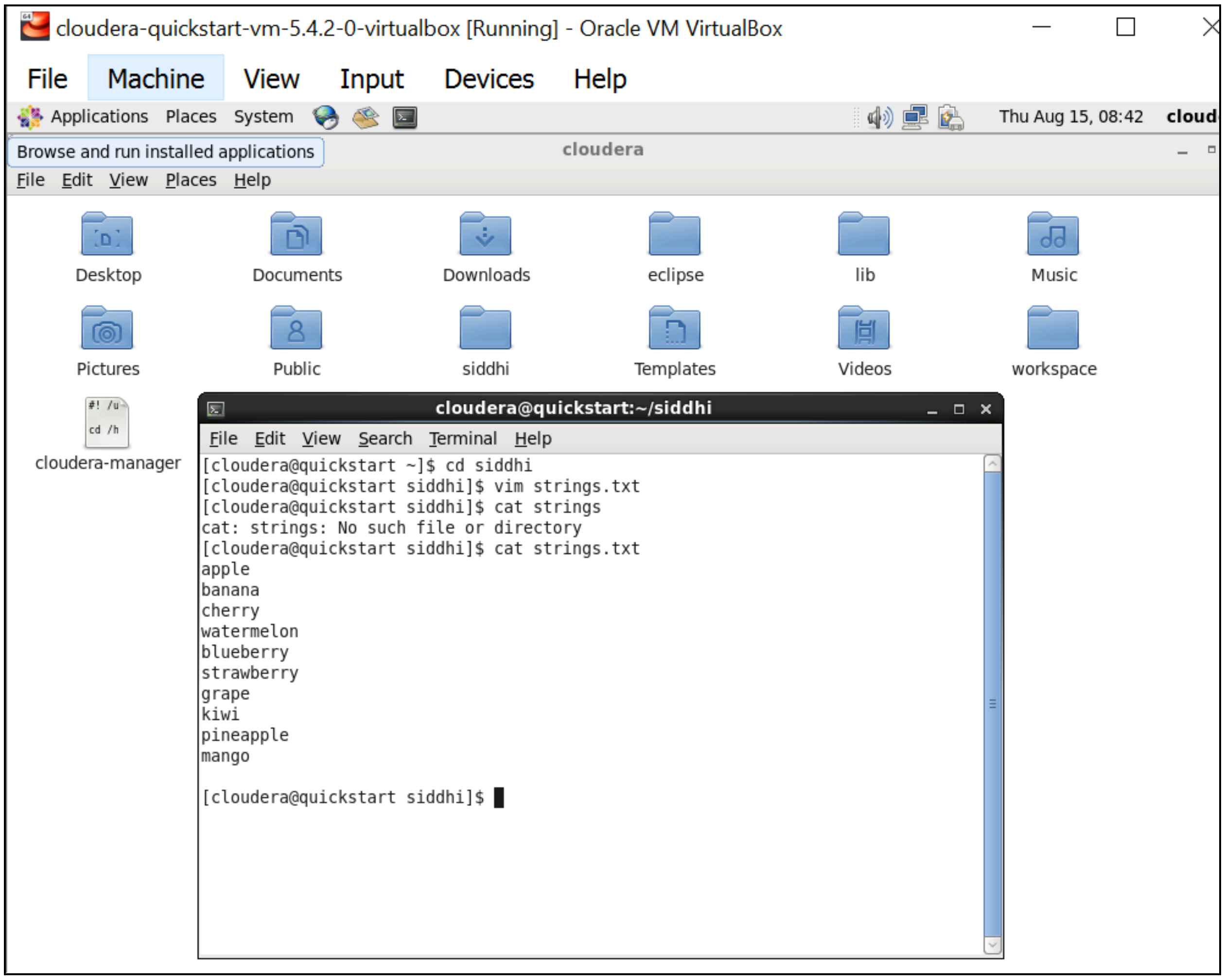
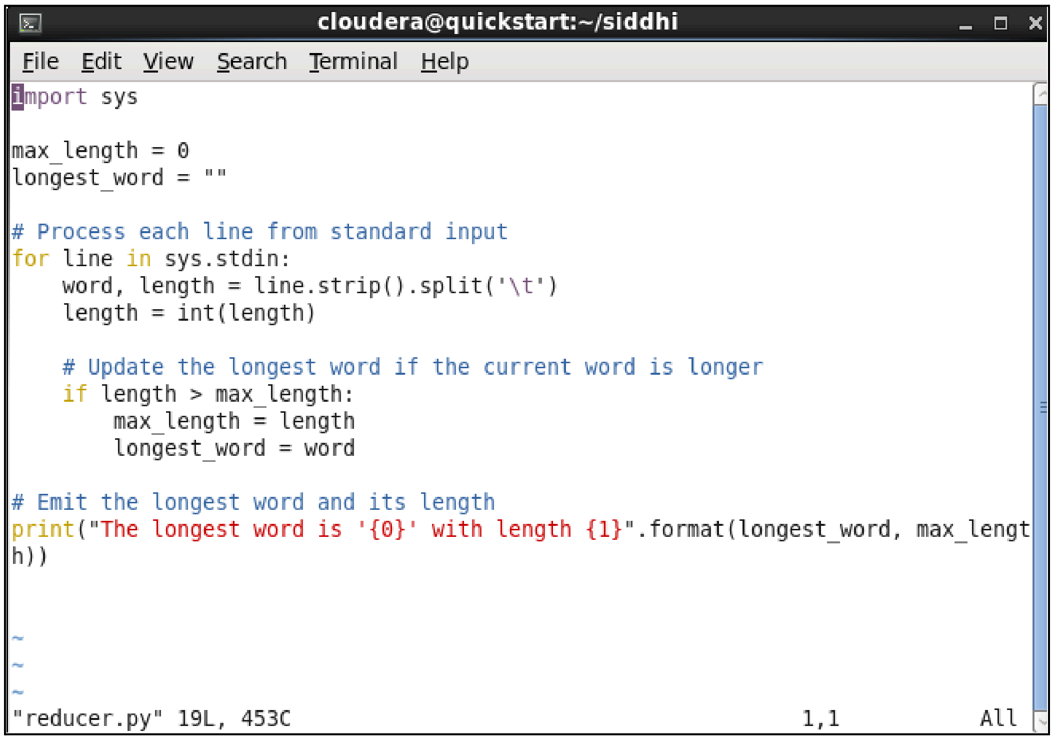
Big Data Analytics and Visualization

