**DS Lab 5 – MapReduce Programming using Python**

**Name:** GP Anirudh

**Registration Number:** 180905452

**Section:** B

**Batch:** B2

**Roll Number:** 59

1) Write a basic wordcount program.

**q1Mapper:**

import sys  
for line in sys.stdin:

line = line.strip()

words = line.split()

for word in words:

print ('%s\t%s' % (word, 1))

**q1Reducer:**

from operator import itemgetter

import sys  
current\_word = None

current\_count = 0

word = None

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\_word = word

if current\_word == word:

print ('%s\t%s' % (current\_word, current\_count))

**Output:**

Shape

Description automatically generatedHeart Disease :

Text

Description automatically generatedCOVID-19:

Text

Description automatically generatedExample:

German Credit:

A picture containing text

Description automatically generated

2) MapReduce program to find frequent words

**q2Mapper:**

from \_\_future\_\_ import print\_function

import sys

for line in sys.stdin:

word, count = line.strip().split('\t', 1)

count = int(count)

print(f'{count} \t {word}')

**q2Reducer:**

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 )

print()

for word in mostFreq:

print(f'{word} \t {currentMax}')

print()

**Output:**

Heart Disease :

Graphical user interface, text, application

Description automatically generatedGraphical user interface, text, application

Description automatically generated

COVID-19:

Graphical user interface, text, application

Description automatically generated

Example:

Graphical user interface, text, application

Description automatically generated

German Credit:

3) MapReduce program to explore the dataset and perform the filtering (typically creating key/value pairs) by mapper and perform the count and summary operation on the instances.

**q3Mapper.py:**

import sys

import pandas as pd

# Heart Disease

df = pd.read\_csv('heart\_disease\_data.csv')

df = df.dropna()

words1 = list(df['age'].values)

words2 = list(df['chol'].values)

# Covid 19

df = pd.read\_csv('covid\_19\_data.csv')

df = df.dropna()

words1 = list(df['Country/Region'].values)

words2 = list(df['Confirmed'].values)

# Example

df = pd.read\_csv('example.txt', delimiter='\t')

df.columns = ['date', 'time', 'city', 'type', 'amount', 'payment']

df = df.dropna()

words1 = list(df['city'].values)

words2 = list(df['amount'].values)

# German Credit

df = pd.read\_csv('german\_credit.csv')

df = df.dropna()

words1 = list(df['DurationOfCreditInMonths'].values)

words2 = list(df['CreditAmount'].values)

for (word1, word2) in zip(words1, words2):

print(f'{word1} \t {word2}')

**q3Reducer.py:**

import fileinput

item\_count = 0

item\_total = 0

for line in fileinput.input():

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

if len(data) != 2:

continue

current\_key, current\_value = data

item\_count += 1

item\_total += float(current\_value)

print()

print (item\_count, "\t", item\_total)

print()

Graphical user interface, text, application

Description automatically generated**Output:**

Heart Disease :

Graphical user interface, text, application, email

Description automatically generatedCOVID-19:

Graphical user interface, application

Description automatically generatedExample:

Graphical user interface, application

Description automatically generated

German Credit:

Q4) Write a mapper and reducer program for word count by defining separator instead of using “\t”.

**q4Mapper.py:**

import sys

import pandas as pd

# Heart Disease

df = pd.read\_csv('heart\_disease\_data.csv') df = df.dropna()

words = list(df['age'].values)

# Covid 19

df = pd.read\_csv('covid\_19\_data.csv')

df = df.dropna()

words = list(df['Country/Region'].values)

# Example

df = pd.read\_csv('example.txt', delimiter='\t')

df.columns = ['date', 'time', 'city', 'type', 'amount', 'payment'] df = df.dropna()

words = list(df['type'].values)

# German Credit Dataset

df = pd.read\_csv('german\_credit.csv')

df = df.dropna()

words = list(df['DurationOfCreditInMonths'].values)

def main(separator=' --> '):

for word in words:

print('%s%s%d' % (word, separator, 1))

if \_\_name\_\_ == "\_\_main\_\_":

main()

**q4Reducer.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=' --> '):

data = read\_mapper\_output(sys.stdin, separator=separator)

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:

print("Value Error!")

pass

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Output:**

A screenshot of a computer

Description automatically generated with low confidenceHeart Disease :

Text

Description automatically generatedCOVID-19:

Example:

Text

Description automatically generated

A picture containing text

Description automatically generated

German Credit:

5) Try to apply finding max value using map reduce concept for the output of Heart Disease dataset, covid\_19\_data dataset, example dataset and German Credit dataset.

