## LABORATORY PROGRAM – 7

## For a given Text file, Create a Map Reduce program to sort the content in an alphabetic order listing only top 10 maximum occurrences of words.

Code, command with output:

### Driver Code:

```
package samples.topn;

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

public class TopNDriver {

    public static void main(String[] args) throws Exception {
        if (args.length != 3) {
            System.err.println("Usage: TopNDriver <in> <temp-out> <final-out>");
            System.exit(2);
        }

        Configuration conf = new Configuration();

        // === Job 1: Word Count ===
        Job wcJob = Job.getInstance(conf, "word count");
        wcJob.setJarByClass(TopNDriver.class);
        wcJob.setMapperClass(WordCountMapper.class);
        wcJob.setCombinerClass(WordCountReducer.class);
        wcJob.setReducerClass(WordCountReducer.class);
        wcJob.setOutputKeyClass(Text.class);
        wcJob.setOutputValueClass(IntWritable.class);

        FileInputFormat.addInputPath(wcJob, new Path(args[0]));
        Path tempDir = new Path(args[1]);
        FileOutputFormat.setOutputPath(wcJob, tempDir);

        if (!wcJob.waitForCompletion(true)) {
            System.exit(1);
        }

        // === Job 2: Top N ===
        Job topJob = Job.getInstance(conf, "top 10 words");
        topJob.setJarByClass(TopNDriver.class);
        topJob.setMapperClass(TopNMapper.class);
        topJob.setReducerClass(TopNReducer.class);
        topJob.setMapOutputKeyClass(IntWritable.class);
        topJob.setMapOutputValueClass(Text.class);
        topJob.setOutputKeyClass(Text.class);
        topJob.setOutputValueClass(IntWritable.class);

        FileInputFormat.addInputPath(topJob, tempDir);
        FileOutputFormat.setOutputPath(topJob, new Path(args[2]));

        System.exit(topJob.waitForCompletion(true) ? 0 : 1);
    }
}
```

## Mapper Code:

```
package samples.topn;

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

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class WordCountMapper
    extends Mapper<Object, Text, Text, IntWritable> {

    private final static IntWritable ONE = new IntWritable(1);
    private Text word = new Text();
    // characters to normalize into spaces
    private String tokens = "[_|$#<>\\^=\\[\\]\\*/\\\\,;,.\\-:()?!\""]";

    @Override
    protected void map(Object key, Text value, Context context)
        throws IOException, InterruptedException {

        // clean & tokenize
        String clean = value.toString()
                       .toLowerCase()
                       .replaceAll(tokens, " ");
        StringTokenizer itr = new StringTokenizer(clean);
        while (itr.hasMoreTokens()) {
            word.set(itr.nextToken().trim());
            context.write(word, ONE);
        }
    }
}
```

