

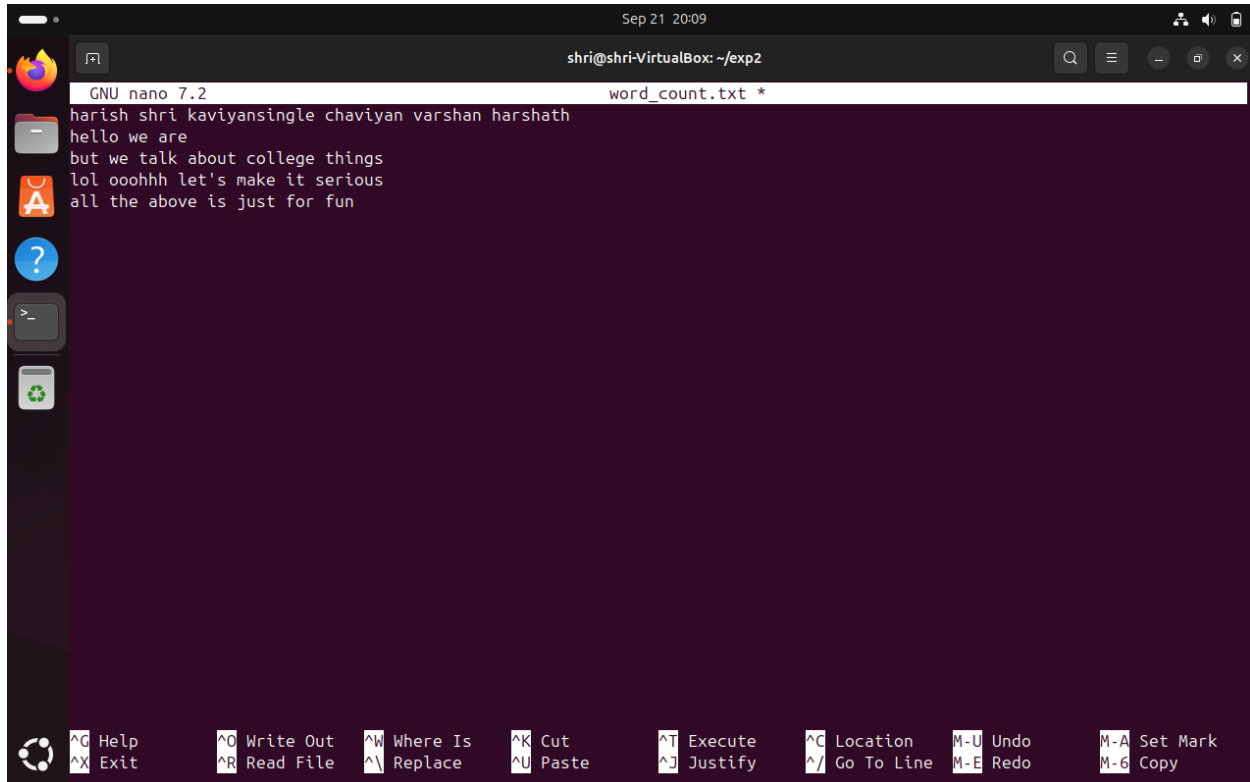
Exp.No: 2**Run a basic Word Count Map Reduce program to understand Map Reduce Paradigm****AIM:**

To run a basic Word Count MapReduce program.

PROCEDURE:

Step 1: Create Data File: Create a file named "word_count_data.txt" and populate it with text data that you wish to analyse. Login with your hadoop user.

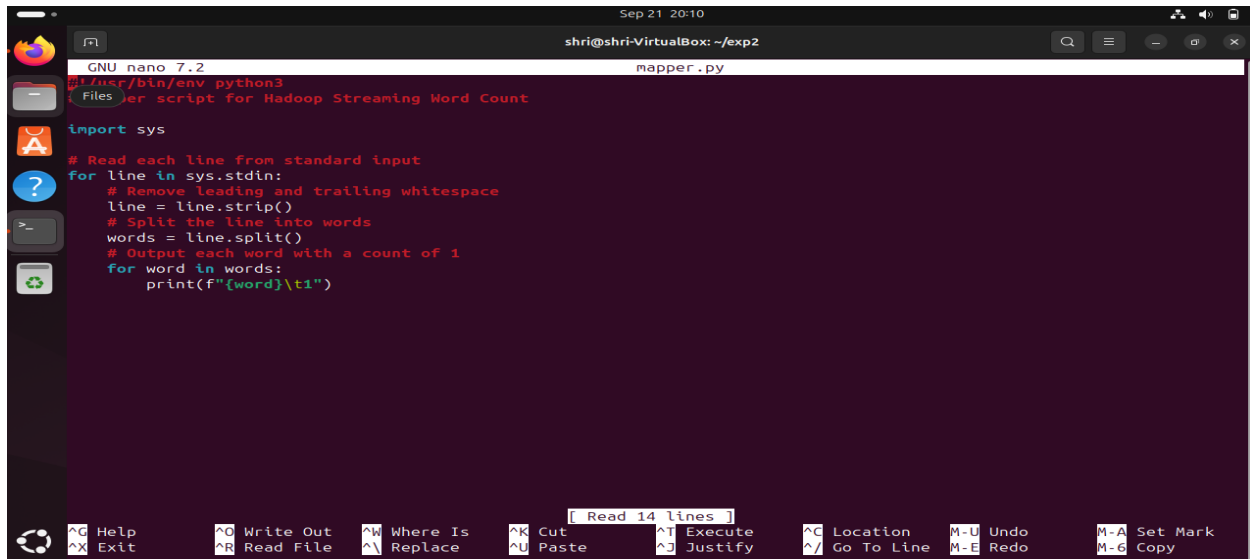
nano word_count.txt



```
shri@shri-VirtualBox: ~/exp2
GNU nano 7.2 word_count.txt *
harish shri kaviyansingle chaviyan varshan harshath
hello we are
but we talk about college things
lol ooohhh let's make it serious
all the above is just for fun
```

