```
package weather;

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 WeatherMapper extends Mapper <LongWritable, Text, Text, IntWritable> {

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

		//get year
		String year=value.toString().substring(15,19);
		//get temperature
		//int temperature=Integer.parseInt(value.toString().substring(87,91));

		int temperature;
		String line =value.toString();
		if(line.charAt(87)=='+') {
			temperature=Integer.parseInt(line.substring(88,92));
		}
		else {
			temperature=Integer.parseInt(line.substring(87,92));
		}

		//clean the data and write to context object
		if((temperature)!=9999) {
			con.write(new Text(year), new IntWritable(temperature));
		}
	}
}

package weather;

import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class WeatherReducer extends Reducer <Text, IntWritable, Text, IntWritable> {

    public void reduce(Text word, Iterable<IntWritable> values, Context con) throws IOException,
InterruptedException {

	int min=9999, max=-9999;

	//find min & max temp
	for(IntWritable temp:values)
	{
		if((temp.get())<min)
```

```
                {
                        min=temp.get();
                }
                else if((temp.get())>max)
                {
                        max=temp.get();
                }
        }
        //write to context object
        con.write(word, new IntWritable(max));
        con.write(word, new IntWritable(min));
    }
}

package weather;

import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FSDataInputStream;
import org.apache.hadoop.fs.FileStatus;
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.JobConf;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;



public class WeatherDriver {

public static void main(String[] args) throws IOException, ClassNotFoundException,
InterruptedException {

                float max=-9999, min=9999, temp;
                String str, year1 = null, year2=null;


                JobConf conf = new JobConf(WeatherDriver.class);
      conf.setJobName("Weatherdetails");
      Job job=new Job(conf);


                job.setMapperClass(WeatherMapper.class);
                job.setReducerClass(WeatherReducer.class);

                job.setOutputKeyClass(Text.class);
                job.setOutputValueClass(IntWritable.class);

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

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

        job.waitForCompletion(true);

        FileSystem fs=FileSystem.get(conf);
        FileStatus[] status=fs.listStatus(new Path("hdfs://localhost:9000"+args[1]));
        //System.out.println("status[0]:"+status[0]+"  and  status[1]:"+status[1]);
        FSDataInputStream fd= fs.open(status[1].getPath());

        str=fd.readLine();
        int a=1;
        while(str!=null)
        {
            String[] arr=str.split("\t");
            temp=Integer.parseInt(arr[1]);
            if(a%2)
            {
                if(temp>max)
                {
                    max=temp;
                    year1=arr[0];
                }
            }
            else
            {
                if(temp<min)
                {
                    min=temp;
                    year2=arr[0];
                }
            }
            a++;
            str=fd.readLine();
        }
        System.out.println("Max temperature is " + max/10 + "  from year " + year1);
        System.out.println("Min temperature is " + min/10 + "  from year " + year2);
    }
}
```