

```

// MaxTemperatureDriver.java

package MaxMinTemp;

import org.apache.hadoop.conf.Configured;
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 org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class MaxTemperatureDriver extends Configured implements Tool{
    public int run(String[] args) throws Exception {
        if(args.length !=2) {
            System.err.println("Usage: MaxTemperatureDriver <input path>
<outputpath>");
            System.exit(-1);
        }
        Job job = new Job();
        job.setJarByClass(MaxTemperatureDriver.class);
        job.setJobName("Max Temperature");
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));
        job.setMapperClass(MaxTemperatureMapper.class);
        job.setReducerClass(MaxTemperatureReducer.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        System.exit(job.waitForCompletion(true) ? 0:1);
        boolean success = job.waitForCompletion(true);
        return success ? 0 : 1;
    }

    public static void main(String[] args) throws Exception {
        MaxTemperatureDriver driver = new MaxTemperatureDriver();
        int exitCode = ToolRunner.run(driver, args);
        System.exit(exitCode);
    }
}

```

```
// MaxTemperatureMapper.java
```

```
package MaxMinTemp;
```

```
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 MaxTemperatureMapper extends Mapper<LongWritable, Text, Text,  
IntWritable> {
```

```
    private static final int MISSING = 9999;
```

```
    @Override
```

```
    public void map(LongWritable key, Text value, Context context) throws  
IOException, InterruptedException {
```

```
        String line = value.toString();
```

```
        String year = line.substring(15, 19);
```

```
        int airTemperature;
```

```
        if (line.charAt(87) == '+') { // parseInt doesn't like leading plus signs
```

```
            airTemperature = Integer.parseInt(line.substring(88, 92));
```

```
        } else {
```

```
            airTemperature = Integer.parseInt(line.substring(87, 92));
```

```
        }
```

```
        String quality = line.substring(92, 93);
```

```
        if (airTemperature != MISSING && quality.matches("[01459]")) {
```

```
            context.write(new Text(year), new IntWritable(airTemperature));
```

```
        }
```

```
    }
```

```
}
```

```
// MaxTemperatureReducer.java
```

```
package MaxMinTemp;
```

```
import java.io.IOException;
```

```
import org.apache.hadoop.io.IntWritable;
```

```
import org.apache.hadoop.io.Text;
```

```
import org.apache.hadoop.mapreduce.Reducer;
```

```
public class MaxTemperatureReducer extends Reducer<Text, IntWritable, Text,  
IntWritable> {
```

```
    @Override
```

```
    public void reduce(Text key, Iterable<IntWritable> values, Context context)  
throws IOException, InterruptedException {
```

```
        int maxValue = Integer.MIN_VALUE;
```

```
        for (IntWritable value : values) {
```

```
            maxValue = Math.max(maxValue, value.get());
```

```
        }
```

```
        context.write(key, new IntWritable(maxValue));
```

```
    }
```

```
}
```

OUTPUT :

1901	317
1902	244
1903	289
1904	256
1905	283
1906	294
1907	283
1908	289
1909	278
1910	294
1911	306
1912	322
1913	300
1914	333
1915	294
1916	278
1917	317
1918	322
1919	378
1920	294