## Mapper Code:

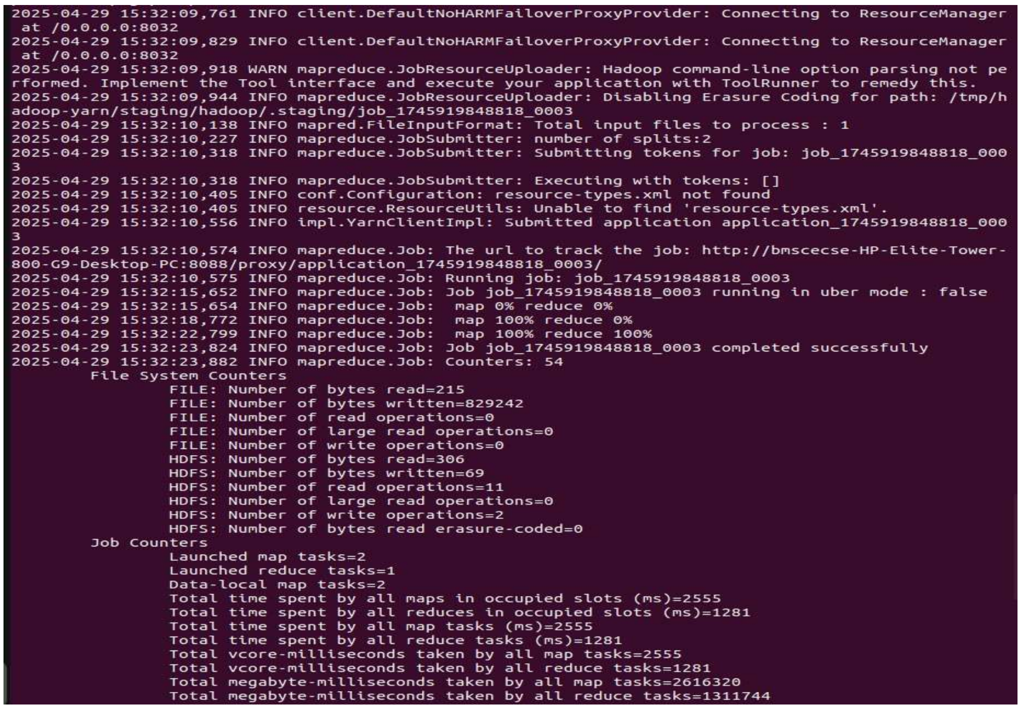
```
package samples.topn;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class TopNMapper
    extends Mapper<Object, Text, IntWritable, Text> {

    private IntWritable count = new IntWritable();
    private Text word = new Text();

    @Override
    protected void map(Object key, Text value, Context context)
        throws IOException, InterruptedException {

        // input line: word \t count
        String[] parts = value.toString().split("\\t");
        if (parts.length == 2) {
            word.set(parts[0]);
            count.set(Integer.parseInt(parts[1]));
            // emit count → word, so Hadoop sorts by count
            context.write(count, word);
        }
    }
}
```

## Reducer Code:

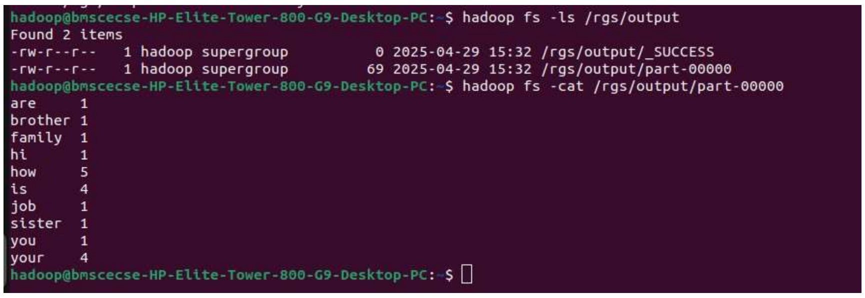
```
package samples.topn;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class WordCountReducer
    extends Reducer<Text, IntWritable, Text, IntWritable> {

    @Override
    protected void reduce(Text key, Iterable<IntWritable> values, Context context)
        throws IOException, InterruptedException {

        int sum = 0;
        for (IntWritable val : values) {
            sum += val.get();
        }
        context.write(key, new IntWritable(sum));
    }
}
```

