**CS595 - Assignment 3**

1. Slightly modify the WordCount program. Call the new program WordCount2.py.

Instead of counting how many words there are in the input documents, modify the program to count how many words begin with the small letters a-n and how many begin with anything else.

The output file should look something like

a\_to\_n, 12

other, 21

(5 points) Submit a copy of this modified program and a screen shot of the results of the program’s execution as the output of your assignment.

**Modified Program Code:** [WordCount2.py]

from mrjob.job import MRJob

import re

WORD\_RE = re.compile(r"[\w']+")

class MRModifiedWordCount(MRJob):

def mapper(self, \_, line):

for word in WORD\_RE.findall(line):

if re.match("^[a-n]+.\*", word.lower()):

yield "a\_to\_n", 1

else:

yield "Other", 1

def combiner(self, word, counts):

yield word, sum(counts)

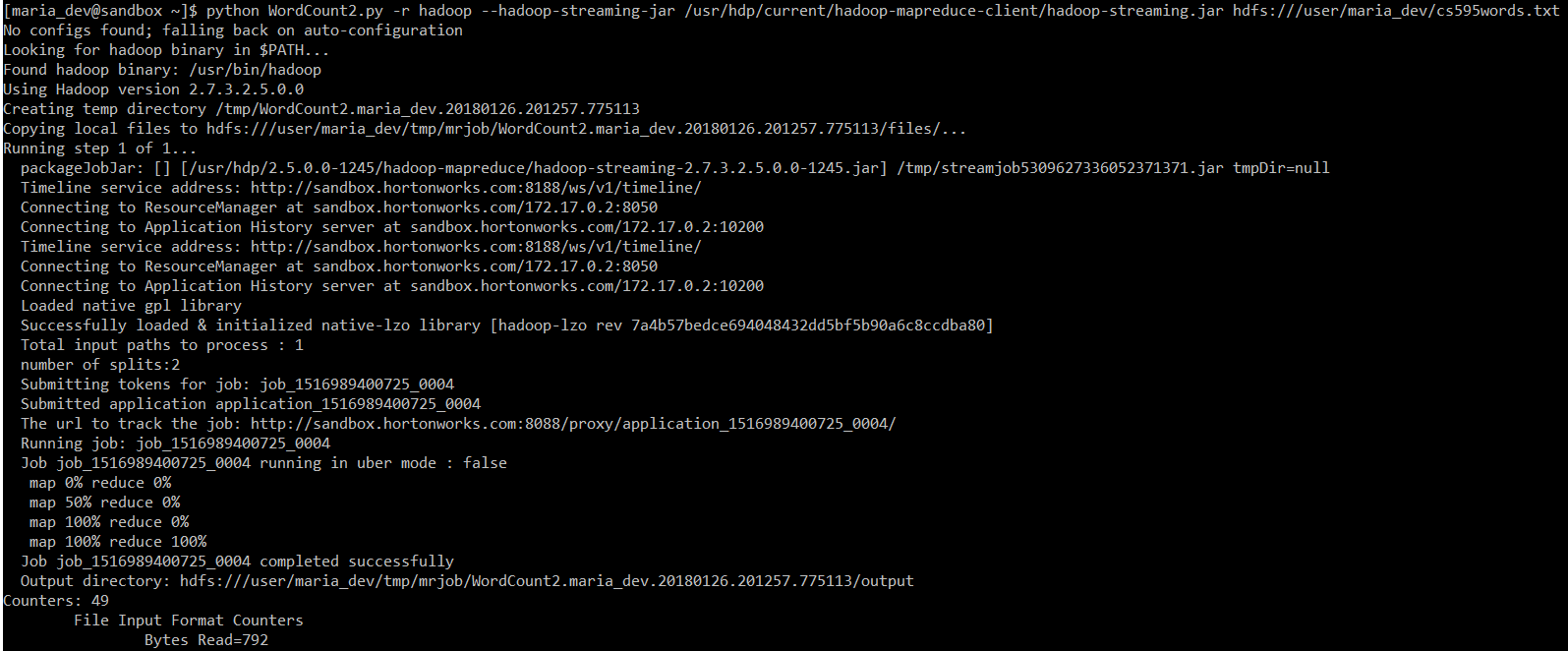
def reducer(self, word, counts):

yield word, sum(counts)