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.

nano mapper.py



```

GNU nano 7.2 mapper.py
#!/usr/bin/env python3
# Reducer script for Hadoop Streaming Word Count

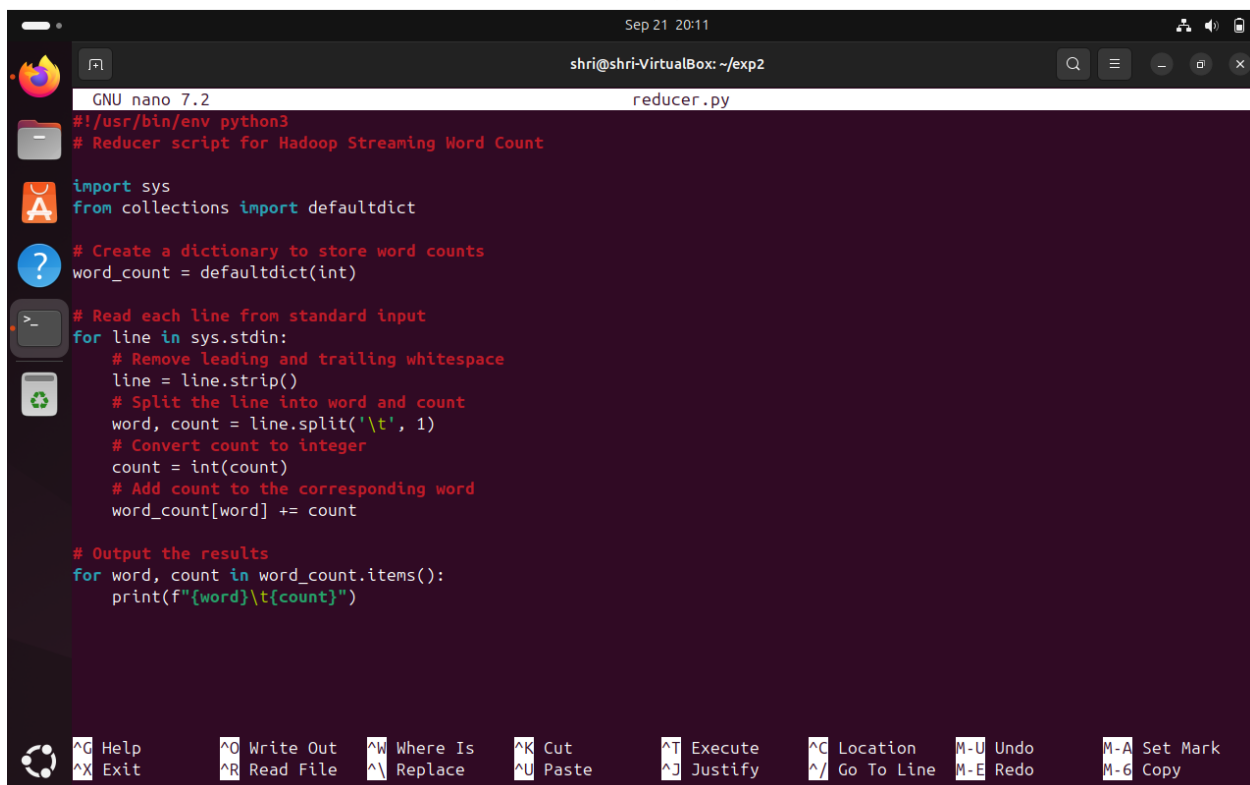
import sys

# Read each line from standard input
for line in sys.stdin:
    # Remove leading and trailing whitespace
    line = line.strip()
    # Split the line into words
    words = line.split()
    # Output each word with a count of 1
    for word in words:
        print(f"{word}\t1")

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo     M-A Set Mark
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/_ Go To Line M-E Redo     M-6 Copy
  
```

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.

