

# LAB PROGRAMS -9<sup>TH</sup> JULY

-Jeevan Raj H (1MS23SCN06)

## 1.Odd Even Sum

### driver.java

```
package oddeven;
import java.io.*;
import java.util.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.fs.Path;

public class driver
{
    public static void main(String args[]) throws IOException
    {
        JobConf conf=new JobConf(driver.class);
        conf.setMapperClass(mapper.class);
        conf.setReducerClass(reducer.class);
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(IntWritable.class);
        FileInputFormat.addInputPath(conf, new Path(args[0]));
        FileOutputFormat.setOutputPath(conf,new Path(args[1]));
        JobClient.runJob(conf);
    }
}
```

### mapper.java

```
package oddeven;
import java.io.*;
import java.util.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.io.*;

public class mapper extends MapReduceBase implements Mapper<LongWritable , Text ,
Text , IntWritable>
{
    public void map(LongWritable key,Text value,OutputCollector<Text,IntWritable>
output,Reporter r) throws IOException
```

```

{
    String[] line=value.toString().split(" ");
    for(String num:line){
        int number=Integer.parseInt(num);
        if(number%2==0) {
            output.collect(new Text("even"),new IntWritable(number));
        }
        else{
            output.collect(new Text("odd"),new IntWritable(number));
        }
    }
}
}

```

## **reducer.java**

```

package oddeven;
import java.io.*;
import java.util.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.io.*;
public class reducer extends MapReduceBase implements
Reducer<Text,IntWritable,Text,IntWritable>
{
    public void reduce(Text key,Iterator<IntWritable>
value,OutputCollector<Text,IntWritable> output ,Reporter r) throws IOException
    {
        int sum=0,count=0;
        while(value.hasNext()){
            sum+=value.next().get();
            count++;
        }
        output.collect(new Text("Sum of "+key+" Numbers"),new IntWritable(sum));
        output.collect(new Text(key+" Number count"),new IntWritable(count));
    }
}

```

## **int.txt**

1 2 3 4 5 6 7 8 9 10

## **Steps to run**

1. Create a New File named Bash.sh
2. Copy the Below code and Paste inside Bash.sh and save that File.  
export JAVA\_HOME=\$(readlink -f \$(which javac) | awk 'BEGIN {FS="/bin"} {print

```
$1}})
```

```
export PATH=$(echo $PATH):$(pwd)/bin
```

```
export CLASSPATH=$(hadoop classpath)
```

3. Execute the bash.sh File using following command source Bash.sh.

4. Verify JAVA\_HOME variable to be set to Java Path and PATH variable has your USN

Hadoop Folder.If any previous PATH set to Hadoop Folder remove that inside .bashrc file.

5. Verify Hadoop is Installed or not by executing hadoop command.if command gives Information about Hadoop command then Hadoop is Successfully Installed.

6. Create a folder oddeven and move to that folder

7. Make the driver.java , mapper.java and reducer.java files

8. Compile all java files (driver.java mapper.java reducer.java)

```
javac -d . *.java
```

9. Set driver class in manifest

```
echo Main-Class: oddeven.driver > Manifest.txt
```

10. Create an executable jar file

```
jar cfm oddeven.jar Manifest.txt oddeven/*.class
```

11. oe.txt is input file for Oddeven create Input File

```
echo 1 2 3 4 5 6 7 8 9 10 > oe.txt
```

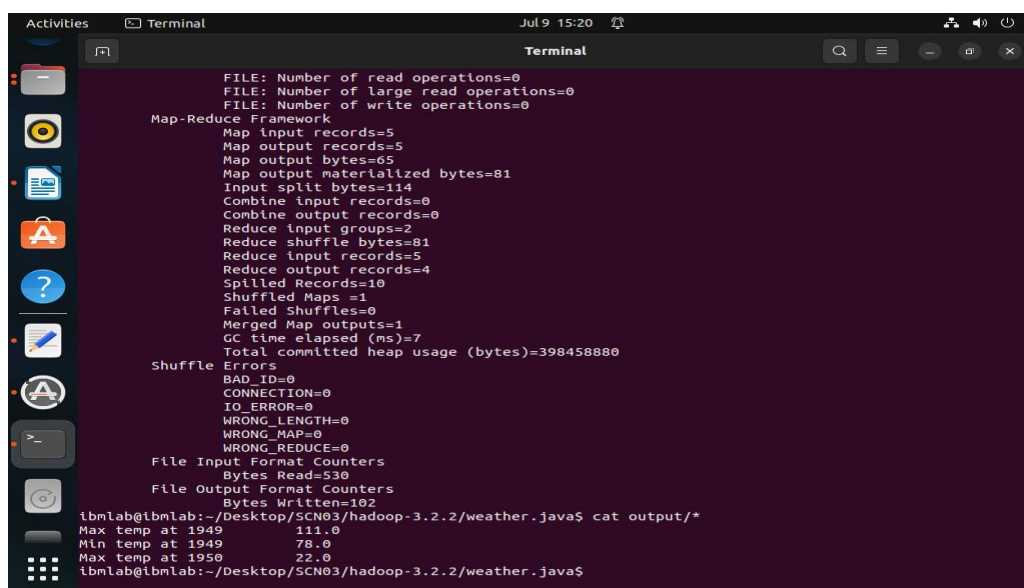
12. Run the jar file

```
hadoop jar oddeven.jar oe.txt output
```