Lab 2
MapReduce

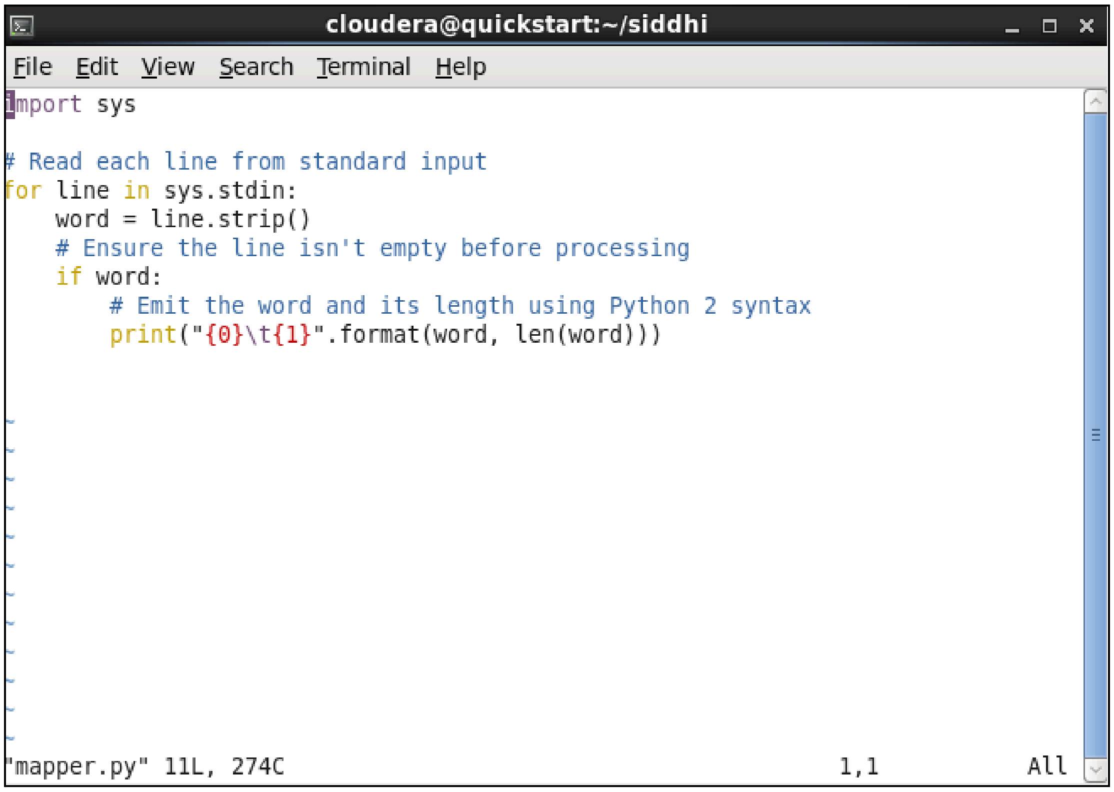
Aim: 1. Write a mapper and reducer to find the longest string in a text file. (strings.txt, create one with at least 10 words)
Output:


cloudera-quickstart-vm-5.4.2-0-virtualbox [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places System
Thu Aug 15, 08:42
Browse and run installed applications
cloudera
File Edit View Places Help
Desktop
Documents
Downloads
eclipse
lib
Music
Pictures
Public
siddhi
Templates
Videos
workspace
cloudera-manager
cloudera@quickstart:~/siddhi
File Edit View Search Terminal Help

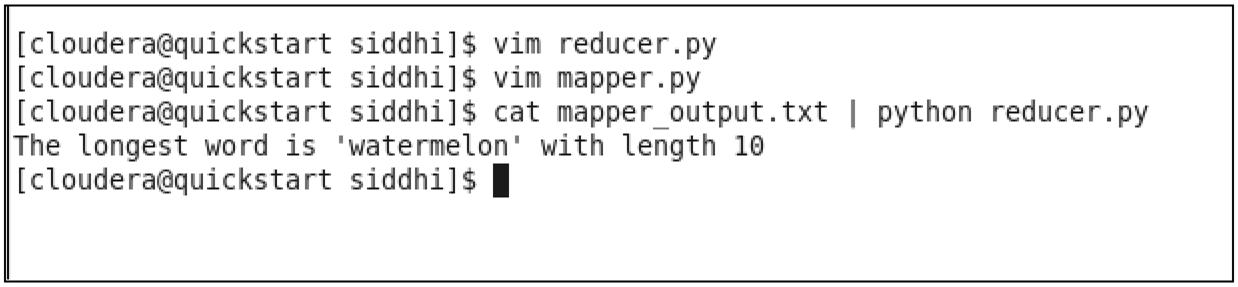
```
[cloudera@quickstart ~]$ cd siddhi
[cloudera@quickstart siddhi]$ vim strings.txt
[cloudera@quickstart siddhi]$ cat strings
cat: strings: No such file or directory
[cloudera@quickstart siddhi]$ cat strings.txt
apple
banana
cherry
watermelon
blueberry
strawberry
grape
kiwi
pineapple
mango

[cloudera@quickstart siddhi]$ 
```


cloudera@quickstart:~/siddhi
File Edit View Search Terminal Help

```
import sys

max_length = 0
longest_word = ""

# Process each line from standard input
for line in sys.stdin:
    word, length = line.strip().split('\t')
    length = int(length)

    # Update the longest word if the current word is longer
    if length > max_length:
        max_length = length
        longest_word = word

# Emit the longest word and its length
print("The longest word is '{0}' with length {1}".format(longest_word, max_lengt
h))



~
~
~
"reducer.py" 19L, 453C                                        1,1           All
```


cloudera@quickstart:~/siddhi
File Edit View Search Terminal Help

```
import sys

# Read each line from standard input
for line in sys.stdin:
    word = line.strip()
    # Ensure the line isn't empty before processing
    if word:
        # Emit the word and its length using Python 2 syntax
        print("{0}\t{1}".format(word, len(word)))



~
~
~
~
~
~
~
~
~
~
~
~
"mapper.py" 11L, 274C                                         1,1           All
```

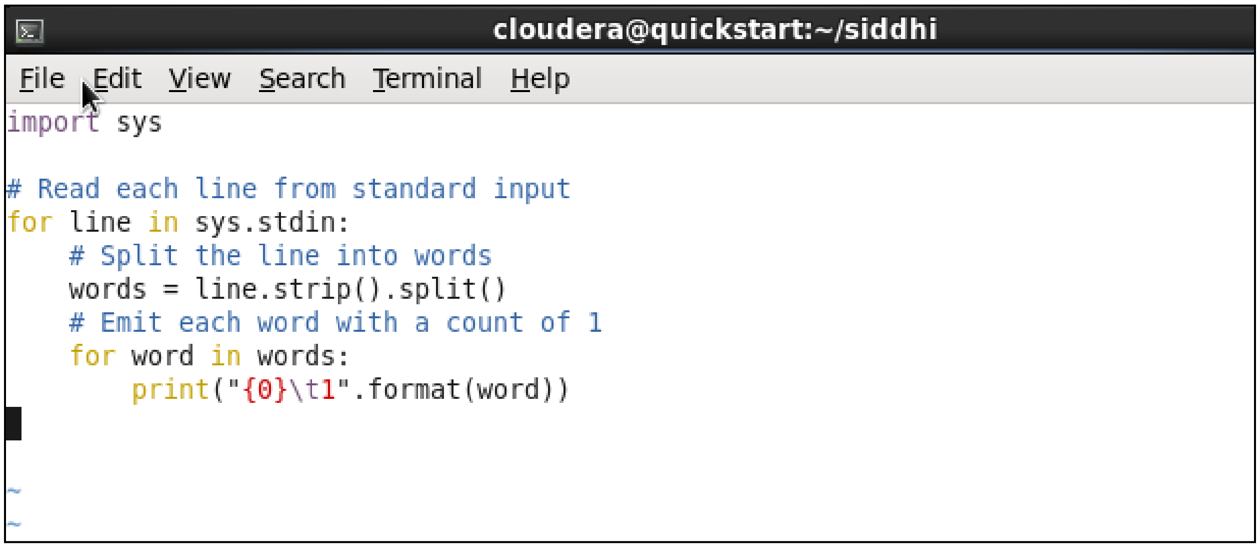


```
[cloudera@quickstart siddhi]$ vim reducer.py
[cloudera@quickstart siddhi]$ vim mapper.py
[cloudera@quickstart siddhi]$ cat mapper_output.txt | python reducer.py
The longest word is 'watermelon' with length 10
[cloudera@quickstart siddhi]$ 
```