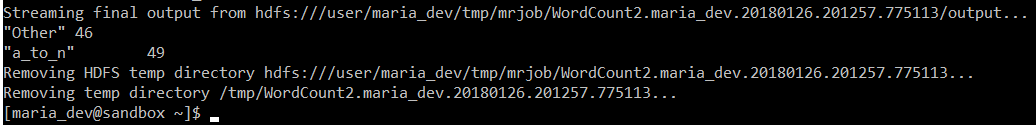
if \_\_name\_\_ == '\_\_main\_\_':

MRModifiedWordCount.run()

**Execution Command:**

****

**Output Result:**

****

1. Modify the Salaries.py program. Call it Salaries2.py

Instead of counting the number of workers per department, change the program to provide the number of workers having High, Medium or Low annual salaries. This is defined as follows:

|  |  |
| --- | --- |
| High | 100,000.00 and above |
| Medium | 50,000.00 to 99,999.99 |
| Low | 0.00 to 49,999.99 |

The output of the program should be something like the following (in any order):

High 20

Medium 30

Low 10

Some important hints:

* The annual salary is a string that will need to be converted to a float.
* The mapper should output tuples with one of three keys depending on the annual salary: High, Medium and Low
* The value part of the tuple is not a salary. (What should it be?)

(5 points) Submit a copy of this modified program and a screen shot of the results of the program’s execution as the output of your assignment.

**Modified Program Code:** [Salaries2.py]

from mrjob.job import MRJob

class MRModifiedSalaries(MRJob):

def mapper(self, \_, line):

(name,jobTitle,agencyID,agency,hireDate,annualSalary,grossPay) = line.split('\t')

if float(annualSalary) >= float(100000.00):

yield "High", 1

elif float(annualSalary) >= float(50000.00) and float(annualSalary) < float(100000.00):

yield "Medium", 1

else:

yield "Low", 1

def combiner(self, jobTitle, counts):

yield jobTitle, sum(counts)

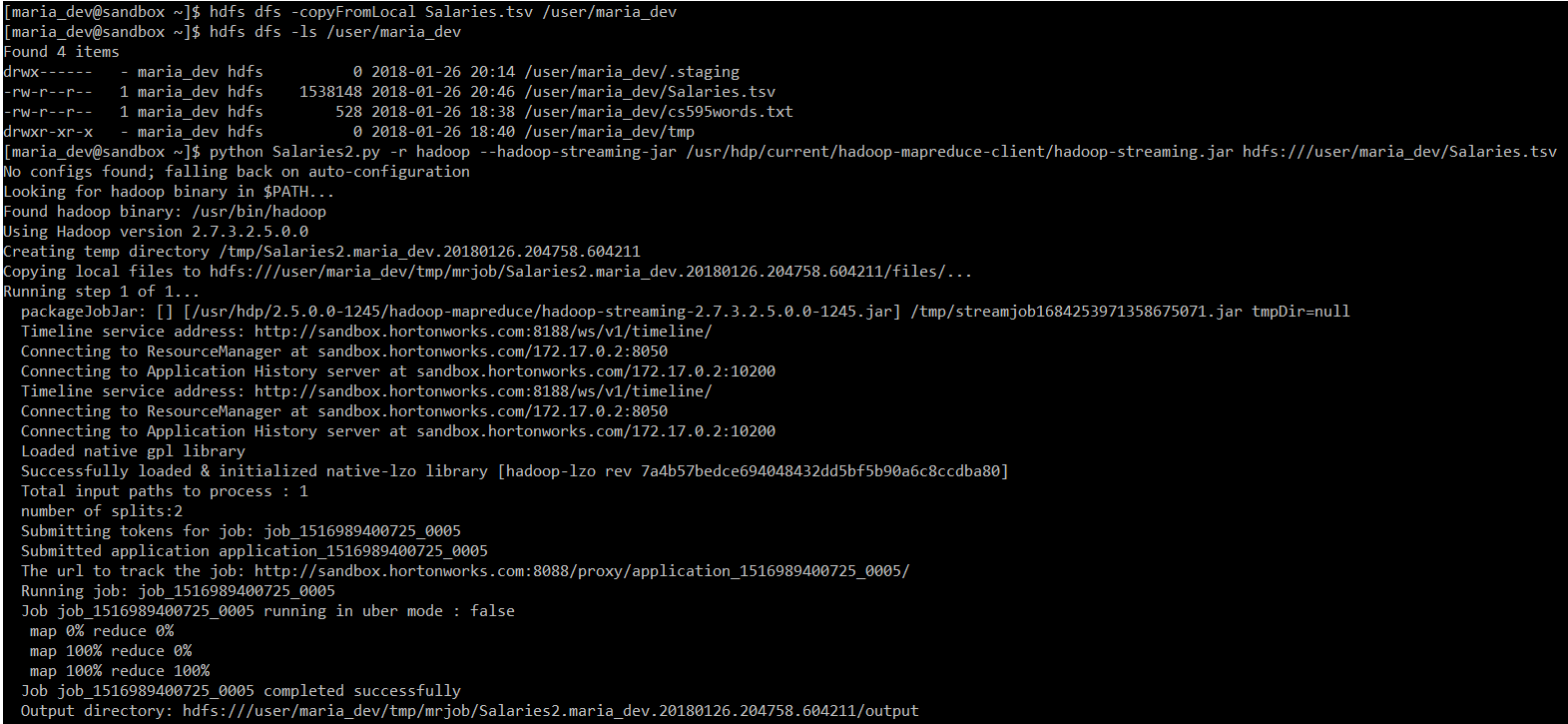
def reducer(self, jobTitle, counts):

