LAB 5 – MapReduce Programming in Python

AKSHAT KANSAL

180905206

Section B – 29

1.

Word Count program.

Map.py

```
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 result to STDOUT
    #what we output here is input for REDUCE step
    #word count is 1
    print('%s\t%s' %(word,1))
```

reduce.py

```
from operator import itemgetter
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 map.py
  word,count = line.split('\t',1)
  #conver count to int
  try:
    count=int(count)
```

```
  except ValueError:
    #count was not a number so ignore
    continue
  if current_word == word:
    current_count+=count
  else:
    if current_word:
      print('%s\t%s'%(current_word,current_count))
    current_count = count
    current_word = word
#do not forget to output the last word
if current_word == word:
  print('%s\t%s'%(current_word,current_count))
```

Section of the output for covid_19_data.csv

```
Afghanistan     213
Africa  203
African 193
Albania 199
Alberta"        5
Algeria 212
Andorra 206
Angola  188
Antigua 195
Arab    239
Arabia  206
Argentina       205
Armenia 207
Aruba   7
Australia       1804
Austria 212
Azerbaijan      208
Bahamas 192
Bahrain 213
Bangladesh      200
Bank    182
Barbados        191
Barbuda 195
Barthelemy      7
Belarus 209
Belgium 233
Belize  185
Benin   192
Bhutan  202
Bolivia 197
Bosnia  203
Botswana        178
Brazil  3533
Brunei  199
Bulgaria        200
Burkina 198
Burma   181
Burundi 177
CA      17
CA"     270
CO"     14
CT"     3
Cabo    188
Cambodia        241
Cameroon        202
Canada  2787
Cape    1
Cayman  3
```

2.

Frequent word count program.

Map1.py

```
#!/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:
  # create tuples of all words in line
  L = [ (word.strip().lower(), 1 ) for word in line.strip().split() ]
  # increase counters
  for word, n in L:
    # 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%d'%(word, n))
```

reduce1.py

```
#!/usr/bin/env python
# reducer.py
from __future__ import print_function
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
# output last word
```

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

map2.py

```
#!/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) )
```

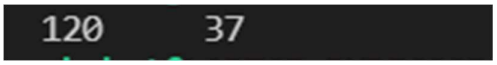
reduce2.py

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

Output for example.txt

Output for covid dataset, country

Output for heart dataset, trestbps



Output for German Credit dataset, duration

3.

Item explore and count program

Map.py

```
import csv
from operator import delitem

filename = 'covid_19_data.csv'
with open(filename) as csv_file:
  csv_reader = csv.reader(csv_file, delimiter=',')
  line_count = 0
  for row in csv_reader:
    if line_count == 0:
      line_count+=1
      continue
    else:
      print(row[3]+'\t'+row[5]+'\n')
```

reduce.py

```
import fileinput

country_count = 0
cases_total = 0

for line in fileinput.input():
  data = line.strip().split('\t')
  if len(data) != 2:
```

```
    continue
  current_key,current_value = data
  country_count += 1
  cases_total += float(current_value)

print(country_count,'\t',cases_total)
```

Output for covid dataset, (country,confirmed cases)

```
116805    2228892527.0
```

4.

Map.py

```
import sys
def read_input(file):
  for line in file:
    # split the line into words
    yield line.split()
def main(separator='\t'):
  # input comes from STDIN (standard input)
  data = read_input(sys.stdin)
  for words in data:
    # 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
    for word in words:
      print ('%s%s%d' % (word, separator, 1))
if __name__ == "__main__":
  main()
```

reduce.py

```
from itertools import groupby
from operator import itemgetter
import sys
def read_mapper_output(file, separator='\t'):
  for line in file:
    yield line.rstrip().split(separator, 1)

def main(separator='\t'):
  # input comes from STDIN (standard input)
```

```
  data = read_mapper_output(sys.stdin, separator=separator)
  # groupby groups multiple word-count pairs by word,
  # and creates an iterator that returns consecutive keys and their group:
  # current_word - string containing a word (the key)
  # group - iterator yielding all ["&lt;current_word&gt;","&lt;count&gt;"] ite
ms
  for current_word, group in groupby(data, itemgetter(0)):
    try:
      total_count = sum(int(count) for current_word, count in group)
      print ("%s%s%d" % (current_word, separator, total_count))
    except ValueError:
      # count was not a number, so silently discard this item
      pass
if __name__ == "__main__":
  main()
```

Output for heart disease dataset, counts of each age

```
29      1
34      2
35      4
37      2
38      3
39      4
40      3
41      10
42      8
43      8
44      11
45      8
46      7
47      5
48      7
49      5
50      7
51      12
52      13
53      8
54      16
55      8
56      11
57      17
58      19
59      14
60      11
61      8
62      11
63      9
64      10
65      8
66      7
67      9
68      4
69      3
70      4
71      3
74      1
76      1
77      1
```