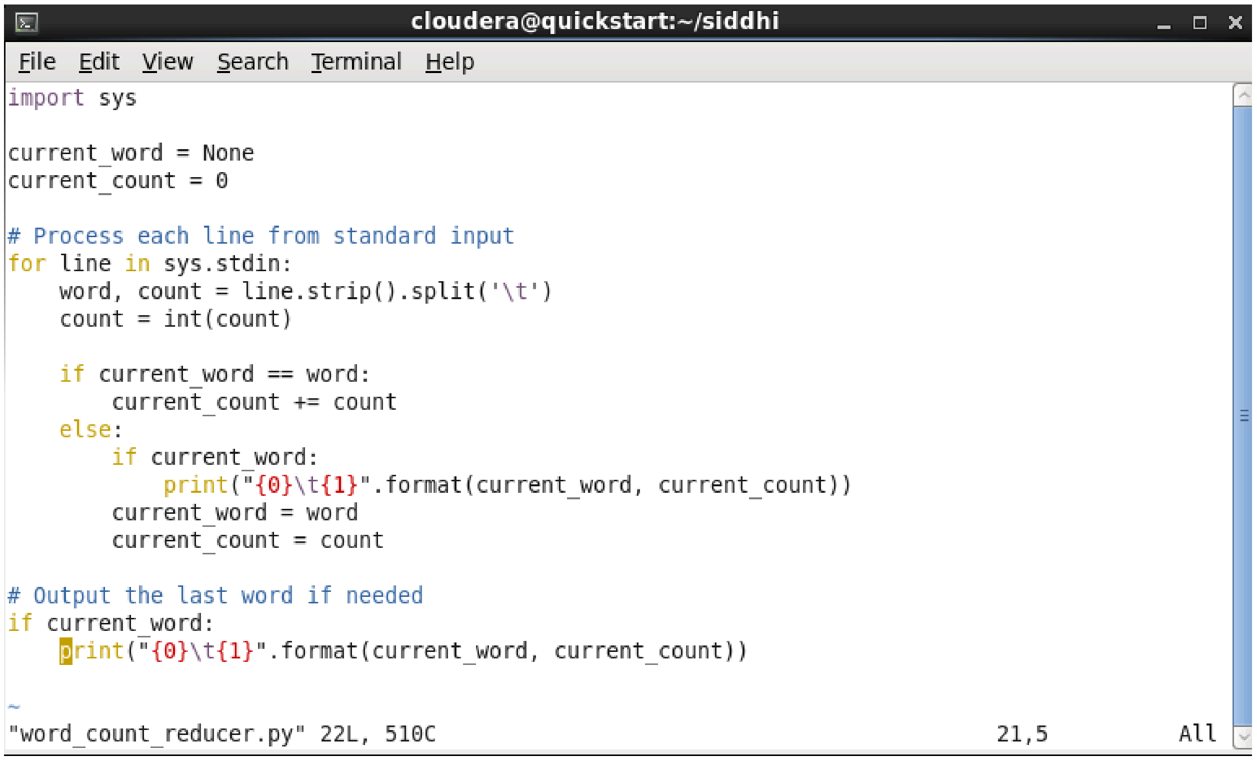
Aim: 2. Write the following Map Reduce program to understand Map Reduce Paradigm. (Create your own .txt file)
a. WordCount
b. Average WordCount
c. Word with minimum count and word with maximum count

Output:

a. WordCount

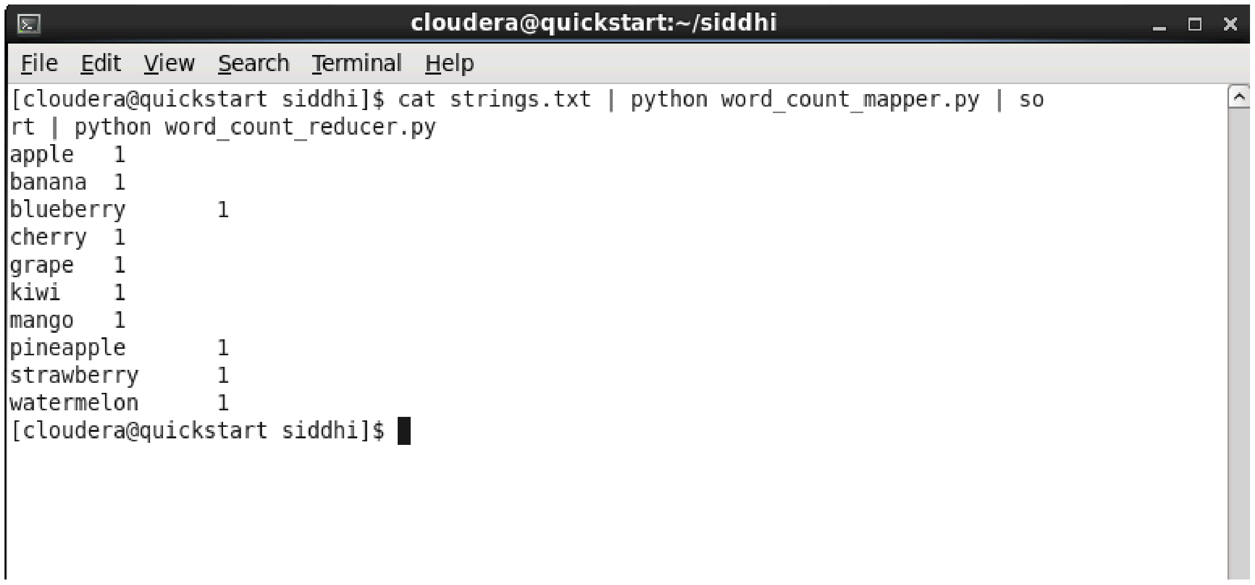

cloudera@quickstart:~/siddhi
File Edit View Search Terminal Help

```
import sys

# Read each line from standard input
for line in sys.stdin:
    # Split the line into words
    words = line.strip().split()
    # Emit each word with a count of 1
    for word in words:
        print("{0}\t1".format(word))

~
~
```


cloudera@quickstart:~/siddhi
File Edit View Search Terminal Help

```
import sys

current_word = None
current_count = 0

# Process each line from standard input
for line in sys.stdin:
    word, count = line.strip().split('\t')
    count = int(count)

    if current_word == word:
        current_count += count
    else:
        if current_word:
            print("{0}\t{1}".format(current_word, current_count))
        current_word = word
        current_count = count

# Output the last word if needed
if current_word:
    print("{0}\t{1}".format(current_word, current_count))

~
"word_count_reducer.py" 22L, 510C                              21,5          All
```

