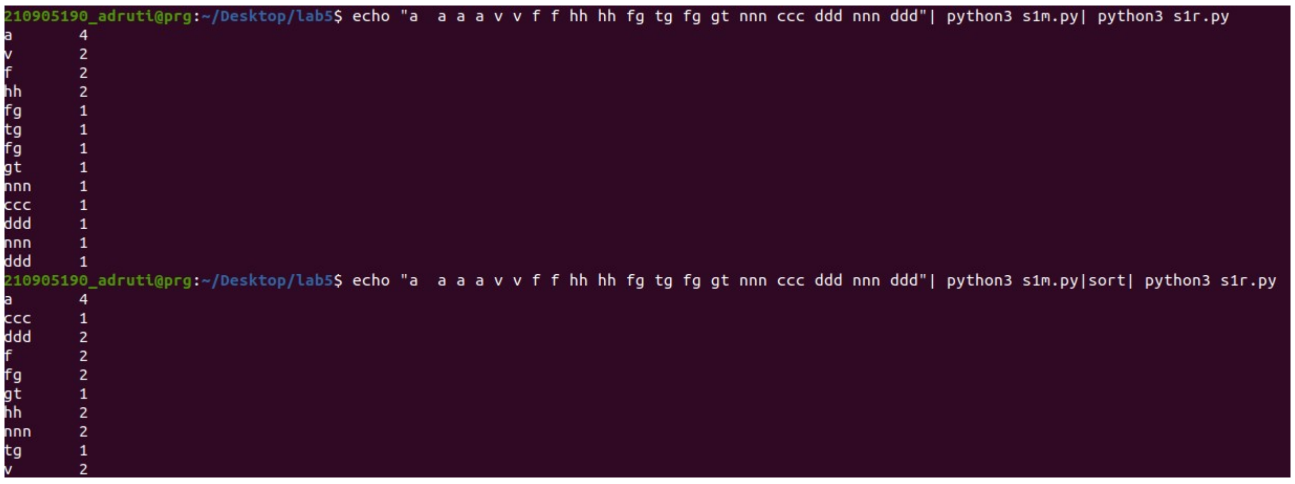
1. Word Count Program

Mapper:

```
import sys

# Input comes from STDIN (standard input)
for line in sys.stdin:
    # Remove leading and trailing whitespace
    line = line.strip()
    # Split the line into words
    words = line.split()
    # Increase counters
    for word in words:
        # Write the results to STDOUT (standard output);
        # what we output here will be the input for the
        # Reduce step, i.e. the input for reducer.py
        # Tab-delimited; the trivial word count is 1
        print("%s\t\t%s" % (word, 1))
```

Reducer:

```
import sys

current_word = None
current_count = 0
word = None

# Input comes from STDIN
for line in sys.stdin:
    # Remove leading and trailing whitespace
    line = line.strip()
    # Parse the input we got from mapper.py
    word, count = line.split('\t', 1)
    # Convert count (currently a string) to int
    try:
        count = int(count)
    except ValueError:
        # Count was not a number, so silently
        # ignore/discard this line
        continue
    # This IF-switch only works because Hadoop sorts map output
    # by key (here: word) before it is passed to the reducer
    if current_word == word:
        current_count += count
    else:
        if current_word:
            # Write result to STDOUT
            print('%s\t%s' % (current_word, current_count))
        current_count = count
        current_word = word

# Do not forget to output the last word if needed!
if current_word == word:
    print('%s\t%s' % (current_word, current_count))
```



```
210905190_adruti@prg:~/Desktop/lab5$ echo "a  a a a v v f f hh hh fg tg fg gt nnn ccc ddd nnn ddd"| python3 s1m.py| python3 s1r.py
a       4
v       2
f       2
hh      2
fg      1
tg      1
fg      1
gt      1
nnn     1
ccc     1
ddd     1
nnn     1
ddd     1
210905190_adruti@prg:~/Desktop/lab5$ echo "a  a a a v v f f hh hh fg tg fg gt nnn ccc ddd nnn ddd"| python3 s1m.py|sort| python3 s1r.py
a       4
ccc     1
ddd     2
f       2
fg      2
gt      1
hh      2
nnn     2
tg      1
v       2
```

2. MapReduce Program to Find Frequent Words

Mapper:

```
import sys
```

```
for line in sys.stdin:
    L = [(word.strip().lower(), 1) for word in line.strip().split()]
    for word, n in L:
        print('%s\t%d' % (word, n))
```

Reducer:

```
import sys

lastWord = None
sum = 0

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

    if lastWord == None:
        lastWord = word
        sum = count
        continue

    if word == lastWord:
        sum += count
    else:
        print("%s\t%d" % (lastWord, sum))
        sum = count
        lastWord = word

if lastWord == word:
    print('%s\t%s' % (lastWord, sum))
```

Mapper:

