**WeatherMaxMin Program:**

**package com.wd;**

**import java.io.IOException;**

**import java.util.Iterator;**

**import org.apache.hadoop.fs.Path;**

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

**import org.apache.hadoop.io.LongWritable;**

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

**import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;**

**import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;**

**import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;**

**import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;**

**import org.apache.hadoop.mapreduce.Job;**

**import org.apache.hadoop.mapreduce.Mapper;**

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

**import org.apache.hadoop.conf.Configuration;**

**public class WeatherTemp {**

**//Mapper**

**public static class MaxTemperatureMapper extends**

**Mapper<LongWritable, Text, Text, Text> {**

**@Override**

**public void map(LongWritable arg0, Text Value, Context context)**

**throws IOException, InterruptedException {**

**//Converting the record (single line) to String and storing it in a String variable line**

**String line = Value.toString();**

**//Checking if the line is not empty**

**float max=9999.0f;**

**float min=-0.1f;**

**if (!(line.length() == 0))**

**{**

**String date = line.substring(6, 14); //date**

**float temp\_Max = Float.parseFloat(line.substring(39, 45).trim()); //maximum temperature**

**float temp\_Min = Float.parseFloat(line.substring(47, 53).trim()); //minimum temperature**

**if (temp\_Max == max){**

**context.write(new Text("Maximam Temperature: " + date),new Text(String.valueOf(temp\_Max)));**

**}**

**if (temp\_Min == min){**

**context.write(new Text("Minimum Temperature: " + date),new Text(String.valueOf(temp\_Min))); // Cold day**

**}**

**}**

**}**

**}**

**//Reducer**

**public static class MaxTemperatureReducer extends**

**Reducer<Text, Text, Text, Text> {**

**public void reduce(Text Key, Iterator<Text> Values, Context context)**

**throws IOException, InterruptedException {**

**//putting all the values in temperature variable of type String**

**String temperature = Values.next().toString();**

**context.write(Key, new Text(temperature));**

**}**

**}**

**public static void main(String[] args) throws Exception {**

**Configuration conf = new Configuration();**

**Job job = new Job(conf, "weather example");**

**job.setJarByClass(WeatherTemp.class);**

**job.setMapOutputKeyClass(Text.class);**

**job.setMapOutputValueClass(Text.class);**

**job.setMapperClass(MaxTemperatureMapper.class);**

**job.setReducerClass(MaxTemperatureReducer.class);**

**job.setInputFormatClass(TextInputFormat.class);**

**job.setOutputFormatClass(TextOutputFormat.class);**

**Path OutputPath = new Path(args[1]);**

**FileInputFormat.addInputPath(job, new Path(args[0]));**

**FileOutputFormat.setOutputPath(job, new Path(args[1]));**

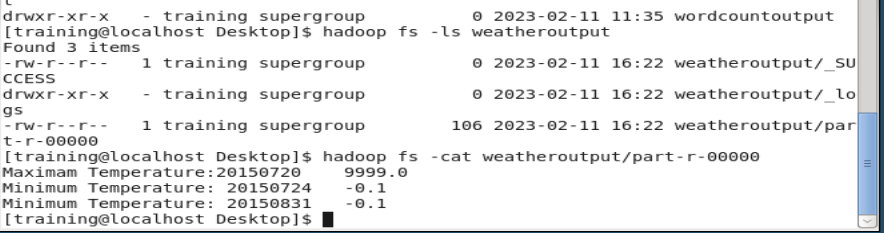
**OutputPath.getFileSystem(conf).delete(OutputPath);**

**System.exit(job.waitForCompletion(true) ? 0 : 1);**

**}**

**}**

**Output:**

****