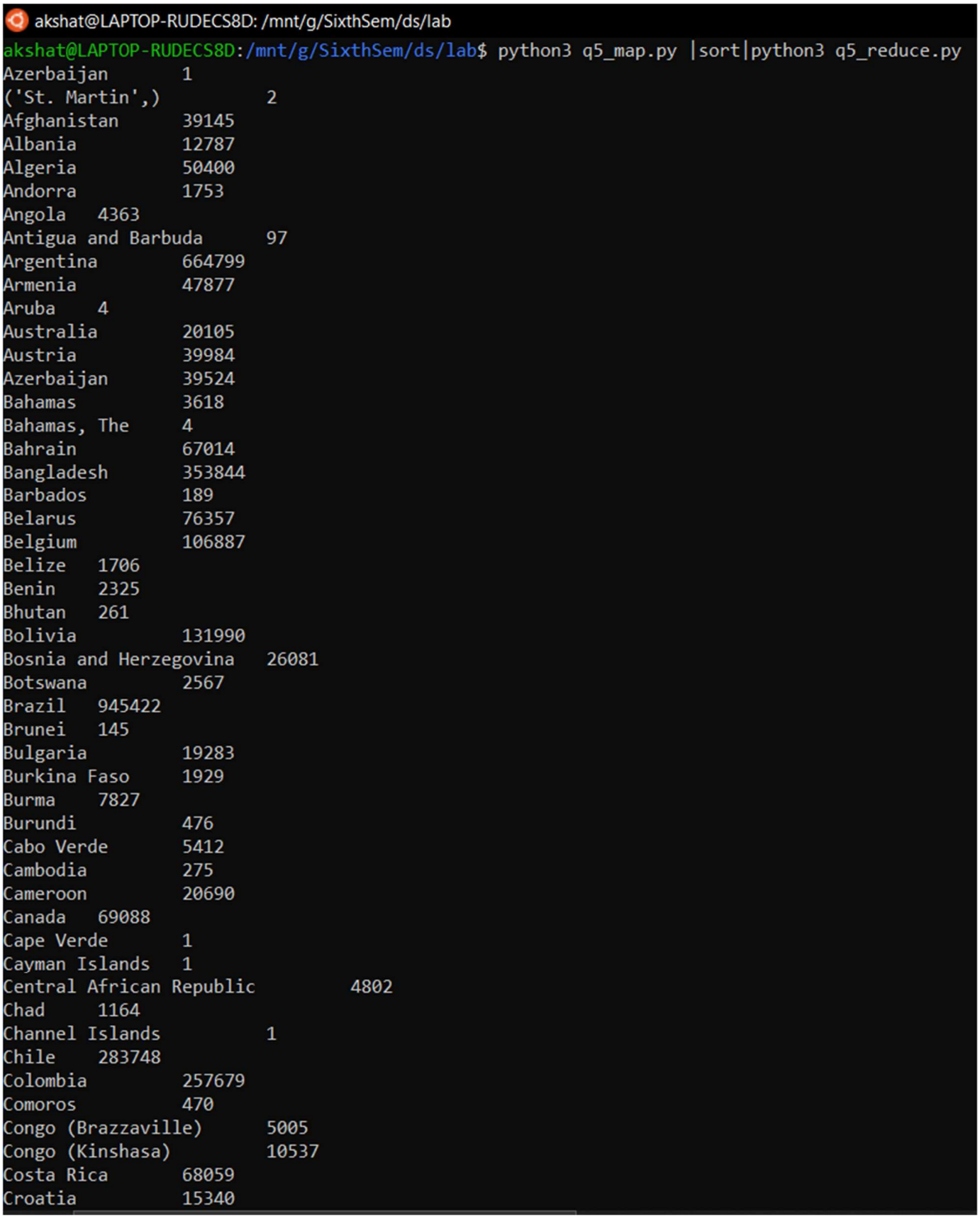
5.

Finding max value using MapReduce Concept

```
import csv
from operator import delitem

filename = 'covid_19_data.csv'
with open(filename) as csv_file:
  csv_reader = csv.reader(csv_file, delimiter=',')
  line_count = 0
  for row in csv_reader:
    if line_count == 0:
      line_count+=1
      continue
    else:
      print(row[3]+'\t'+row[5]+'\n')
```

```
import fileinput

max_value = 0
old_key = None

for line in fileinput.input():
  data = line.strip().split("\t")
  if len(data) != 2:
    # Something has gone wrong. Skip this line.
    continue
  current_key, current_value = data
  # Refresh for new keys (i.e. locations in the example context)
  if old_key and old_key != current_key:
    print (old_key, "\t", max_value)
    old_key = current_key
    max_value = 0
  old_key = current_key
  if int(current_value) > int(max_value):
    max_value = int(current_value)
if old_key != None:
  print (old_key, "\t", max_value)
```

For covid 19 dataset, max value of confirmed cases in a state/provice grouped by country



6.