**q5Mapper.py:**

import sys

import pandas as pd

# Heart Disease

df = pd.read\_csv('heart\_disease\_data.csv')

df = df.dropna()

words1 = list(df['age'].values)

words2 = list(df['chol'].values)

# Covid 19

df = pd.read\_csv('covid\_19\_data.csv')

df = df.dropna()

words1 = list(df['Country/Region'].values)

words2 = list(df['Confirmed'].values)

# Example

df = pd.read\_csv('example.txt', delimiter='\t')

df.columns = ['date', 'time', 'city', 'type', 'amount', 'payment']

df = df.dropna()

words1 = list(df['city'].values)

words2 = list(df['amount'].values)

# German Credit

df = pd.read\_csv('german\_credit.csv')

df = df.dropna()

words1 = list(df['DurationOfCreditInMonths'].values)

words2 = list(df['CreditAmount'].values)

for (word1, word2) in zip(words1, words2):

print(f'{word1} \t {word2}')

**q5Reducer.py:**

import fileinput

max\_value = 0

old\_key = None

for line in fileinput.input():

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

if len(data) != 2:

continue

current\_key, current\_value = data

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 float(current\_value) > float(max\_value):

max\_value = float(current\_value)

if old\_key != None:

print (old\_key, "\t", max\_value)

**Output:**

A picture containing text, screenshot

Description automatically generatedHeart Disease :

A picture containing text, black, screenshot

Description automatically generatedCOVID-19:

Example:

A picture containing text

Description automatically generated

A picture containing table

Description automatically generatedGerman Credit:

6) Write a mapreduce program to evaluate the PI.

**q6Mapper.py:**

import sys

def f( x ):

return 4.0 / ( 1.0 + x\*x )

for line in sys.stdin:

line = line.strip()

words = line.split()

N = int( words[0] )

deltaX = 1.0 / N

for i in range( 0, N ):

print( "1\t%1.10f" % ( f( i \* deltaX )\*deltaX ) )

**q6Reducer:**

from \_\_future\_\_ import print\_function

from operator import itemgetter

import sys  
sum = 0

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

try:

count = float(count)

except ValueError:

# count was not a number, so silently  
# ignore/discard this line  
#print( "--skipping (%s, %s)" % ( str(word), str(count) ) )

continue

sum += count

print( '%1.10f\t0' % sum )

**Output:**



7) Write a MapReduce program to generate a report with Number of males, females and total births in each year, number of males, females and total births in each month of a particular year from national birth data.

**q7.py:**

import pandas as pd

df = pd.read\_csv('national\_birth\_data.csv')

# n\_males, n\_females, n\_births per year

# q6mapper

mapper = ""

years = list(df['year'].values)

genders = list(df['gender'].values)

for (year, gender) in zip(years, genders):

mapper = mapper + str(year) + ' \t ' + gender + ' \t ' + '1' + '

\n'

mapper = mapper[:-1]

# q6reducer

male = 0

female = 0

total = 0

current\_year = None

print('year \t total \t male \t female')

for line in mapper.split('\n'):

year, gender, count = line.split(' \t ')

try:

count = int(count)

except ValueError:

print("Value Error!")

continue

if current\_month == month:

total = total + count

if gender == 'male':

male = male + count

if gender == 'female':

female = female + count

if current\_month:

print(f'2001 \t {current\_month} \t {total} \t {male} \t

{female}')

total = count

if gender == 'male':

male = count

if gender == 'female':

female = count

current\_month = month

if current\_month == month:

print(f'2001 \t {current\_month} \t {total} \t {male} \t {female}

Text

Description automatically generated with medium confidence')

**Output:**

8) Write a MapReduce program to count even or odd numbers in randomly generated natural numbers.

**q8.py:**

from operator import itemgetter

import sys

odd\_count = 0

even\_count = 0

for line in sys.stdin:

line = line.strip()

num = line.split()

for currnum in num:

try:

odd\_count = int(odd\_count)

even\_count = int(even\_count)

currnum = int(currnum)

except ValueError:

continue

if currnum%2 == 0:

even\_count += 1

else:

odd\_count += 1

print ('%s odd and %s even' % (odd\_count, even\_count))

Graphical user interface, text, application

Description automatically generated**Output:**