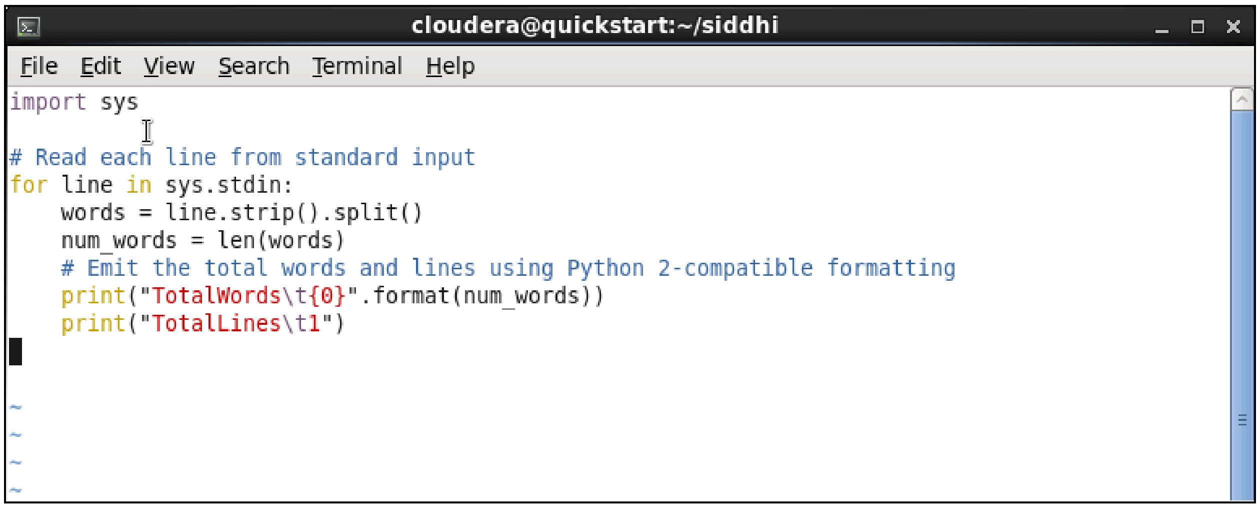
Output:



cloudera@quickstart:~/siddhi
File Edit View Search Terminal Help

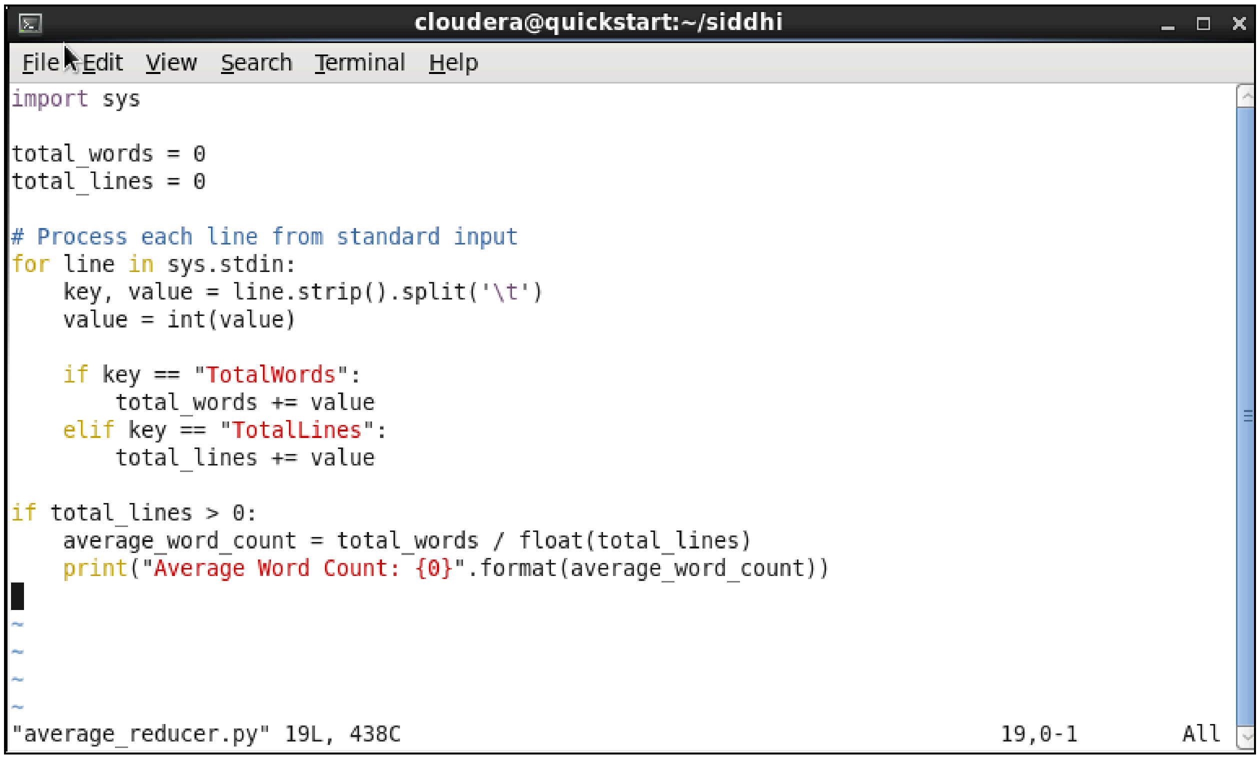
```
[cloudera@quickstart siddhi]$ cat strings.txt | python word_count_mapper.py | so
rt | python word_count_reducer.py
apple   1
banana  1
blueberry       1
cherry  1
grape   1
kiwi    1
mango   1
pineapple       1
strawberry      1
watermelon      1
[cloudera@quickstart siddhi]$
```

b. Average WordCount

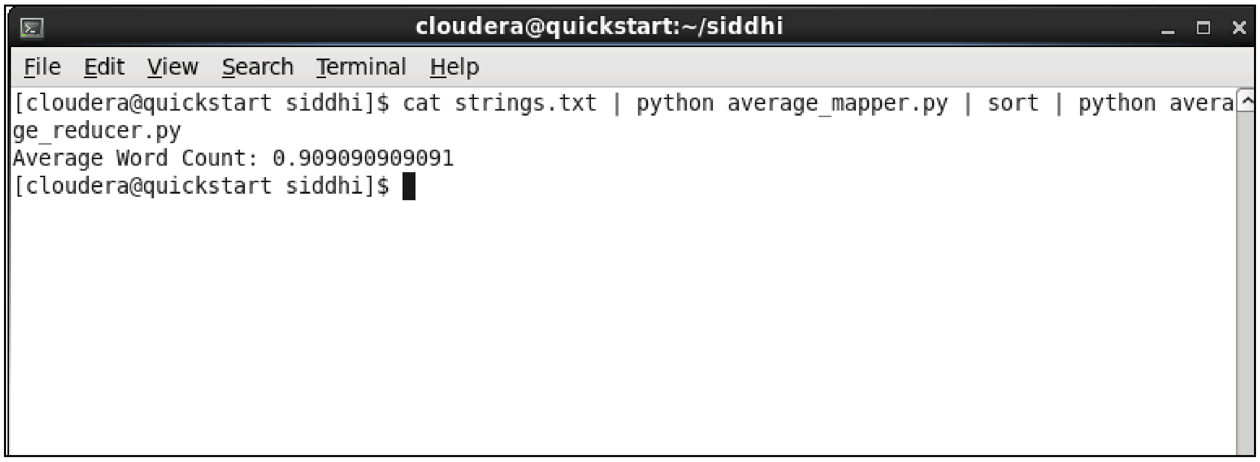


cloudera@quickstart:~/siddhi
File Edit View Search Terminal Help

```
import sys

# Read each line from standard input
for line in sys.stdin:
    words = line.strip().split()
    num_words = len(words)
    # Emit the total words and lines using Python 2-compatible formatting
    print("TotalWords\t{0}".format(num_words))
    print("TotalLines\t1")

~
~
~
~
```

