Group B
Practical 3
Locate dataset (e.g., sample_weather.txt) for working on weather data which reads the text input files and finds average for temperature, dew point and wind speed.

WeatherDriver.java :

```
package com.example.weather;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.*;
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 Exception {

          System.out.println("Number of args: " + args.length);
          for (int i = 0; i < args.length; i++) {
        System.out.println("Arg[" + i + "]: " + args[i]);
          }

      if (args.length != 2) {
         System.err.println("Usage: WeatherDriver <input path> <output path>");
         System.exit(-1);
      }

      Configuration conf = new Configuration();
      Job job = Job.getInstance(conf, "Weather Data Average");

      job.setJarByClass(WeatherDriver.class);
      job.setMapperClass(WeatherMapper.class);
      job.setReducerClass(WeatherReducer.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);
   }
}
```

WeatherMapper.java :

```
package com.example.weather;

import java.io.IOException;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.Mapper;

public class WeatherMapper extends Mapper<LongWritable, Text, Text, Text> {
   private final static Text outKey = new Text("weather");

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

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

        // Skip header line
        if (line.startsWith("Formatted Date") || line.contains("Temperature (C)")) {
            return;
        }

        // Split CSV on commas
        String[] parts = line.split(",");

        // Ensure we have at least 7 columns
        if (parts.length > 6) {
            try {
                String temperature = parts[3].trim();    // Temperature (C)
                String humidity    = parts[5].trim();    // Humidity
                String windSpeed   = parts[6].trim();    // Wind Speed (km/h)

                context.write(outKey, new Text(temperature + "," + humidity + "," + windSpeed));
            } catch (Exception e) {
                // Skip malformed rows
            }
        }
    }
}
```

WeatherReducer.java :

```
package com.example.weather;

import java.io.IOException;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.Reducer;

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

    @Override
    public void reduce(Text key, Iterable<Text> values, Context context) throws IOException, InterruptedException {
        double totalTemp     = 0;
        double totalHumidity = 0;
        double totalWind     = 0;
        int count            = 0;

        for (Text val : values) {
            String[] nums = val.toString().split(",");
            if (nums.length == 3) {
                try {
                    totalTemp     += Double.parseDouble(nums[0]);
                    totalHumidity += Double.parseDouble(nums[1]);
                    totalWind     += Double.parseDouble(nums[2]);
                    count++;
                } catch (NumberFormatException ignored) {}
            }
        }

        if (count > 0) {
            double avgTemp     = totalTemp / count;
            double avgHumidity = totalHumidity / count;
            double avgWind     = totalWind / count;
```

```
        context.write(new Text("Average Temperature (C):"), new Text(String.format("%.2f", avgTemp)));
        context.write(new Text("Average Humidity:"),      new Text(String.format("%.2f", avgHumidity)));
        context.write(new Text("Average Wind Speed (km/h):"), new Text(String.format("%.2f", avgWind)));
      }
   }
}
```

pom.xml :

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
 <modelVersion>4.0.0</modelVersion>
 <groupId>weather.forcast</groupId>
 <artifactId>WeatherDataForecasting</artifactId>
 <packaging>jar</packaging>
 <version>1.0-SNAPSHOT</version>
 <name>WeatherDataForecasting</name>
 <url>http://maven.apache.org</url>
 <dependencies>
  <!-- Hadoop Common -->
  <dependency>
     <groupId>org.apache.hadoop</groupId>
     <artifactId>hadoop-common</artifactId>
     <version>3.4.0</version> <!-- Match this with your Hadoop version -->
  </dependency>

  <!-- Hadoop MapReduce -->
  <dependency>
     <groupId>org.apache.hadoop</groupId>
     <artifactId>hadoop-mapreduce-client-core</artifactId>
     <version>3.4.0</version> <!-- Match this with your Hadoop version -->
  </dependency>

  <!-- Hadoop IO -->
  <dependency>
     <groupId>org.apache.hadoop</groupId>
     <artifactId>hadoop-mapreduce-client-jobclient</artifactId>
     <version>3.4.0</version> <!-- Match this with your Hadoop version -->
  </dependency>

  <!-- SLF4J API and Logger bindings for Hadoop -->
  <dependency>
     <groupId>org.slf4j</groupId>
     <artifactId>slf4j-api</artifactId>
     <version>1.7.32</version>
  </dependency>
  <dependency>
     <groupId>org.slf4j</groupId>
     <artifactId>slf4j-log4j12</artifactId>
     <version>1.7.32</version>
 </dependency>

 <!-- JUnit for test cases -->
  <dependency>
     <groupId>junit</groupId>
     <artifactId>junit</artifactId>
     <version>4.13.2</version>
```

```
      <scope>test</scope>
    </dependency>
  </dependencies>
  <build>
    <sourceDirectory>src/main/java</sourceDirectory>
    <resources>
      <resource>
        <directory>src/main/resources</directory>
      </resource>
    </resources>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>3.8.1</version>
        <configuration>
          <source>1.8</source>
          <target>1.8</target>
        </configuration>
      </plugin>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-jar-plugin</artifactId>
        <version>3.2.0</version>
        <configuration>
          <archive>
            <manifest>
              <mainClass>weather.forcast.WeatherDriver</mainClass>
            </manifest>
          </archive>
        </configuration>
      </plugin>
    </plugins>
  </build>
</project>
```

Output :

```
pratik@DESKTOP-T62QGCD:~/TE_Practical/DSBDA-Group-B/WeatherDataForecasting$ mvn clean package

pratik@DESKTOP-T62QGCD:~/TE_Practical/DSBDA-Group-B/WeatherDataForecasting$ hadoop fs -mkdir /weather

pratik@DESKTOP-T62QGCD:~/TE_Practical/DSBDA-Group-B/WeatherDataForecasting$ cp ~/weatherforcast.csv
src/main/resources/

pratik@DESKTOP-T62QGCD:~/TE_Practical/DSBDA-Group-B/WeatherDataForecasting$ hadoop fs -put
src/main/resources/weatherHistory.csv /weather

pratik@DESKTOP-T62QGCD:~/TE_Practical/DSBDA-Group-B/WeatherDataForecasting$ hadoop jar
target/WeatherDataForecasting-1.0-SNAPSHOT.jar /weather /weather/output

pratik@DESKTOP-T62QGCD:~/TE_Practical/DSBDA-Group-B/WeatherDataForecasting$ hadoop fs -cat
/weather/output/part-r-00000
Average Temperature (C):        11.93
Average Humidity:       0.73
Average Wind Speed (km/h):      10.81
```