yield jobTitle, sum(counts)

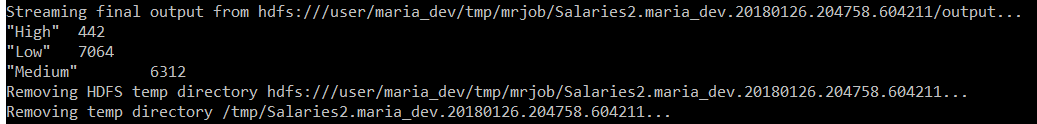
if \_\_name\_\_ == '\_\_main\_\_':

MRModifiedSalaries.run()

**Execution Command:**

****

**Output Result:**

****

1. Copy the file u.data to /user/maria\_dev. This is similar to the file used for some examples in Module 03b. NOTE: unlike the slide deck examples, this version of u.data has fields separated by commas and not tabs.

(5 points) Review the slides 17-22 in lecture notes Module 3b. Now write a program to perform the task of outputting a count of the number of movies each user (identified via their user id) reviewed.

Output might look something like the following:

* + 186: 2
  + 192: 2
  + 112: 1
  + etc.

Submit a copy of this program and a screen shot of the results of the program’s execution (only 10 lines or so of the result) as the output of your assignment.

**Modified Program Code:** [Movies.py]

from mrjob.job import MRJob

class MRMoviesReviewed(MRJob):

def mapper(self, \_, line):

(userID,movieID,rating,timestamp) = line.split(',')

yield userID, movieID

def combiner(self, userID, movies):

yield userID, len(list(movies))

