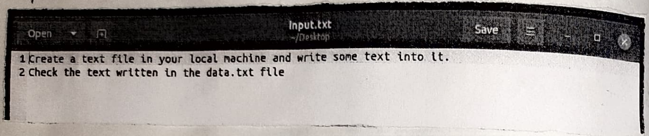
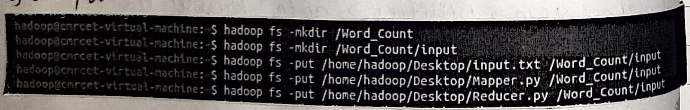
Input file - inside text


input.txt
~/Desktop

```
1 Create a text file in your local machine and write some text into it.
2 Check the text written in the data.txt file
```

a) output



```
hadoop@cmrcet-virtual-machine:~$ hadoop fs -mkdir /Word_Count
hadoop@cmrcet-virtual-machine:~$ hadoop fs -mkdir /Word_Count/input
hadoop@cmrcet-virtual-machine:~$ hadoop fs -put /home/hadoop/Desktop/input.txt /Word_Count/input
hadoop@cmrcet-virtual-machine:~$ hadoop fs -put /home/hadoop/Desktop/Mapper.py /Word_Count/input
hadoop@cmrcet-virtual-machine:~$ hadoop fs -put /home/hadoop/Desktop/Reducer.py /Word_Count/input
```

MapReduce Program.

a) Write a MapReduce program for counting number of words in a given file or Document.
b) Create a jar file &
c) Run the jar file and observe mapper process and reducer process.
d) Read the output file and display the results.

Step 1: Start hadoop
$ start-all.sh

Step 2: Check everything is running or not
$ jps

Step 3: Create a directory to hadoop fs to store input file:
$ hadoop fs -mkdir /Word_Count
$ hadoop fs -mkdir /Word_Count/input.

Step 4: Now take your input file and put inside hadoop fs:
$ hadoop fs -put /home/hadoop/Desktop/input.txt /Word_count/input

Ex. No. | Date | Page No.

Step 5: Now create Mapper.py inside your fs:
Mapper.py which maps input data and outputs key value pair as every word as 1

Code:
```
import sys
for line in sys.stdin:
    line = line.strip()
    words = line.split()
    for word in words:
        print ('%s \t %s' % (word,1))
```

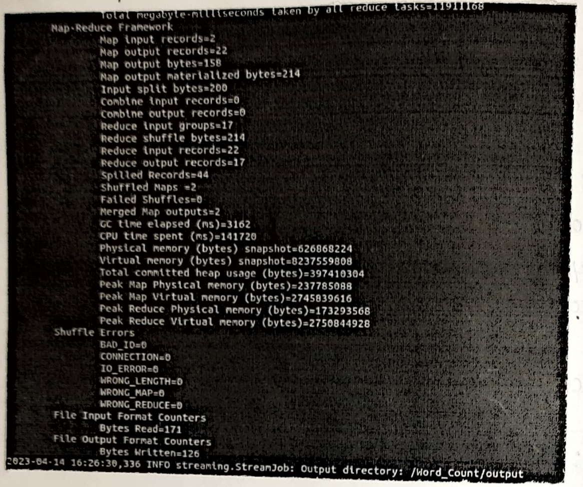
Step 6: Now create Reducer.py inside your fs:
Reducer.py which takes input from mapper and gets count for each word:

code:
```
from operator import itemgetter
import sys
current_word = None
~~current_word~~ current count = 0
word = None
```

Ex. No. | Date | Page No.

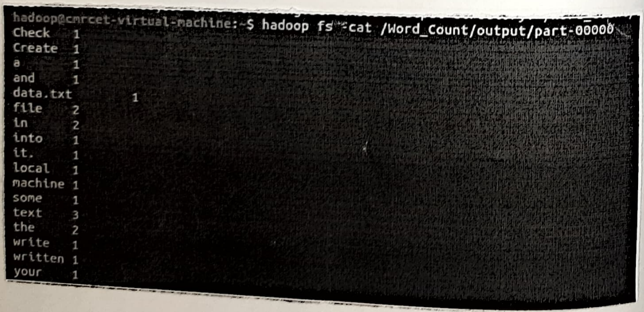
```
for line in sys.stdin:
    line = line.strip()
    word, count = line.split ('\t',1)

    try:
        count = int ( count)
    except ValueError :
        continue

    if current_word == word:
        current_count += count:
    else:
        if current_word:
            print ('%s \t%s' % (current_word, current_count))

        current_count = count
        current_~~count~~ word = word

if current_word == word :
    print ('%s \t%s' % (current_word, current_count))
```

b&c) output



```
            Total megabyte-milliseconds taken by all reduce tasks=11911168
        Map-Reduce Framework
                Map input records=2
                Map output records=22
                Map output bytes=158
                Map output materialized bytes=214
                Input split bytes=200
                Combine input records=0
                Combine output records=0
                Reduce input groups=17
                Reduce shuffle bytes=214
                Reduce input records=22
                Reduce output records=17
                Spilled Records=44
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=3162
                CPU time spent (ms)=141720
                Physical memory (bytes) snapshot=626868224
                Virtual memory (bytes) snapshot=8237559808
                Total committed heap usage (bytes)=397410304
                Peak Map Physical memory (bytes)=237785088
                Peak Map Virtual memory (bytes)=2745839616
                Peak Reduce Physical memory (bytes)=173293568
                Peak Reduce Virtual memory (bytes)=2750844928
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=171
        File Output Format Counters
                Bytes Written=126
2023-04-14 16:26:30,336 INFO streaming.StreamJob: Output directory: /Word_Count/output
```

d) output



```
hadoop@cmrcet-virtual-machine:~$ hadoop fs -cat /Word_Count/output/part-00000
Check   1
Create  1
a       1
and     1
data.txt        1
file    2
in      2
into    1
it.     1
local   1
machine 1
some    1
text    3
the     2
write   1
written 1
your    1
```

Ex. No. | Date | Page No.

Step 7: Execute command

```
$ hadoop jar /home/hadoop/hadoop/share/hadoop/tools/lib
/hadoop-streaming-3.3.4.jar -file /home/hadoop/Desktop/
Mapper.py -mapper 'pytho3 Mapper.py' -file /home/
hadoop/Desktop/Reducer.py -reducer 'python3 Reducer.py'
-input /Word_count/input/input.txt -output /Word_count/
output.
```

Step 8: To view output

```
$ hadoop fs -cat /Word_Count/output/part-00000
```