

**EX.NO. 9**

**Roll no: 210701272**

**HADOOP DEMONSTRATE THE MAP REDUCE PROGRAMMING  
MODEL BYCOUNTING 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** - Open Terminal

```
$ su hduser
```

Password:

**Step 2** - Start dfs and mapreduce services

```
$ cd /usr/local/hadoop/hadoop-2.7.2/sbin
```

```
$ start-dfs.sh
```

```
$ start-yarn.sh
```

```
$ jps
```

**Step 3** - Check Hadoop through web UI

// Go to browser type <http://localhost:8088> – All Applications Hadoop Cluster

// Go to browser type <http://localhost:50070> – Hadoop Namenode

**Step 4** – Open New Terminal

```
$ cd Desktop/
```

```
$ mkdir inputdata
```

```
$ cd inputdata/
```

```
$ echo "Java Dart Java Hello World" >>input.txt
```

```
$ cat>> input.txt
```

**Step 5** – Go back to old Terminal

```
$ hadoop fs -copyFromLocal /home/hduser/Desktop/inputdata/input.txt
```

/folder/hduser // Check in input.txt in Namenode using Web UI

## Step 6 – WordCount Program

- Mapper.py
- Reducer.py

### Mapper.py

```
#!/C:/ProgramData/chocolatey/bin/python3.exe
```

```
import sys
```

```
for line in sys.stdin:
```

```
    line = line.strip()
```

```
    words = line.split()
```

```
    for word in words:
```

```
        print('%s\t%s' % (word, 1))
```

### Reducer.py

```
#!/C:/ProgramData/chocolatey/bin/python3.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))
```

## OUTPUT:

```
C:\>hadoop
Usage: hadoop [--config confdir] [--loglevel loglevel] COMMAND
where COMMAND is one of:
  fs                run a generic filesystem user client
  version           print the version
  jar <jar>         run a jar file
                   note: please use "yarn jar" to launch
                   YARN applications, not this command.
  checknative [-a|-h] check native hadoop and compression libraries availability
  conftest         validate configuration XML files
  distch path:owner:group:permission distributed metadata changer
  distcp <srcurl> <desturl> copy file or directories recursively
  archive -archiveName NAME -p <parent path> <src>* <dest> create a hadoop archive
  classpath        prints the class path needed to get the
                   Hadoop jar and the required libraries
  credential       interact with credential providers
  jnipath          prints the java.library.path
  kerbname         show auth_to_local principal conversion
  kdiag           diagnose kerberos problems
  key             manage keys via the KeyProvider
  trace           view and modify Hadoop tracing settings
  daemonlog       get/set the log level for each daemon
  or
  CLASSNAME       run the class named CLASSNAME

Most commands print help when invoked w/o parameters.
```

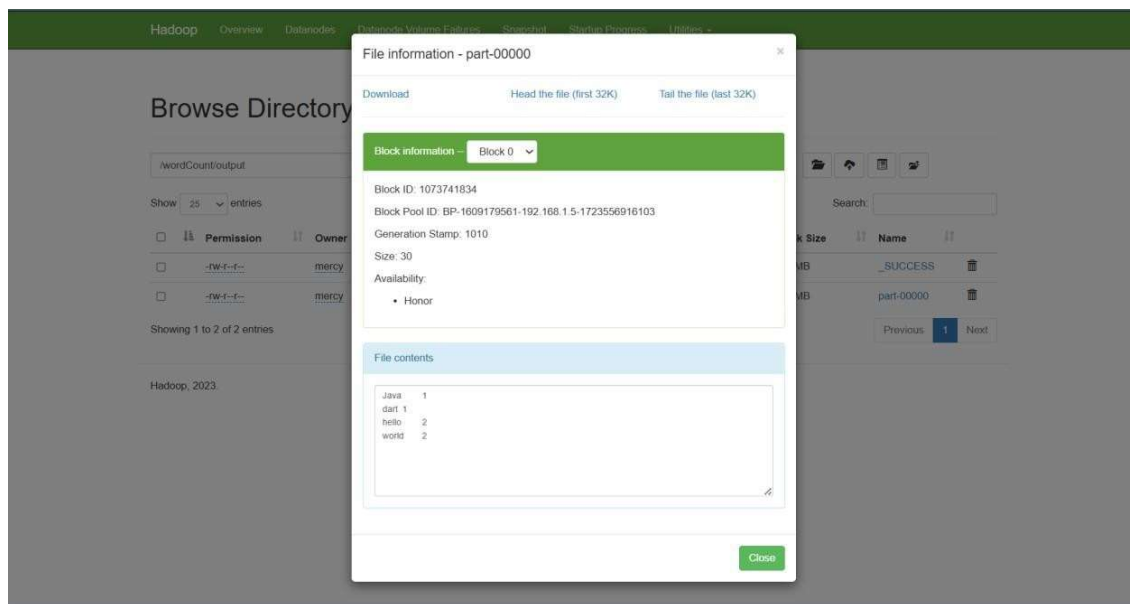
```
C:\>hadoop version
Hadoop 3.3.6
Source code repository https://github.com/apache/hadoop.git -r 1be78238728da9266a4f88195058f08fd012bf9c
Compiled by ubuntu on 2023-06-18T08:22Z
Compiled on platform linux-x86_64
Compiled with protoc 3.7.1
From source with checksum 5652179ad55f76cb287d9c633bb53bbd
This command was run using /C:/hadoop-3.3.6/share/hadoop/common/hadoop-common-3.3.6.jar
```

```
C:\>start-all.cmd
This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd
starting yarn daemons
```

```
C:\>jps
19572 ResourceManager
19972 NodeManager
7028 NameNode
360 Jps
15628 Eclipse
19468 DataNode
```

```
C:\>hadoop fs -cat /wordCount/output/part-00000
Java      1
dart      1
hello     2
world     2

C:\>
```



## RESULT:

Thus the implementation of the python mapper and reducer programs using MapReduce to count the words in a text file using Hadoop is executed successfully.