# Assignment: Running Wordcount with Hadoop streaming, using Python code

You have not submitted. You must earn 1/1 points to pass.

**Deadline** Pass this assignment by January 10, 11:59 PM PT

## Lesson 1, Introduction to Map/Reduce Module, Running Wordcount with streaming, using Python code

This assignment will build upon the assignment at the end of previous course. You will examine the options for streaming that control the number of reducers. In the Cloudera Quickstart VM terminal, follow the instructions below to execute a simple word count example in Python. There will be one file to upload, and quiz questions about word count results with different numbers of reducers.

## 1. Open a Terminal (Right-click on Desktop or click Terminal icon in the top toolbar) 2. Review the following to create the python code

Section 1: wordcount_mapper.py

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

```
#the above just indicates to use python to intepret this
file

# ---------------------------------------------------------
--------
#This mapper code will input a line of text and output <w
ord, 1>
#
# ---------------------------------------------------------
--------

import sys             #a python module with system funct
ions for this OS

# ---------------------------------------------------------
-----
#  this 'for loop' will set 'line' to an input line from 
system
#    standard input file
# ---------------------------------------------------------
-----
for line in sys.stdin:

#-----------------------------------
#sys.stdin call 'sys' to read a line from standard input,

# note that 'line' is a string object, ie variable, and i
t has methods that you can apply to it,
# as in the next line
# ---------------------------------
    line = line.strip()  #strip is a method, ie function,
 associated
                         #  with string variable, it will
 strip
                         #   the carriage return (by defa
ult)
    keys = line.split()  #split line at blanks (by defaul
t),
                         #   and return a list of keys
    for key in keys:     #a for loop through the list of 
keys
        value = 1
        print('{0}\t{1}'.format(key, value) ) #the {} is 
replaced by 0th,1st items in format list
                            #also, note that the Hadoop d
efault is 'tab' separates key from the value
```

## Section 2: wordcount_reducer.py

The reducer code has some basic parts, see the comments in the code. The Lesson 2 assignment will have a similar basic structure.

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

# ---------------------------------------------------------------
--------
#This reducer code will input a line of text and
#    output <word, total-count>
# ---------------------------------------------------------------
--------
import sys

last_key      = None              #initialize these varia
bles
running_total = 0

# -------------------------------------
# Loop thru file
#  --------------------------------
for input_line in sys.stdin:
    input_line = input_line.strip()

    # --------------------------------
    # Get Next Word    # --------------------------------
    this_key, value = input_line.split("\t", 1)  #the Had
oop default is tab separates key value
                          #the split command returns a li
st of strings, in this case into 2 variables
    value = int(value)           #int() will convert a st
ring to integer (this program does no error checking)

    # ---------------------------------
    # Key Check part
    #    if this current key is same
    #          as the last one Consolidate
    #    otherwise  Emit
    # ---------------------------------
    if last_key == this_key:     #check if key has change
d ('==' is                                #      logic
al equalilty check
        running_total += value   # add value to running t
otal

    else:
        if last_key:             #if this key that was ju
```

```
st read in
                                 #   is different, and th
e previous
                                 #   (ie last) key is not
 empy,
                                 #   then output
                                 #   the previous <key ru
nning-count>
            print( "{0}\t{1}".format(last_key, running_to
tal) )
                                 # hadoop expects tab(ie
'\t')
                                 #    separation
        running_total = value    #reset values
        last_key = this_key

if last_key == this_key:
    print( "{0}\t{1}".format(last_key, running_total))
```

**NOTE: If you have not programmed with Python please read the following:**

```
Python notes:
#    1 indentations are required to indicate blocks of co
de,
#    2  all code to be executed as part of some flow cont
rol
#          (e.g. if or for statements) must have the same
 indentation
#          (to be safe use 4 space per indentation level,
 and don't
#            mix with tabs)
#    3 flow control conditions have a ':' before
#          the corresponding block of code
#
```

You can cut and paste the above into a text file as follows from the terminal prompt in Cloudera VM.

Type in the following to open a text editor, and then cut and paste the above lines for wordcount_mapper.py into the text editor, save, and exit. Repeat for wordcount_reducer.py

> gedit wordcount_mapper.py

> gedit wordcount_reducer.py

Enter the following to see that the indentations line up as above

> more wordcount_mapper.py

> more wordcount_reducer.py

Enter the following to make it executable

> chmod +x wordcount_mapper.py

> chmod +x wordcount_reducer.py

Enter the following to see what directory you are in

> pwd

It should be /user/cloudera , or something like that.

## 3. Create some data:

> echo "A long time ago in a galaxy far far away" > /home/cloudera/testfile1

> echo "Another episode of Star Wars" > /home/cloudera/testfile2

## 4. Create a directory on the HDFS file system (if already exists that's OK):

hdfs dfs -mkdir /user/cloudera/input

## 5. Copy the files from local filesystem to the

## HDFS filesystem:

hdfs dfs -put /home/cloudera/testfile1 /user/cloudera/input

hdfs dfs -put /home/cloudera/testfile2 /user/cloudera/input

## 6. You can see your files on HDFS

hdfs dfs -ls /user/cloudera/input

## 7. Run the Hadoop WordCount example with the input and output specified.

## Note that your file paths may differ. The '\' just means the command continues on next line.

```
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar
 \
   -input /user/cloudera/input \
   -output /user/cloudera/output_new \
   -mapper /home/cloudera/wordcount_mapper.py \
   -reducer /home/cloudera/wordcount_reducer.py
```

Hadoop prints out a whole lot of logging or error information. If it runs you will see something like the following on the screen scroll by:

```
....

INFO mapreduce.Job: map 0% reduce 0%

INFO mapreduce.Job: map 67% reduce 0%

INFO mapreduce.Job: map 100% reduce 0%

INFO mapreduce.Job: map 100% reduce 100%

INFO mapreduce.Job: Job job_1442937183788_0003 completed successfully

...
```

## 8. Check the output file to see the results:

hdfs dfs -cat /user/cloudera/output_new/part-00000

## 9. View the output directory:

hdfs dfs -ls /user/cloudera/output_new

Look at the files there and check out the contents, e.g.:

hdfs dfs -cat /user/cloudera/output_new/part-00000

## 10. Streaming options:

Try: hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar --help

or see hadoop.apache.org/docs/r1.2.1/

Let’s change the number of reduce tasks to see its effects. Setting it to 0 will execute no reducer and only produce the map output. (Note the output directory is changed in the snippet below because Hadoop doesn’t like to overwrite output)

```
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar
 \
   -input /user/cloudera/input \
   -output /user/cloudera/output_new_0 \
   -mapper /home/cloudera/wordcount_mapper.py \
   -reducer /home/cloudera/wordcount_reducer.py \
   -numReduceTasks 0
```

Get the output file from this run, and then upload it:

> hdfs dfs -getmerge /user/cloudera/output_new_0/* wordcount_num0_output.txt

## 11. Change the number of reducers to 2 and answer the related quiz question at the end of the video lesson

## How to submit

When you're ready to submit, you can upload files for each part of the assignment on the "My submission" tab.