Program 2

AIM: Word count application using Mapper-Reducer on single node cluster.

In Hadoop, MapReduce is a computation that decomposes large manipulation jobs into individual tasks that can be executed in parallel cross a cluster of servers. The results of tasks can be joined together to compute final results.

MapReduce works by breaking the processing into two phases: the map phase and the reduce phase. Each phase has key-value pairs as input and output, the types of which may be chosen by the programmer.

**** Write down about Map and Shuffle phase with flow chart.**

**** Steps:**

Step 1. Open Eclipse> File > New > Java Project >(Name it – WordCountProject) > Finish

Step 2. Right Click > New > Package (Name it - wordcount) > Finish

Step 3. Right Click on Package > New > Class (Name it - WordCount)

Step 4. Add Following Reference Libraries –

Right Click on Project > Build Path> Add External Archivals

(Here you add ALLJAR)

Step 5. Type following Program :

```
import java.io.IOException;
import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
```

```
public class WordCount {

public static class TokenizerMapper
extends Mapper<Object, Text, Text, IntWritable>{

private final static IntWritable one = new IntWritable(1);
private Text word = new Text();

public void map(Object key, Text value, Context context) throws
IOException, InterruptedException {
StringTokenizer itr = new StringTokenizer(value.toString());
while (itr.hasMoreTokens()) {
word.set(itr.nextToken());
context.write(word, one);
}
}
}

public static class IntSumReducer
extends Reducer<Text,IntWritable,Text,IntWritable> {
private IntWritable result = new IntWritable();

public void reduce(Text key, Iterable<IntWritable> values,Context context)
throws IOException, InterruptedException {
int sum = 0;
for (IntWritable val : values) {
sum += val.get();
}
result.set(sum);
context.write(key, result);
}
}

public static void main(String[] args) throws Exception {
Configuration conf = new Configuration();
Job job = Job.getInstance(conf, "word count");
job.setJarByClass(WordCount.class);
job.setMapperClass(TokenizerMapper.class);
job.setCombinerClass(IntSumReducer.class);
```

```
job.setReducerClass(IntSumReducer.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(IntWritable.class);
FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job, new Path(args[1]));
System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}
```

Explanation

The program consist of 3 classes:

- Driver class (Public void static main- the entry point)
- Map class which **extends** public class Mapper<KEYIN, VALUEIN, KEYOUT, VALUEOUT> and implements the Map function.
- Reduce class which extends public class Reducer<KEYIN,VALUEIN, KEYOUT,VALUEOUT> and implements the Reduce function.

Step 6: Make Jar File

Right Click on Project> Export> Select export destination as **Jar File** > next> Finish

Step 7: Take a text file and move it in HDFS

Before moving it into HDFS Create a directory in HDFS

```
$ hadoop fs -mkdir /inputdata
$ hadoop fs -put File.txt /inputdata
```

Step 8: Run Jar file

(Hadoop jar jarpath/jarfilename.jar packageName.ClassName PathToInputTextFile PathToOutputDirectry)

```
$ hadoop jar /home/hduser/Desktop/test/test.jar
wordcount.WordCount /inputdata/File.txt /outputdata
```

Step 9: Open Result

You can see the result on terminal

```
$ hadoop fs -ls /outputdata
$ hadoop fs -cat /outputdata/part-r-00000
```

Or you can see your result in Hadoop Web Interface

http://localhost:50070/

Goto utilities> Browse file System> /outputdata

*****Note: for results take a screen shot of above and paste it in your record with part-r-00000 file.**

Program 3

AIM: Analysis of weather dataset using Mapper-Reducer on single node cluster.

** First, explain how to analyze weather data with Hadoop by taking example of NCDC dataset.
** Write down everything as same sequence as program 2.
** No need to write Map Reduce Flow chart again.