nano reducer.py



```

GNU nano 7.2 reducer.py
#!/usr/bin/env python3
# Reducer script for Hadoop Streaming Word Count

import sys
from collections import defaultdict

# Create a dictionary to store word counts
word_count = defaultdict(int)

# Read each line from standard input
for line in sys.stdin:
    # Remove leading and trailing whitespace
    line = line.strip()
    # Split the line into word and count
    word, count = line.split('\t', 1)
    # Convert count to integer
    count = int(count)
    # Add count to the corresponding word
    word_count[word] += count

# Output the results
for word, count in word_count.items():
    print(f"{word}\t{count}")

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo     M-A Set Mark
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/_ Go To Line M-E Redo     M-6 Copy
  
```

Step 4: Prepare Hadoop Environment: Start the Hadoop daemons and create a directory in HDFS to store your data.

```
start-all.sh
```

```
hdfs dfs -mkdir /word_count_in_python
```

```
hdfs dfs -copyFromLocal /path/to/word_count.txt /word_count_in_python
```

Step 5: Make Python Files Executable: Give executable permissions to your mapper.py and reducer.py files.

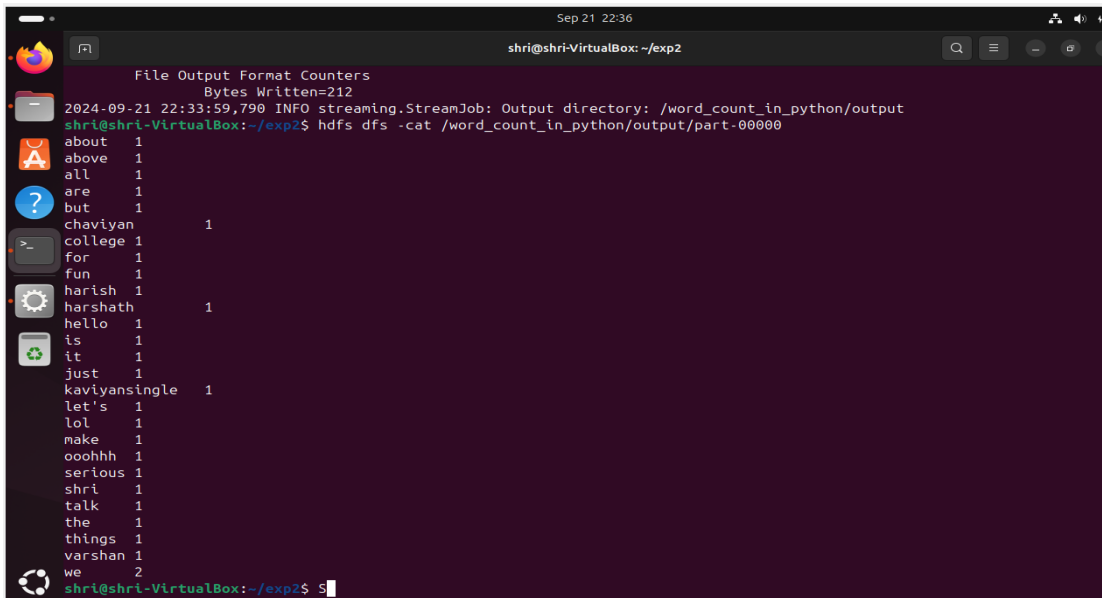
```
chmod 777 mapper.py reducer.py
```

Step 6: Run Word Count using Hadoop Streaming: Download the latest hadoop-streaming jar file and place it in a location you can easily access. Then run the Word Count program using Hadoop Streaming.

```
hadoop jar /path/to/hadoop-streaming-3.3.6.jar \ -input  
/word_count_in_python/word_count.txt \ -output  
/word_count_in_python/output \ -mapper /path/to/mapper.py \ -  
reducer /path/to/reducer.py
```

Step 7: Check Output: Check the output of the Word Count program in the specified HDFS output directory.

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



```
File Output Format Counters
  Bytes Written=212
2024-09-21 22:33:59,790 INFO streaming.StreamJob: Output directory: /word_count_in_python/output
shri@shri-VirtualBox: ~/exp2$ hdfs dfs -cat /word_count_in_python/output/part-00000
about 1
above 1
all 1
are 1
but 1
chaviyan 1
college 1
for 1
fun 1
harish 1
harshath 1
hello 1
is 1
it 1
just 1
kaviyansingle 1
let's 1
lol 1
make 1
ooohhh 1
serious 1
shri 1
talk 1
the 1
things 1
varshan 1
we 2
shri@shri-VirtualBox: ~/exp2$
```

Step-8: Check the Output in the browser.

The screenshot shows a web browser window with the address bar displaying 'localhost:9870/explorer.html#/word_count_in_python/output'. A modal window titled 'File information - part-00000' is open, showing file details and contents.

File information - part-00000

Download Head the file (first 32K) Tail the file (last 32K)

Block information -- Block 0 ▾

Block ID: 1073741878
Block Pool ID: BP-1260272768-127.0.1.1-1725100397044
Generation Stamp: 1054
Size: 212
Availability:
• shri-VirtualBox

File contents

```
about 1
above 1
all 1
are 1
but 1
chaviyan 1
college 1
for 1
```

RESULT:

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