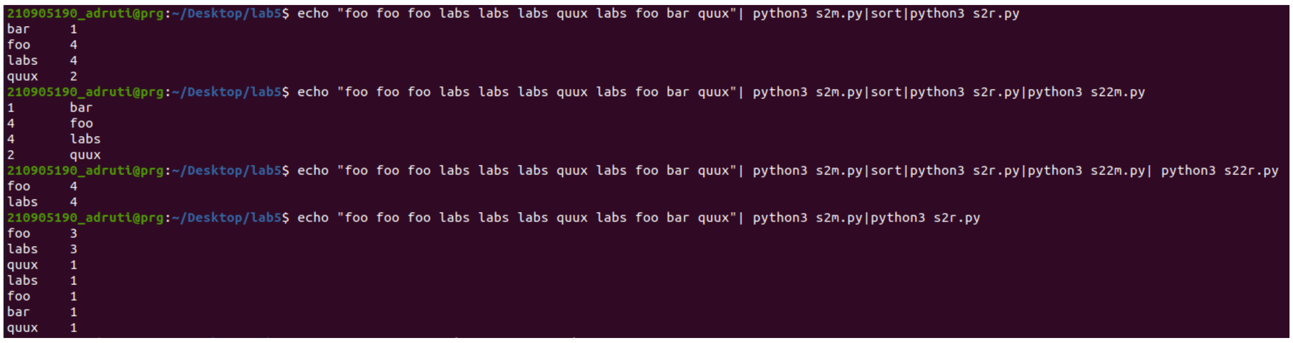
```
#!/usr/bin/env python
```

```
# A basic mapper function/program that
# takes whatever is passed on the input and
# outputs tuples of all the words formatted
# as (word, 1)
from __future__ import print_function
import sys

# input comes from STDIN (standard input)
for line in sys.stdin:
    word, count = line.strip().split('\t', 1)
    count = int(count)
    print('%d\t%s' % (count, word))
```

Reducer:

```
#!/usr/bin/env python
# reducer.py
from __future__ import print_function
import sys

mostFreq = []
currentMax = -1

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

    if count > currentMax:
        currentMax = count
        mostFreq = [word]
    elif count == currentMax:
        mostFreq.append(word)

# Output most frequent word(s)
```

```
for word in mostFreq:
    print('%s\t%s' % (word, currentMax))
```



```
210905190_adruti@prg:~/Desktop/lab5$ echo "foo foo foo labs labs labs quux labs foo bar quux"| python3 s2m.py|sort|python3 s2r.py
bar     1
foo     4
labs    4
quux    2
210905190_adruti@prg:~/Desktop/lab5$ echo "foo foo foo labs labs labs quux labs foo bar quux"| python3 s2m.py|sort|python3 s2r.py|python3 s22m.py
1       bar
4       foo
4       labs
2       quux
210905190_adruti@prg:~/Desktop/lab5$ echo "foo foo foo labs labs labs quux labs foo bar quux"| python3 s2m.py|sort|python3 s2r.py|python3 s22m.py| python3 s22r.py
foo     4
labs    4
210905190_adruti@prg:~/Desktop/lab5$ echo "foo foo foo labs labs labs quux labs foo bar quux"| python3 s2m.py|python3 s2r.py
foo     3
labs    3
quux    1
labs    1
foo     1
bar     1
quux    1
```

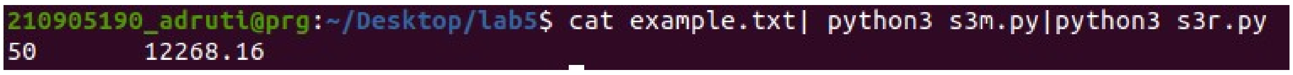
3. MapReduce Program to Explore and Filter Dataset

Mapper:

```
import fileinput

# Loop through each line of input from file or stdin

for line in fileinput.input():

    # Split the line into fields

    data = line.strip().split("\t")

    # Check if the line has 6 fields (assuming it's properly formatted)

    if len(data) == 6:

        # Extract location and cost fields

        date, time, location, item, cost, payment = data

        # Output location and cost as key-value pairs

        print("{0}\t{1}".format(location, cost))
```

Reducer:

```
import fileinput

# Initialize counters
transactions_count = 0
sales_total = 0

# Loop through each line of input from file or stdin
for line in fileinput.input():
    # Split the line into key-value pairs
    data = line.strip().split("\t")
    # Check if the line has 2 fields (assuming it's properly formatted)
    if len(data) != 2:
        # Something has gone wrong. Skip this line.
        continue
    # Extract key and value
    current_key, current_value = data
    # Increment transaction count
    transactions_count += 1
    # Add current transaction value to total sales
    sales_total += float(current_value)

# Print the total number of transactions and the total sales amount
print(transactions_count, "\t", sales_total)
```



```
210905190_adruti@prg:~/Desktop/lab5$ cat example.txt| python3 s3m.py|python3 s3r.py
50       12268.16
```