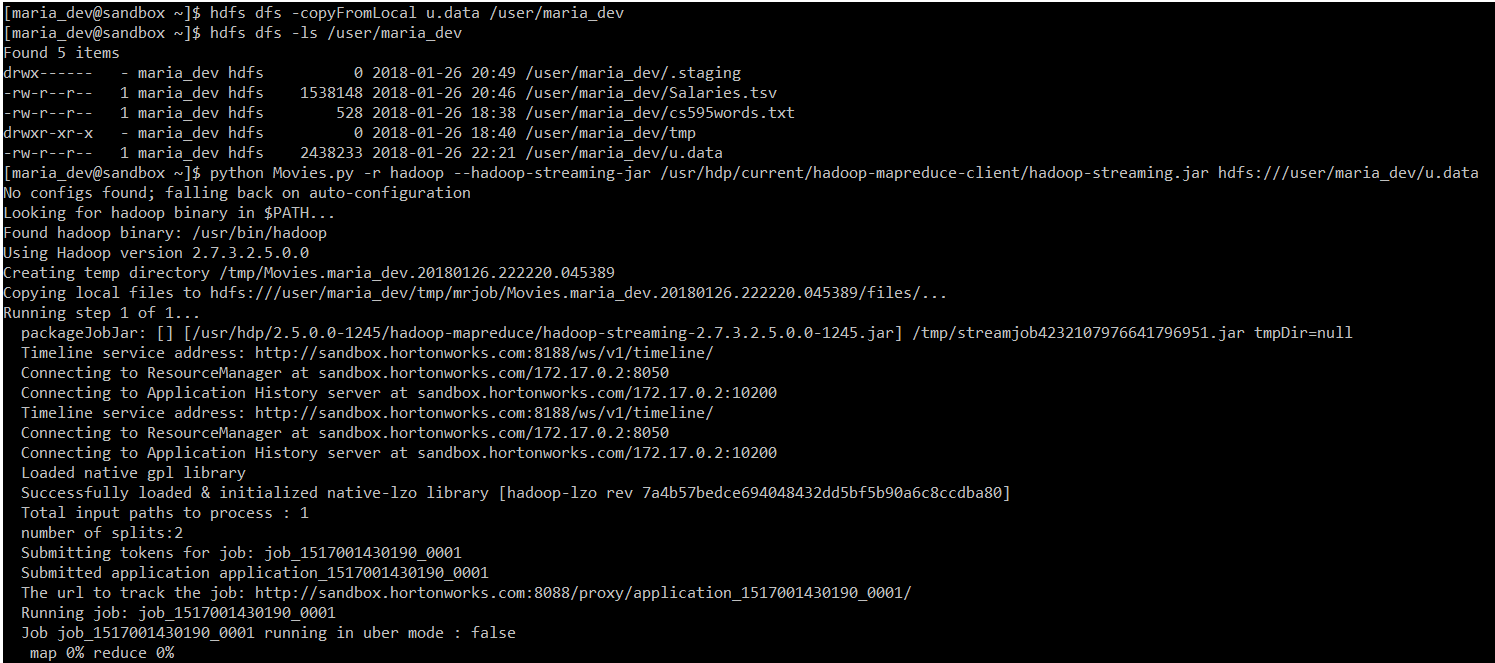
def reducer(self, userID, count):

yield userID, sum(count)

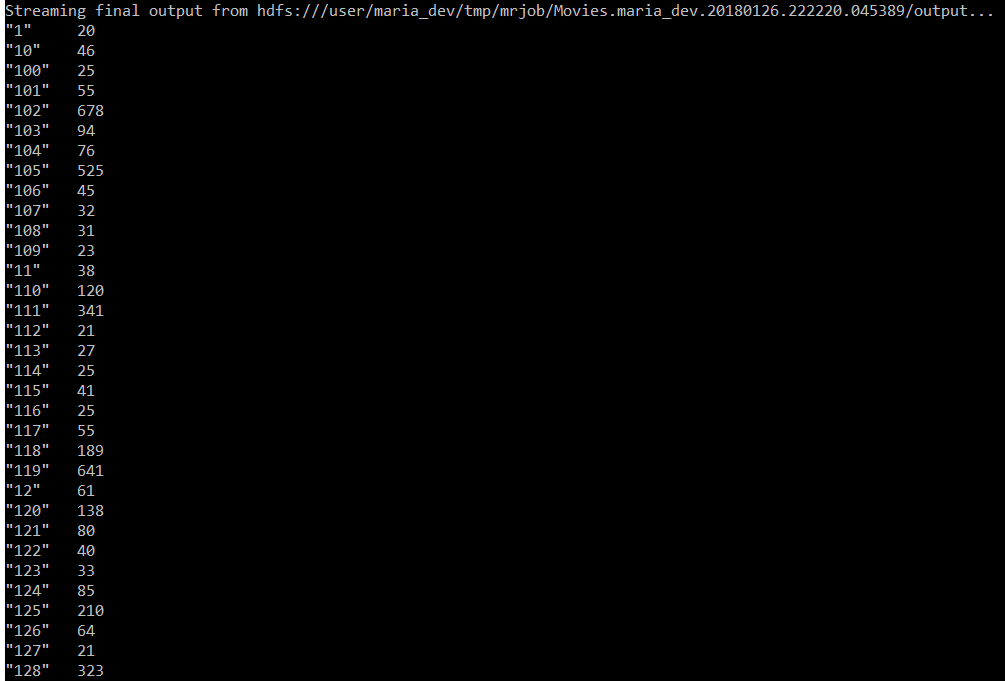
if \_\_name\_\_ == '\_\_main\_\_':

MRMoviesReviewed.run()

**Execution Command:**

****

**Output Result:**

****