(-1);  
 }   
 Job job = **new** Job();  
 job.setJarByClass(MeanMaxDriver.**class**);  
 job.setJobName("Max temperature");  
 FileInputFormat.addInputPath(job, **new** Path(args[0]));  
 FileOutputFormat.setOutputPath(job, **new** Path(args[1]));  
 job.setMapperClass(MeanMaxMapper.**class**);  
 job.setReducerClass(MeanMaxReducer.**class**);  
 job.setOutputKeyClass(Text.**class**);  
 job.setOutputValueClass(IntWritable.**class**);  
 System.exit(job.waitForCompletion(**true**) ? 0 : 1);  
 }  
}  
  
**MeanMaxMapper.class**

**package** meanmax;  
  
**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.mapreduce.Mapper;  
  
**public** **class** MeanMaxMapper **extends** Mapper<LongWritable, Text, Text, IntWritable> {  
 **public** **static** **final** **int** MISSING = 9999;  
   
 **public** **void** map(LongWritable key, Text value, Mapper<LongWritable, Text, Text, IntWritable>.Context context) **throws** IOException, InterruptedException {  
 **int** temperature;  
 String line = value.toString();  
 String month = line.substring(19, 21);  
 **if** (line.charAt(87) == '+') {  
 temperature = Integer.parseInt(line.substring(88, 92));  
 } **else** {  
 temperature = Integer.parseInt(line.substring(87, 92));  
 }   
 String quality = line.substring(92, 93);  
 **if** (temperature != 9999 && quality.matches("[01459]"))  
 context.write(**new** Text(month), **new** IntWritable(temperature));   
 }  
}  
  
**MeanMaxReducer.class**

**package** meanmax;  
  
**import** java.io.IOException;  
**import** org.apache.hadoop.io.IntWritable;  
**import** org.apache.hadoop.io.Text;  
**import** org.apache.hadoop.mapreduce.Reducer;  
  
**public** **class** MeanMaxReducer **extends** Reducer<Text, IntWritable, Text, IntWritable> {  
 **public** **void** reduce(Text key, Iterable<IntWritable> values, Reducer<Text, IntWritable, Text, IntWritable>.Context context) **throws** IOException, InterruptedException {  
 **int** max\_temp = 0;  
 **int** total\_temp = 0;  
 **int** count = 0;  
 **int** days = 0;  
 **for** (IntWritable value : values) {  
 **int** temp = value.get();  
 **if** (temp > max\_temp)  
 max\_temp = temp;   
 count++;  
 **if** (count == 3) {  
 total\_temp += max\_temp;  
 max\_temp = 0;  
 count = 0;  
 days++;  
 }   
 }   
 context.write(key, **new** IntWritable(total\_temp / days));  
 }  
}

**Output**