13. To see the Output

```
cat output/*
```

## Output Screenshots



```
Activities Terminal Jul 9 15:20
Terminal
FILE: Number of read operations=0
FILE: Number of large read operations=0
FILE: Number of write operations=0
Map-Reduce Framework
  Map input records=5
  Map output records=5
  Map output bytes=65
  Map output materialized bytes=81
  Input split bytes=114
  Combine input records=0
  Combine output records=0
  Reduce input groups=2
  Reduce shuffle bytes=81
  Reduce input records=5
  Reduce output records=4
  Spilled Records=10
  Shuffled Maps =1
  Failed Shuffles=0
  Merged Map outputs=1
  GC time elapsed (ms)=7
  Total committed heap usage (bytes)=398458880
Shuffle Errors
  BAD_ID=0
  CONNECTION=0
  IO_ERROR=0
  WRONG_LENGTH=0
  WRONG_MAP=0
  WRONG_REDUCE=0
File Input Format Counters
  Bytes Read=530
File Output Format Counters
  Bytes Written=102
ibmlab@ibmlab:~/Desktop/SCN03/hadoop-3.2.2/weather.java$ cat output/*
Max temp at 1949      111.0
Min temp at 1949      78.0
Max temp at 1950      22.0
ibmlab@ibmlab:~/Desktop/SCN03/hadoop-3.2.2/weather.java$
```

## 2. Weather

### driver.java

```
package weather;
import java.util.*;
import java.io.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.fs.Path;

public class driver
{
    public static void main(String args[]) throws IOException
    {
        JobConf conf=new JobConf(driver.class);
        conf.setMapperClass mapper.class;
        conf.setReducerClass(reducer.class);
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(DoubleWritable.class);
        FileInputFormat.addInputPath(conf, new Path(args[0]));
        FileOutputFormat.setOutputPath(conf,new Path(args[1]));
        JobClient.runJob(conf);
    }
}
```

### mapper.java

```
package weather;
import java.util.*;
import java.io.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.io.*;

public class mapper extends MapReduceBase implements Mapper<LongWritable,
Text,Text,DoubleWritable>{
    public void map(LongWritable key , Text value , OutputCollector<Text,DoubleWritable>
output, Reporter r) throws IOException
    {
        String line=value.toString();
```

```

        String year=line.substring(15,19);
        Double temp=Double.parseDouble(line.substring(87,92));
        output.collect(new Text(year), new DoubleWritable(temp));
    }
}

```

## **reducer.java**

```

package weather;
import java.util.*;
import java.io.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.io.*;
class reducer extends MapReduceBase implements
Reducer<Text,DoubleWritable,Text,DoubleWritable> {
    public void reduce(Text key, Iterator<DoubleWritable> value,
OutputCollector<Text,DoubleWritable> output, Reporter r) throws IOException{
        Double max=-9999.0;
        Double min=9999.0;
        while(value.hasNext()){
            Double temp=value.next().get();
            max=Math.max(max,temp);
            min=Math.min(min,temp);
        }
        output.collect(new Text("Max temp at "+ key), new DoubleWritable(max));
        output.collect(new Text("Min temp at "+ key), new DoubleWritable(min));
    }
}