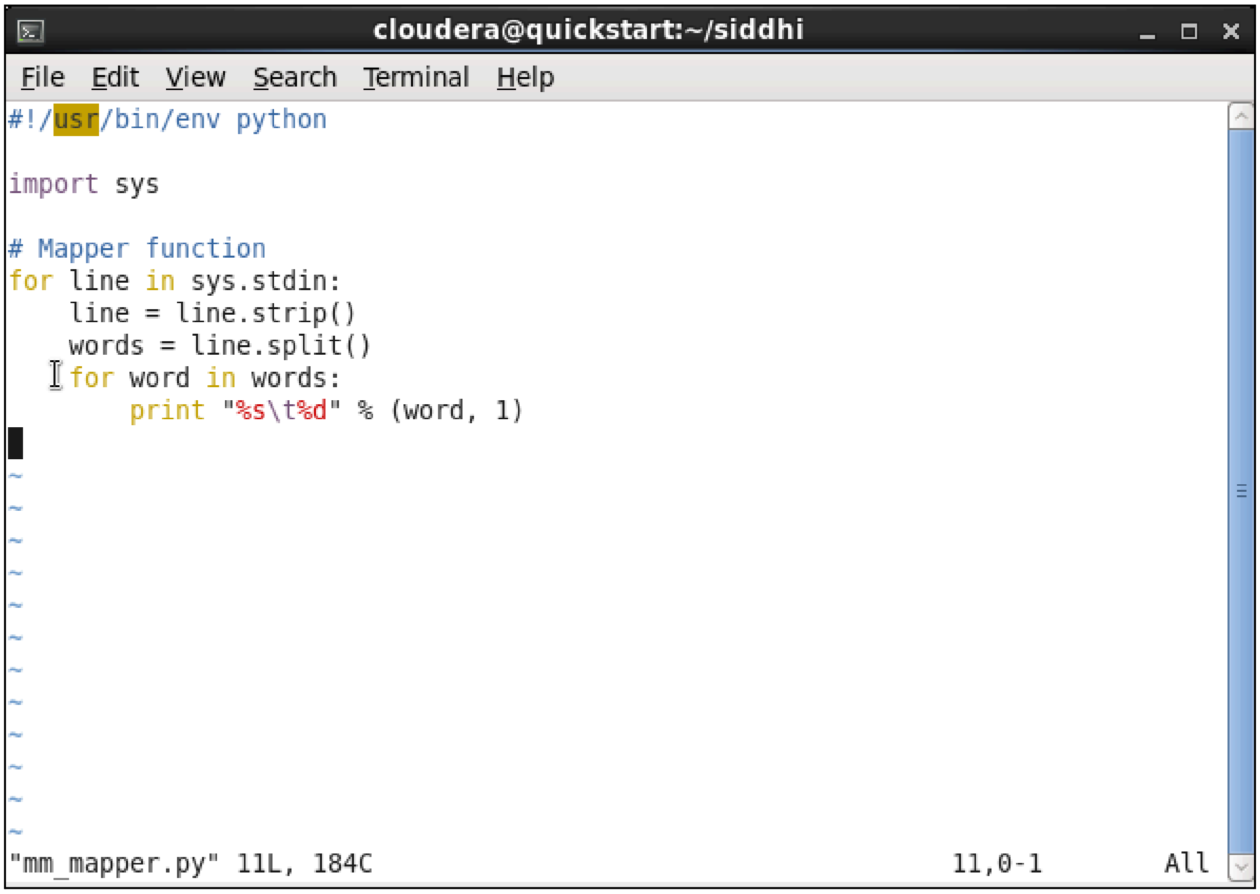

cloudera@quickstart:~/siddhi
File Edit View Search Terminal Help

```
import sys

total_words = 0
total_lines = 0

# Process each line from standard input
for line in sys.stdin:
    key, value = line.strip().split('\t')
    value = int(value)

    if key == "TotalWords":
        total_words += value
    elif key == "TotalLines":
        total_lines += value

if total_lines > 0:
    average_word_count = total_words / float(total_lines)
    print("Average Word Count: {0}".format(average_word_count))

~
~
~
~
"average_reducer.py" 19L, 438C                                    19,0-1        All
```

