

**Exp .No : 9****Date :**

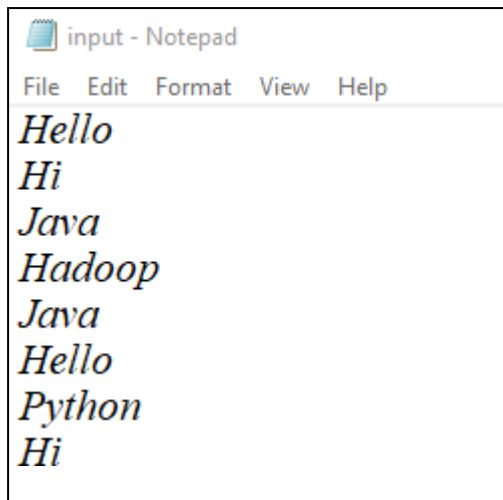
## **DEMONSTRATE THE MAP REDUCE PROGRAMMING MODEL BY COUNTING THE NUMBER OF WORDS IN A FILE**

**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
```

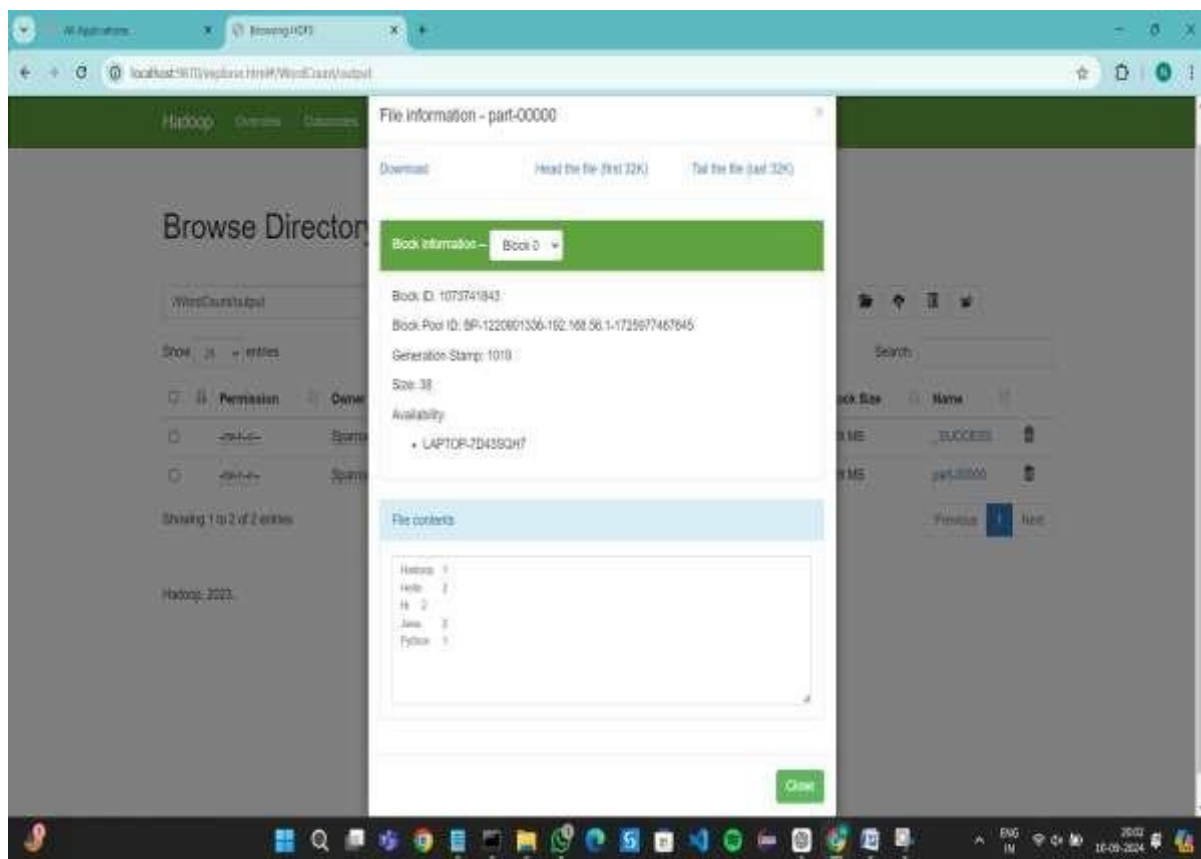
**OUTPUT:**

[illegible]

```

AdmireOne Connect Stage
Job Variables
  Submitted map tasks=0
  Submitted reduce tasks=1
  Data-local map tasks=0
  Total time spent by all maps in occupied slots (ms)=37632
  Total time spent by all reducers in occupied slots (ms)=14096
  Total time spent by all map tasks (ms)=37632
  Total time spent by all reduce tasks (ms)=14096
  Total write milliseconds taken by all map tasks=19920
  Total write milliseconds taken by all reduce tasks=61680
  Total megabyte-milliseconds taken by all map tasks=2448720
  Total megabyte-milliseconds taken by all reduce tasks=14676720
Map-Reduce Summary
  Map Input Records=1
  Map Output Records=0
  Map Output Bytes=0
  Map Output Materialized Bytes=0
  Input Split Bytes=128
  Combine Input Records=0
  Combine Output Records=0
  Reduce Input Groups=1
  Reduce Shuffle Bytes=0
  Reduce Input Records=0
  Reduce Output Records=0
  Spilled Records=0
  Spilled Bytes=0
  Failed Shuffle=0
  Report Map Output=0
  GC Time (ms)=107
  CPU time spent (ms)=180
  Physical memory (bytes) committed=94976000
  Virtual memory (bytes) committed=147038400
  Total committed heap usage (bytes)=51177600
  Peak Map Physical memory (bytes)=34438400
  Peak Map Virtual memory (bytes)=66688128
  Peak Reduce Physical memory (bytes)=20028800
  Peak Reduce Virtual memory (bytes)=34367700
Shuffle Errors
  SOF=0
  CORRUPTED=0
  TO_RETRY=0
  WRONG_LENGTH=0
  WRONG_SIZE=0
  Wrong offset=0
File Input Format Counters
  Bytes Read=0
File Output Format Counters
  Bytes Written=0
2016-06-01 01:51:57,088 INFO chomping.Shredder: Output directory: /dev/shm/hadoop

```



**RESULT:**

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