

**Exp .No : 9**

**Date :**

## **DEMONSTRATE THE MAP REDUCE PROGRAMMING MODEL BY COUNTING THE NUMBER OF WORDS IN A FILE**

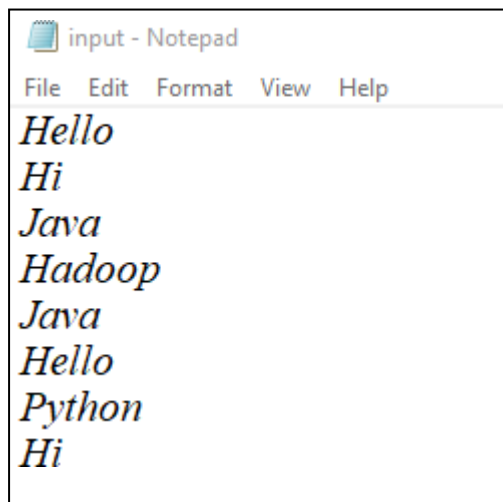
### **AIM:**

To demonstrate the MAP REDUCE programming model for counting the number of words in a file.

### **PROCEDURE:**

#### **Step 1: Create Data File:**

Create a file named "input.txt" and populate it with text data that you wish to analyse.



#### **Step 2: Mapper Logic - mapper.py:**

Create a file named "mapper.py" to implement the logic for the mapper. The mapper will read input data from STDIN, split lines into words, and output each word with its count.

##### **mapper.py:**

```
#!/C:/Users/user/AppData/Local/Microsoft/WindowsApps/python.exe
import sys
for line in sys.stdin:
    line = line.strip()
    words = line.split()
    for word in words:
        print('%s\t%s'%(word,1))
```

#### **Step 3: Reducer Logic - reducer.py:**

Create a file named "reducer.py" to implement the logic for the reducer. The reducer will aggregate the occurrences of each word and generate the final output.

##### **reducer.py:**

```
#!/C:/Users/user/AppData/Local/Microsoft/WindowsApps/python.exe
import sys
prev_word = None
prev_count = 0
for line in sys.stdin:
```

```

line = line.strip()
word, count = line.split('\t')
count = int(count)
if prev_word == word:
    prev_count += count
else:
    if prev_word:
        print('%s\t%s' %(prev_word, prev_count))
    prev_count = count
    prev_word = word
if prev_word == word:
    print('%s\t%s' %(prev_word, prev_count))

```

### Step 4: Prepare Hadoop Environment:

Start the Hadoop daemons and create a directory in HDFS to store your data. Run the following commands to store the data in the WordCount Directory.

```
start-all.cmd
cd C:/Hadoop/sbin
hdfs dfs -mkdir /WordCount
hdfs dfs -put C:/Users/user/Documents/DataAnalytics/input.txt /WordCount
hadoop jar C:\hadoop\share\hadoop\tools\lib\hadoop-streaming-3.3.6.jar ^
-input /WordCount/input.txt ^
-output /WordCount/output ^
-mapper "python C:/Users/user/Documents/DataAnalytics/mapper.py" ^
-reducer "python C:/Users/user/Documents/DataAnalytics/reducer.py"
```

### Step 5: Check Output:

Check the output of the Word Count program in the specified HDFS output directory.

```
hdfs dfs -cat /WordCount/output/part-00000
```