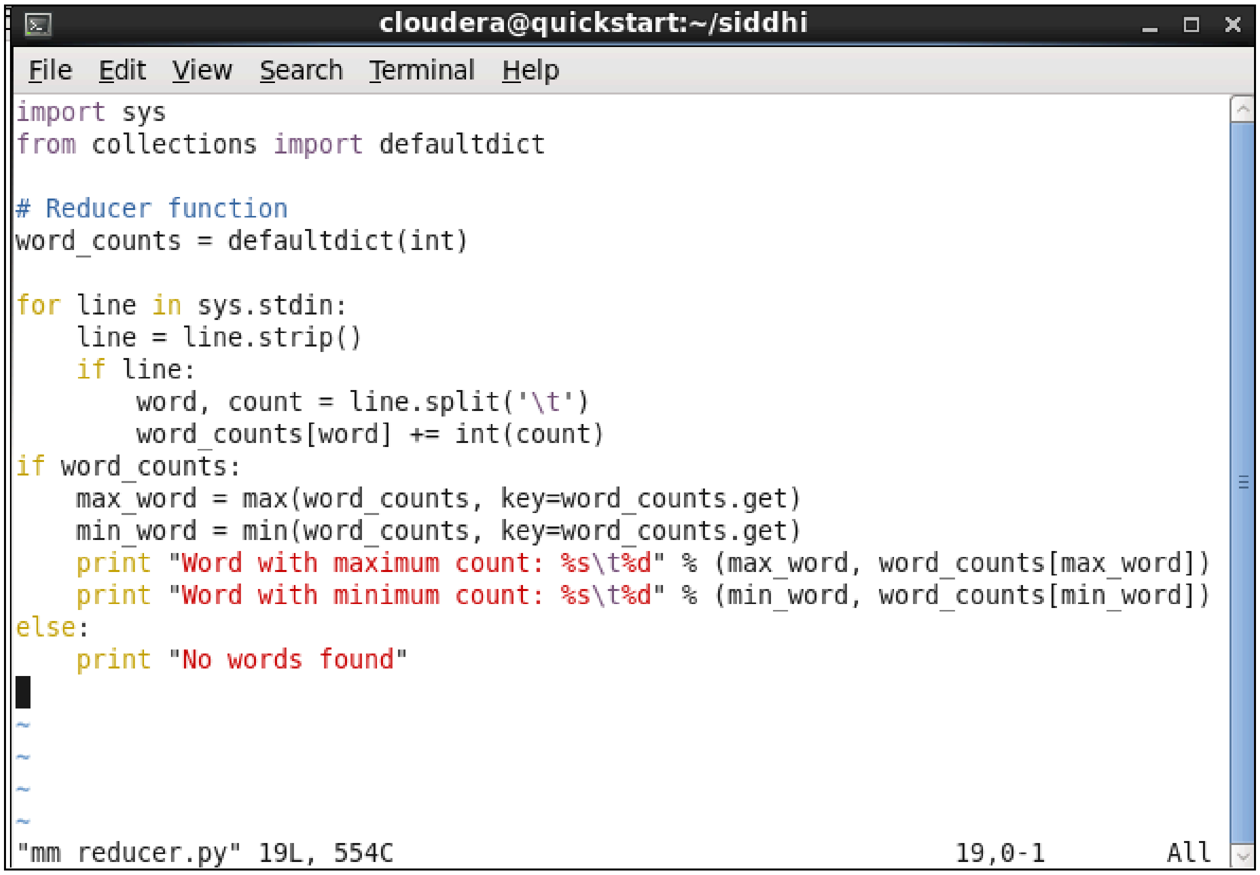

cloudera@quickstart:~/siddhi
File Edit View Search Terminal Help

```
[cloudera@quickstart siddhi]$ cat strings.txt | python average_mapper.py | sort | python avera
ge_reducer.py
Average Word Count: 0.909090909091
[cloudera@quickstart siddhi]$
```

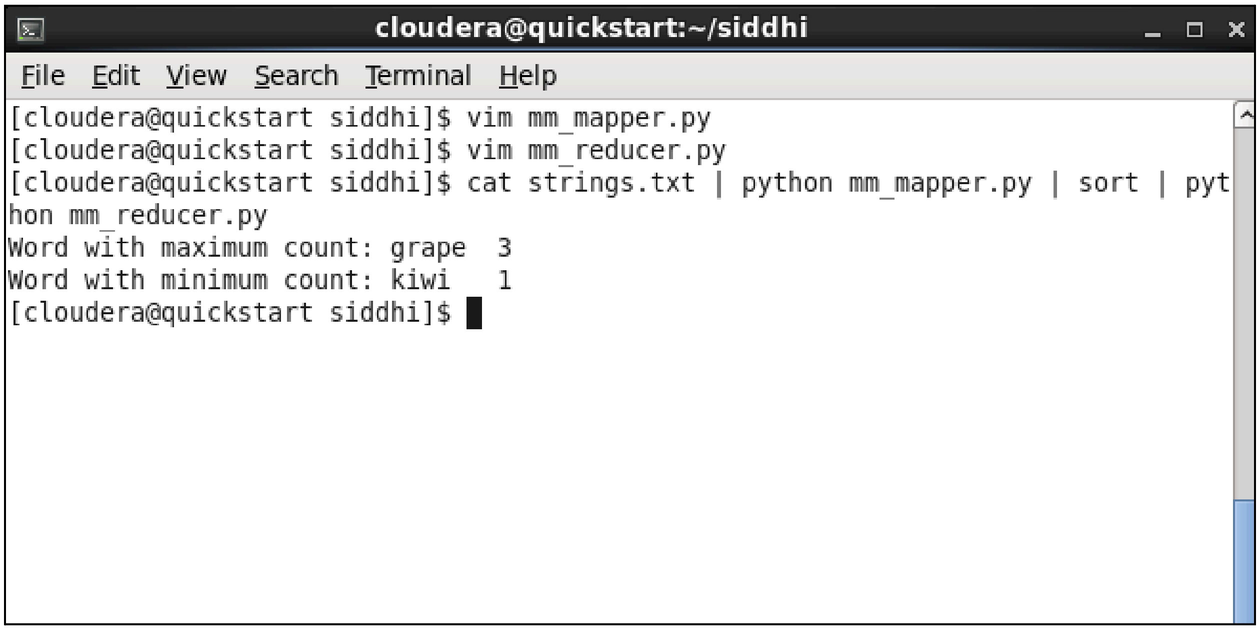
c. Word with minimum count and word with maximum count


cloudera@quickstart:~/siddhi
File Edit View Search Terminal Help

```
#!/usr/bin/env python

import sys

# Mapper function
for line in sys.stdin:
    line = line.strip()
    words = line.split()
    for word in words:
        print "%s\t%d" % (word, 1)

~
~
~
~
~
~
~
~
~
~
~
~
"mm_mapper.py" 11L, 184C                                     11,0-1        All
```


cloudera@quickstart:~/siddhi
File Edit View Search Terminal Help

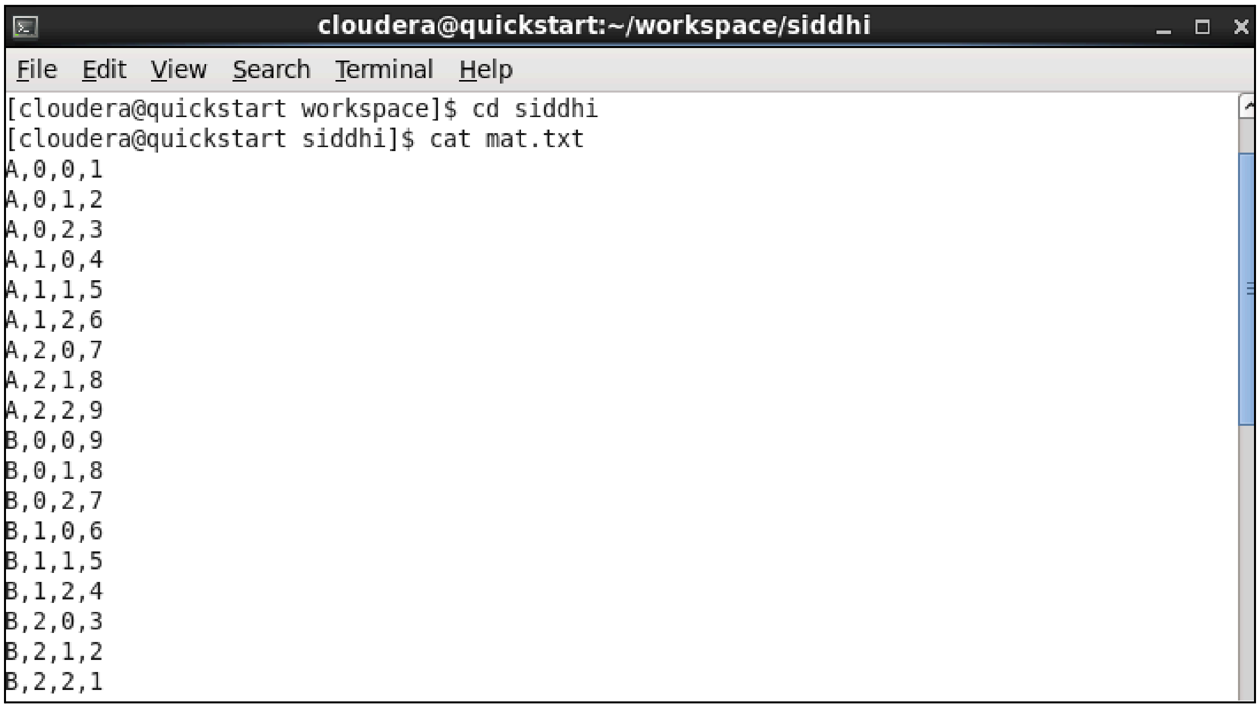
```
import sys
from collections import defaultdict

# Reducer function
word_counts = defaultdict(int)

for line in sys.stdin:
    line = line.strip()
    if line:
        word, count = line.split('\t')
        word_counts[word] += int(count)
if word_counts:
    max_word = max(word_counts, key=word_counts.get)
    min_word = min(word_counts, key=word_counts.get)
    print "Word with maximum count: %s\t%d" % (max_word, word_counts[max_word])
    print "Word with minimum count: %s\t%d" % (min_word, word_counts[min_word])
else:
    print "No words found"

~
~
~
~
"mm_reducer.py" 19L, 554C                                  19,0-1        All
```


