3. Develop a mapreduce program that mines weather data and display appropriate messages indicating the weather conditions of the day.

// java program “WeatherDataAnalysis”

```
import java.io.*;

import java.util.*;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.*;

import org.apache.hadoop.mapreduce.*;

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

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


public class WeatherDataAnalysis {


   public static class WeatherMapper extends Mapper<LongWritable, Text, Text, Text> {

      @Override

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

         String[] tokens = value.toString().split(",");


         if (tokens.length == 5 && !tokens[0].equals("Date")) { // Ignore header line

            String date = tokens[0];

            String location = tokens[1];

            int temperature = Integer.parseInt(tokens[2]);


            String condition;

            if (temperature > 35) {

               condition = "Hot";

            } else if (temperature < 15) {

               condition = "Cold";

            } else {

               condition = "Normal";
```

```
            }

            context.write(new Text(date + "," + location), new Text(condition));
        }
    }
}

public static class WeatherReducer extends Reducer<Text, Text, Text, Text> {
    @Override
    protected void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
InterruptedException {
        for (Text value : values) {
            context.write(key, value);
        }
    }
}

public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = Job.getInstance(conf, "Weather Data Analysis");

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

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

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

```
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));
        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
}
```

Input data

- Date,Location,Temperature,Humidity,Pressure

2025-03-15,NewYork,35,60,1012

2025-03-15,LosAngeles,40,30,1010

2025-03-15,Chicago,28,70,1008

Output: Messages indicating the weather condition for each location, like:

- **"Hot"** if Temperature > 35
- **"Cold"** if Temperature < 15

  "Normal" otherwise

View the Output

```
hdfs dfs -cat /weatheroutput/part-r-00000
```

The output will display messages like:

```
2025-03-15,NewYork      Hot
2025-03-15,LosAngeles   Hot
2025-03-15,Chicago      Normal
```