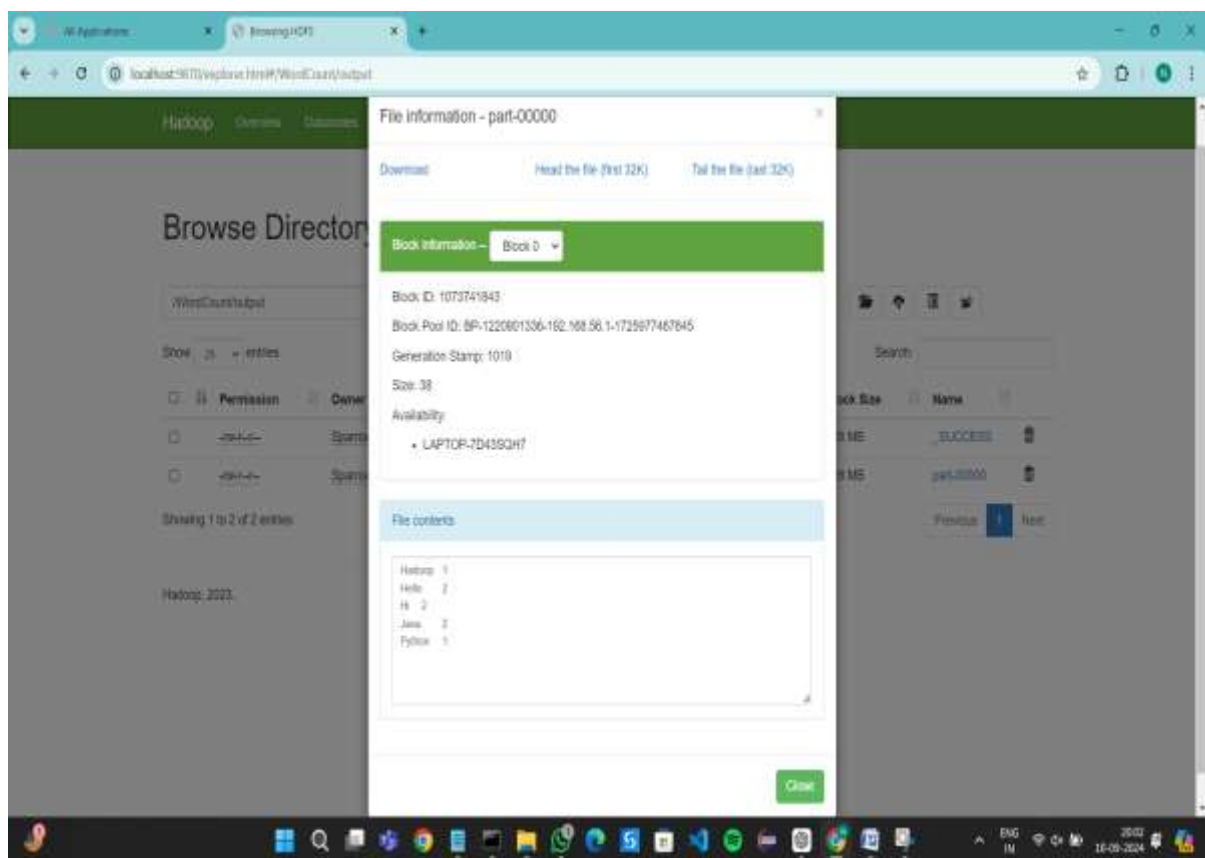
**OUTPUT:**

[illegible]

```
Administrator Command Window
Job: WordCount
Submitted map tasks: 2
Submitted reduce tasks: 1
Data-local map tasks: 0
Total time spent by all maps in occupied slots (ms): 27832
Total time spent by all reducers in occupied slots (ms): 14698
Total time spent by all map tasks (ms): 27832
Total time spent by all reduce tasks (ms): 14698
Total map-milliseconds spent by all map tasks: 27832
Total reduce-milliseconds spent by all reduce tasks: 14698
Total map-bytes-milliseconds spent by all map tasks: 25481728
Total reduce-bytes-milliseconds spent by all reduce tasks: 18576728

Map-Reduce Framework
Map input records: 2
Map output records: 1
Map output bytes: 0
Map output materialized bytes: 71
Input split bytes: 128
Combine input records: 0
Combine output records: 0
Reduce input groups: 1
Reduce shuffle bytes: 71
Reduce input records: 1
Reduce output records: 0
Spilled Records: 1
Spilled Bytes: 2
Failed Shuffles: 0
Merged Map outputs: 0
GC time elapsed (ms): 147
CPU time spent (ms): 1486
Physical memory (bytes) allocated: 4454400
Virtual memory (bytes) allocated: 147838420
Total committed heap usage (bytes): 19377880
Peak Map Physical memory (bytes): 34287680
Peak Map Virtual memory (bytes): 49388184
Peak Reduce Physical memory (bytes): 29020800
Peak Reduce Virtual memory (bytes): 45167504

Shuffle Count
GC 10-0
COMMIT 0ms
IO 1000-0
WRITE 1000-0
WRITE 1000-0
WRITE 1000-0
File Input Format Counters
Bytes Read: 0
File Output Format Counters
Bytes Written: 0
2024-09-09 00:15:17,588 INFO org.apache.hadoop.mapreduce.lib.output.FileOutputFormat: Output directory: /hadoop-3.3.9/output/
C:\Hadoop-3.3.9\bin>
```



## RESULT:

Thus, the program for basic Word Count Map Reduce has been executed successfully.