cloudera@quickstart:~/siddhi
File Edit View Search Terminal Help

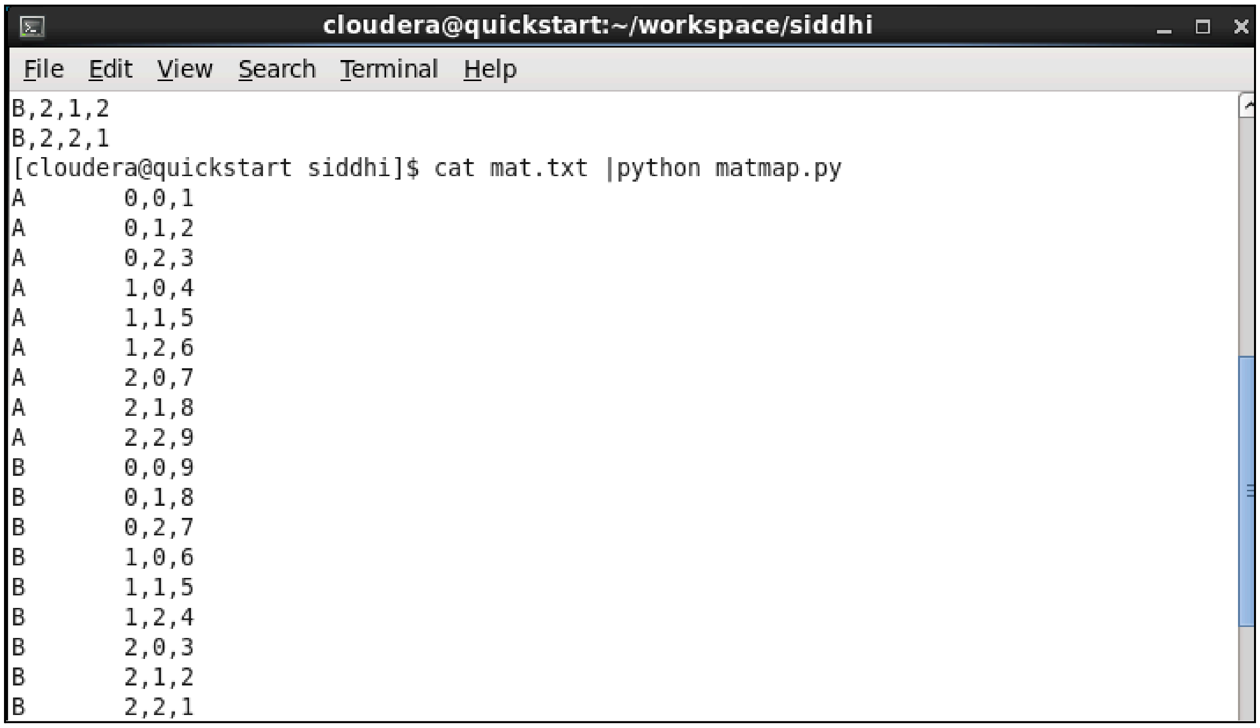
```
[cloudera@quickstart siddhi]$ vim mm_mapper.py
[cloudera@quickstart siddhi]$ vim mm_reducer.py
[cloudera@quickstart siddhi]$ cat strings.txt | python mm_mapper.py | sort | pyt
hon mm_reducer.py
Word with maximum count: grape  3
Word with minimum count: kiwi   1
[cloudera@quickstart siddhi]$ 
```

Aim: 3. Implement matrix multiplication with Hadoop Map Reduce

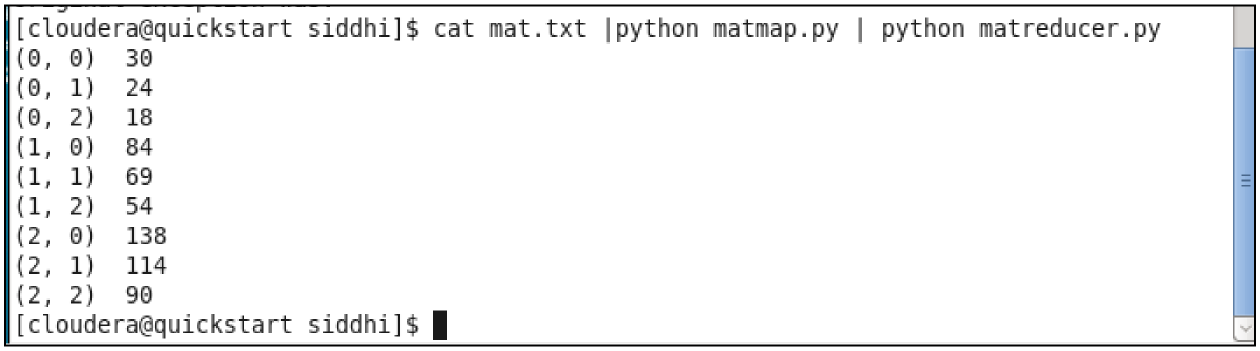
Output:


cloudera@quickstart:~/workspace/siddhi
File Edit View Search Terminal Help

```
[cloudera@quickstart workspace]$ cd siddhi
[cloudera@quickstart siddhi]$ cat mat.txt
A,0,0,1
A,0,1,2
A,0,2,3
A,1,0,4
A,1,1,5
A,1,2,6
A,2,0,7
A,2,1,8
A,2,2,9
B,0,0,9
B,0,1,8
B,0,2,7
B,1,0,6
B,1,1,5
B,1,2,4
B,2,0,3
B,2,1,2
B,2,2,1
```