```

## **Input.txt**

```

00670119909999991950051507004+68750+023550FM-
12+038299999V0203301N00671220001CN9999999N9+00001+99999999999
00430119909999991950051512004+68750+023550FM-
12+038299999V0203201N00671220001CN9999999N9+00221+99999999999
00430119909999991950051518004+68750+023550FM-
12+038299999V0203201N00261220001CN9999999N9-00111+99999999999
00430126509999991949032412004+62300+010750FM-
12+048599999V0202701N00461220001CN0500001N9+01111+99999999999
00430126509999991949032418004+62300+010750FM-
12+048599999V0202701N00461220001CN0500001N9+00781+99999999999

```

## **Steps to run**

1. Create a New File named Bash.sh
2. Copy the Below code and Paste inside Bash.sh and save that File.  

```
export JAVA_HOME=$(readlink -f $(which javac) | awk 'BEGIN {FS="/bin"} {print $1}')
```

```
export PATH=$(echo $PATH):$(pwd)/bin
```

```
export CLASSPATH=$(hadoop classpath)
```
3. Execute the bash.sh File using following command `source Bash.sh`.
4. Verify `JAVA_HOME` variable to be set to Java Path and `PATH` variable has your USN Hadoop Folder.If any previous `PATH` set to Hadoop Folder remove that inside `.bashrc` file.
5. Verify Hadoop is Installed or not by executing `hadoop` command.if command gives Information about Hadoop command then Hadoop is Successfully Installed.
6. Create a folder `weather` and move to that folder
7. Make the `driver.java` , `mapper.java` and `reducer.java` files
8. Compile all java files (`driver.java` `mapper.java` `reducer.java`)  

```
javac -d . *.java
```
9. Set driver class in manifest  

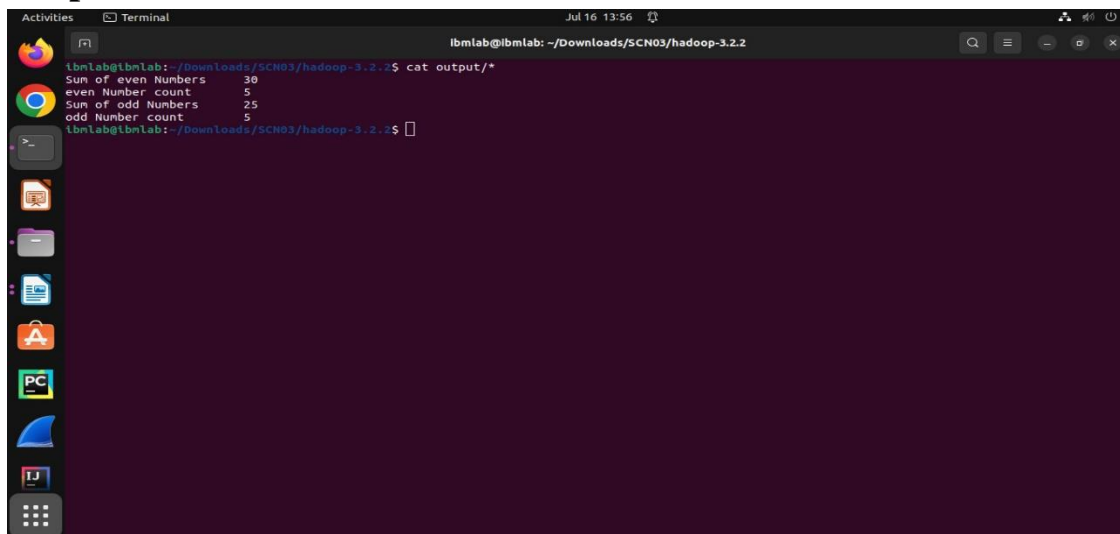
```
echo Main-Class: weather.driver > Manifest.txt
```
10. Create an executable jar file  

```
jar cfm oddeven.jar Manifest.txt weather/*.class
```
11. create input file `input.txt`
12. Run the jar file  

```
hadoop jar weather.jar oe.txt output
```
13. To see the Output  

```
cat output/*
```

## Output Screenshots



A terminal window screenshot showing the output of the Hadoop job. The terminal title is "Terminal" and the window title is "ibmlab@ibmlab: ~/Downloads/SCN03/hadoop-3.2.2". The output is as follows:

```
ibmlab@ibmlab: ~/Downloads/SCN03/hadoop-3.2.2$ cat output/*
Sun of even Numbers      30
even Number count        5
Sun of odd Numbers       25
odd Number count         5
ibmlab@ibmlab: ~/Downloads/SCN03/hadoop-3.2.2$
```

