

Word Count program

Implement WordCount Program on Hadoop framework

Mapper code:

```
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.mapred.MapReduceBase;

import org.apache.hadoop.mapred.Mapper;

import org.apache.hadoop.mapred.OutputCollector;

import org.apache.hadoop.mapred.Reporter;

public class WCMapper extends MapReduceBase implements Mapper<LongWritable,
Text, Text,
IntWritable> {

    public void map(LongWritable key, Text value, OutputCollector<Text,
IntWritable> output, Reporter rep) throws IOException
    {
        String line = value.toString();
        for (String word : line.split(" "))
        {
            if (word.length() > 0)
            {
                output.collect(new Text(word), new IntWritable(1));
            }
        }
    }
}
```

Reducer code:

```
import java.io.IOException;

import java.util.Iterator;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapred.MapReduceBase;

import org.apache.hadoop.mapred.OutputCollector;

import org.apache.hadoop.mapred.Reducer;

import org.apache.hadoop.mapred.Reporter;

public class WCReducer extends MapReduceBase implements Reducer<Text,
IntWritable, Text, IntWritable> {

    public void reduce(Text key, Iterator<IntWritable> value,
OutputCollector<Text, IntWritable> output,
Reporter rep) throws IOException
    {
        int count = 0;

        while (value.hasNext())
        {
            IntWritable i = value.next();

            count += i.get();

        }

        output.collect(key, new IntWritable(count));

    }
}
```

Driver code:

```
import java.io.IOException;
```

```

import org.apache.hadoop.conf.Configured;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapred.FileInputFormat;

import org.apache.hadoop.mapred.FileOutputFormat;

import org.apache.hadoop.mapred.JobClient;

import org.apache.hadoop.mapred.JobConf;

import org.apache.hadoop.util.Tool;

import org.apache.hadoop.util.ToolRunner;

public class WCDriver extends Configured implements Tool {

    public int run(String args[]) throws IOException

    {

        if (args.length < 2)

        {

            System.out.println("Please give valid inputs");

            return -1;

        }

        JobConf conf = new JobConf(WCDriver.class);

        FileInputFormat.setInputPaths(conf, new Path(args[0]));

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

        conf.setMapperClass(WCMapper.class);

        conf.setReducerClass(WCReducer.class);

        conf.setMapOutputKeyClass(Text.class);

        conf.setMapOutputValueClass(IntWritable.class);

```

```
conf.setOutputKeyClass(Text.class);

conf.setOutputValueClass(IntWritable.class);

JobClient.runJob(conf);

return 0;

}

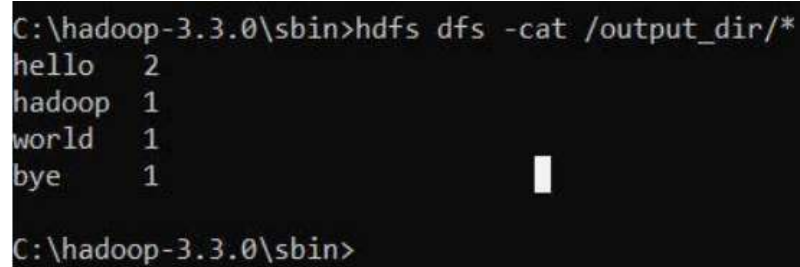
public static void main(String args[]) throws Exception
{

int exitCode = ToolRunner.run(new WCDriver(), args);

System.out.println(exitCode);

}

}
```

A terminal window with a black background and white text. The prompt is 'C:\hadoop-3.3.0\sbin>'. The command entered is 'hdfs dfs -cat /output_dir/*'. The output shows four lines: 'hello 2', 'hadoop 1', 'world 1', and 'bye 1'. A white cursor is visible on the line 'bye 1'.

```
C:\hadoop-3.3.0\sbin>hdfs dfs -cat /output_dir/*
hello 2
hadoop 1
world 1
bye 1
C:\hadoop-3.3.0\sbin>
```