cloudera@quickstart:~/workspace/siddhi
File Edit View Search Terminal Help

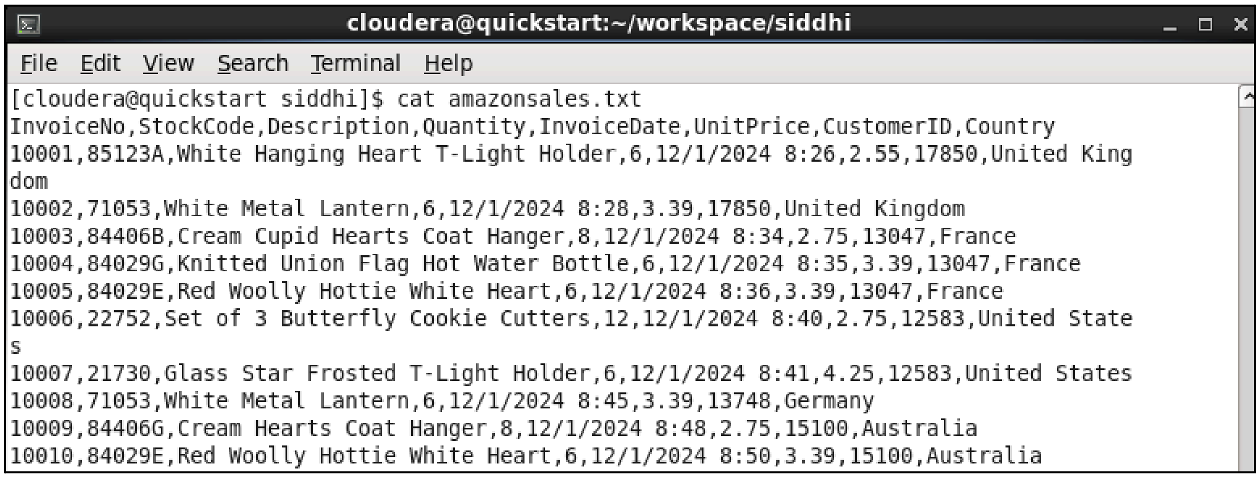
```
B,2,1,2
B,2,2,1
[cloudera@quickstart siddhi]$ cat mat.txt |python matmap.py
A       0,0,1
A       0,1,2
A       0,2,3
A       1,0,4
A       1,1,5
A       1,2,6
A       2,0,7
A       2,1,8
A       2,2,9
B       0,0,9
B       0,1,8
B       0,2,7
B       1,0,6
B       1,1,5
B       1,2,4
B       2,0,3
B       2,1,2
B       2,2,1
```



```
[cloudera@quickstart siddhi]$ cat mat.txt |python matmap.py | python matreducer.py
(0, 0)  30
(0, 1)  24
(0, 2)  18
(1, 0)  84
(1, 1)  69
(1, 2)  54
(2, 0)  138
(2, 1)  114
(2, 2)  90
[cloudera@quickstart siddhi]$ 
```

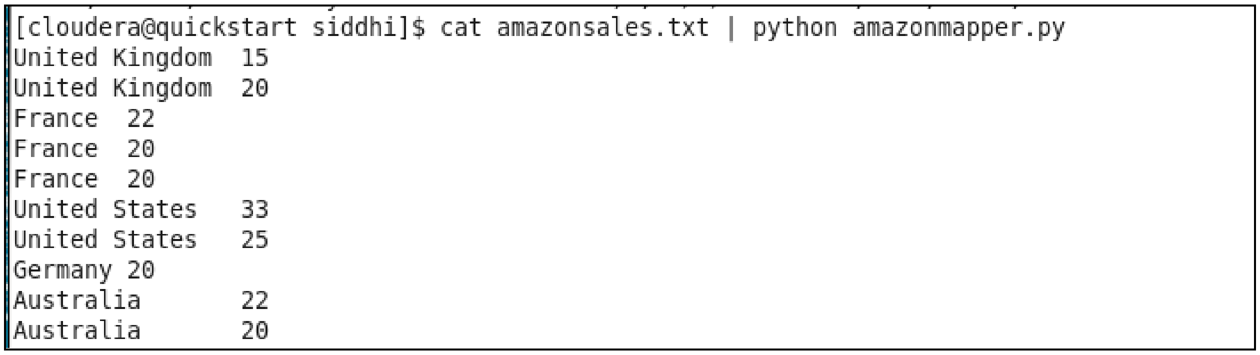
Aim: 4. Amazon collects item sold data every hour at many locations across the globe and gathers a large volume of log data, which is a good candidate for analysis with Map Reduce since it is semi-structured and record-oriented. Write a Map Reduce program that sorts unit sold data. (Refer ordered_unitsold_data.txt file)

Output:

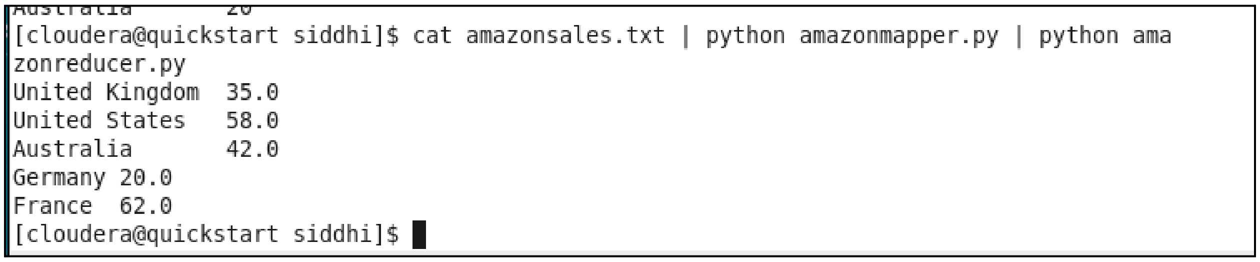


cloudera@quickstart:~/workspace/siddhi

```
[cloudera@quickstart siddhi]$ cat amazonsales.txt
InvoiceNo,StockCode,Description,Quantity,InvoiceDate,UnitPrice,CustomerID,Country
10001,85123A,White Hanging Heart T-Light Holder,6,12/1/2024 8:26,2.55,17850,United King
dom
10002,71053,White Metal Lantern,6,12/1/2024 8:28,3.39,17850,United Kingdom
10003,84406B,Cream Cupid Hearts Coat Hanger,8,12/1/2024 8:34,2.75,13047,France
10004,84029G,Knitted Union Flag Hot Water Bottle,6,12/1/2024 8:35,3.39,13047,France
10005,84029E,Red Woolly Hottie White Heart,6,12/1/2024 8:36,3.39,13047,France
10006,22752,Set of 3 Butterfly Cookie Cutters,12,12/1/2024 8:40,2.75,12583,United State
s
10007,21730,Glass Star Frosted T-Light Holder,6,12/1/2024 8:41,4.25,12583,United States
10008,71053,White Metal Lantern,6,12/1/2024 8:45,3.39,13748,Germany
10009,84406G,Cream Hearts Coat Hanger,8,12/1/2024 8:48,2.75,15100,Australia
10010,84029E,Red Woolly Hottie White Heart,6,12/1/2024 8:50,3.39,15100,Australia
```



```
[cloudera@quickstart siddhi]$ cat amazonsales.txt | python amazonmapper.py
United Kingdom  15
United Kingdom  20
France  22
France  20
France  20
United States   33
United States   25
Germany 20
Australia       22
Australia       20
```



```
[cloudera@quickstart siddhi]$ cat amazonsales.txt | python amazonmapper.py | python ama
zonreducer.py
United Kingdom  35.0
United States   58.0
Australia       42.0
Germany 20.0
France  62.0
[cloudera@quickstart siddhi]$
```