Birth Data

Mapper.py

```
import sys

for line in sys.stdin:
    line = line.strip()
    words = line.split()
    print ('%s %s %s %s %s' % (words[14], words[15], words[16],words[31], 1))
```

reducer.py

```
from operator import itemgetter
import sys

males = 0
females = 0
current_year = 0
year = None

months_females = [0,0,0,0,0,0,0,0,0,0,0,0]
months_males = [0,0,0,0,0,0,0,0,0,0,0,0]
print("yearly data")
for line in sys.stdin:
    line = line.strip()
    year, month, date, gender, count = line.split(' ',4)
    try:
        count = int(count)
    except ValueError:
        continue

    if gender == 'M':
        months_males[int(month)]+=1
    else:
        months_females[int(month)]+=1
    if int(year) == current_year:
        if gender == 'M':
            males += 1
        else:
            females += 1
    else:
        if current_year >=0 :
```

```
            print ('year %s females: %s males: %s total births: %s' % (current
_year, females, males, males+females))
        current_year = int(year)
        if gender == 'M':
            males = 1
        else:
            females = 1

if current_year == year:
    print ('year %s females: %s males: %s total births: %s' % (current_year, f
emales, males, males+females))

print("")
print("monthly data")
for i in range(0,12):
    print ('month %s males: %s females: %s total births: %s' % (i+1, months_ma
les[i], months_females[i], months_males[i]+months_females[i]))
```

```
akshat@LAPTOP-RUDECS8D:/mnt/g/SixthSem/ds/lab$ cat birth_sample.txt|python3 mapper.py|sort|python3 reducer.py
yearly data
year 0 females: 37961 males: 40068 total births: 78029
year 1 females: 7331 males: 47833 total births: 55164
year 2 females: 2242 males: 50041 total births: 52283
year 3 females: 684 males: 50751 total births: 51435
year 4 females: 220 males: 50966 total births: 51186
year 5 females: 87 males: 51054 total births: 51141
year 6 females: 37 males: 51105 total births: 51142
year 7 females: 23 males: 51123 total births: 51146
year 8 females: 12 males: 51134 total births: 51146
year 9 females: 5 males: 51141 total births: 51146
year 10 females: 3 males: 51144 total births: 51147
year 11 females: 1 males: 51144 total births: 51145
year 12 females: 1 males: 1 total births: 2
year 13 females: 1 males: 2 total births: 3
year 14 females: 1 males: 2 total births: 3
year 16 females: 1 males: 2 total births: 3
year 20 females: 1 males: 1 total births: 2
year 21 females: 1 males: 1 total births: 2

monthly data
month 1 males: 0 females: 4 total births: 4
month 2 males: 20801 females: 19806 total births: 40607
month 3 males: 16215 females: 15416 total births: 31631
month 4 males: 8439 females: 8011 total births: 16450
month 5 males: 3421 females: 3280 total births: 6701
month 6 males: 1319 females: 1202 total births: 2521
month 7 males: 529 females: 511 total births: 1040
month 8 males: 244 females: 231 total births: 475
month 9 males: 234 females: 211 total births: 445
month 10 males: 72 females: 54 total births: 126
month 11 males: 0 females: 0 total births: 0
month 12 males: 0 females: 0 total births: 0
akshat@LAPTOP-RUDECS8D:/mnt/g/SixthSem/ds/lab$
```

7.

Count even/odd in randomly generated natural numbers

Random generator

```
import random

for i in range(100):
  n = random.randint(1, 1000)
  print(n)
```

map.py

```
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
  nums = line.split()
  # increase counters
  for num in nums:
    val = int(num)
    if val % 2 == 0:
      print('%s\t%s' % ("EVEN", 1))
    else:
      print('%s\t%s' % ("ODD", 1))
```

reduce.py

```
from operator import itemgetter
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))
```

Output

```
akshat@LAPTOP-RUDECS8D:/mnt/g/SixthSem/ds/lab$ python3 randomGen.py|python3 q7_map.py |sort|python3 q7_reduce.py
EVEN    52
ODD     48
akshat@LAPTOP-RUDECS8D:/mnt/g/SixthSem/ds/lab$ python3 randomGen.py|python3 q7_map.py |sort|python3 q7_reduce.py
EVEN    47
ODD     53
akshat@LAPTOP-RUDECS8D:/mnt/g/SixthSem/ds/lab$ python3 randomGen.py|python3 q7_map.py |sort|python3 q7_reduce.py
EVEN    54
ODD     46
```