## Reducer Code :

```
package samples.topn;

import java.io.IOException;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.Map;
import java.util.TreeMap;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class TopNReducer
    extends Reducer<IntWritable, Text, Text, IntWritable> {

    // TreeMap with descending order of keys (counts)
    private TreeMap<Integer, List<String>> countMap =
        new TreeMap<>(Collections.reverseOrder());

    @Override
    protected void reduce(IntWritable key, Iterable<Text> values, Context context)
        throws IOException, InterruptedException {

        int cnt = key.get();
        List<String> words = countMap.getOrDefault(cnt, new ArrayList<>());
        for (Text w : values) {
            words.add(w.toString());
        }
        countMap.put(cnt, words);
    }

    @Override
    protected void cleanup(Context context)
        throws IOException, InterruptedException {

        // collect top 10 word→count pairs
        List<WordCount> topList = new ArrayList<>();
        int seen = 0;
        for (Map.Entry<Integer, List<String>> entry : countMap.entrySet()) {
            int cnt = entry.getKey();
            for (String w : entry.getValue()) {
                topList.add(new WordCount(w, cnt));
                seen++;
                if (seen == 10) break;
            }
            if (seen == 10) break;
        }

        // sort these 10 entries alphabetically by word
        Collections.sort(topList, (a, b) -> a.word.compareTo(b.word));

        // emit final top 10 in alphabetical order
        for (WordCount wc : topList) {
            context.write(new Text(wc.word), new IntWritable(wc.count));
        }
    }

    // helper class
    private static class WordCount {
        String word;
        int count;
        WordCount(String w, int c) { word = w; count = c; }
    }
}
```



```
2025-04-29 15:32:09,761 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager
 at /0.0.0.0:8032
2025-04-29 15:32:09,829 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager
 at /0.0.0.0:8032
2025-04-29 15:32:09,918 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not pe
rformed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2025-04-29 15:32:09,944 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/h
adoop-yarn/staging/hadoop/.staging/job_1745919848818_0003
2025-04-29 15:32:10,138 INFO mapred.FileInputFormat: Total input files to process : 1
2025-04-29 15:32:10,227 INFO mapreduce.JobSubmitter: number of splits:2
2025-04-29 15:32:10,318 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1745919848818_000
3
2025-04-29 15:32:10,318 INFO mapreduce.JobSubmitter: Executing with tokens: []
2025-04-29 15:32:10,405 INFO conf.Configuration: resource-types.xml not found
2025-04-29 15:32:10,405 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2025-04-29 15:32:10,556 INFO impl.YarnClientImpl: Submitted application application_1745919848818_000
3
2025-04-29 15:32:10,574 INFO mapreduce.Job: The url to track the job: http://bmscecse-HP-Elite-Tower-
800-G9-Desktop-PC:8088/proxy/application_1745919848818_0003/
2025-04-29 15:32:10,575 INFO mapreduce.Job: Running job: job_1745919848818_0003
2025-04-29 15:32:15,652 INFO mapreduce.Job: Job job_1745919848818_0003 running in uber mode : false
2025-04-29 15:32:15,654 INFO mapreduce.Job:  map 0% reduce 0%
2025-04-29 15:32:18,772 INFO mapreduce.Job:  map 100% reduce 0%
2025-04-29 15:32:22,799 INFO mapreduce.Job:  map 100% reduce 100%
2025-04-29 15:32:23,824 INFO mapreduce.Job: Job job_1745919848818_0003 completed successfully
2025-04-29 15:32:23,882 INFO mapreduce.Job: Counters: 54
        File System Counters
                FILE: Number of bytes read=215
                FILE: Number of bytes written=829242
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=306
                HDFS: Number of bytes written=69
                HDFS: Number of read operations=11
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
                HDFS: Number of bytes read erasure-coded=0
        Job Counters
                Launched map tasks=2
                Launched reduce tasks=1
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=2555
                Total time spent by all reduces in occupied slots (ms)=1281
                Total time spent by all map tasks (ms)=2555
                Total time spent by all reduce tasks (ms)=1281
                Total vcore-milliseconds taken by all map tasks=2555
                Total vcore-milliseconds taken by all reduce tasks=1281
                Total megabyte-milliseconds taken by all map tasks=2616320
                Total megabyte-milliseconds taken by all reduce tasks=1311744
```



```
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -ls /rgs/output
Found 2 items
-rw-r--r--   1 hadoop supergroup          0 2025-04-29 15:32 /rgs/output/_SUCCESS
-rw-r--r--   1 hadoop supergroup         69 2025-04-29 15:32 /rgs/output/part-00000
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ hadoop fs -cat /rgs/output/part-00000
are     1
brother 1
family  1
hi      1
how     5
is      4
job     1
sister  1
you     1
your    4
hadoop@bmscecse-HP-Elite-Tower-800-G9-Desktop-PC:~$ 
```