Vicente De Leon

Assignment 4

Big Data Applications

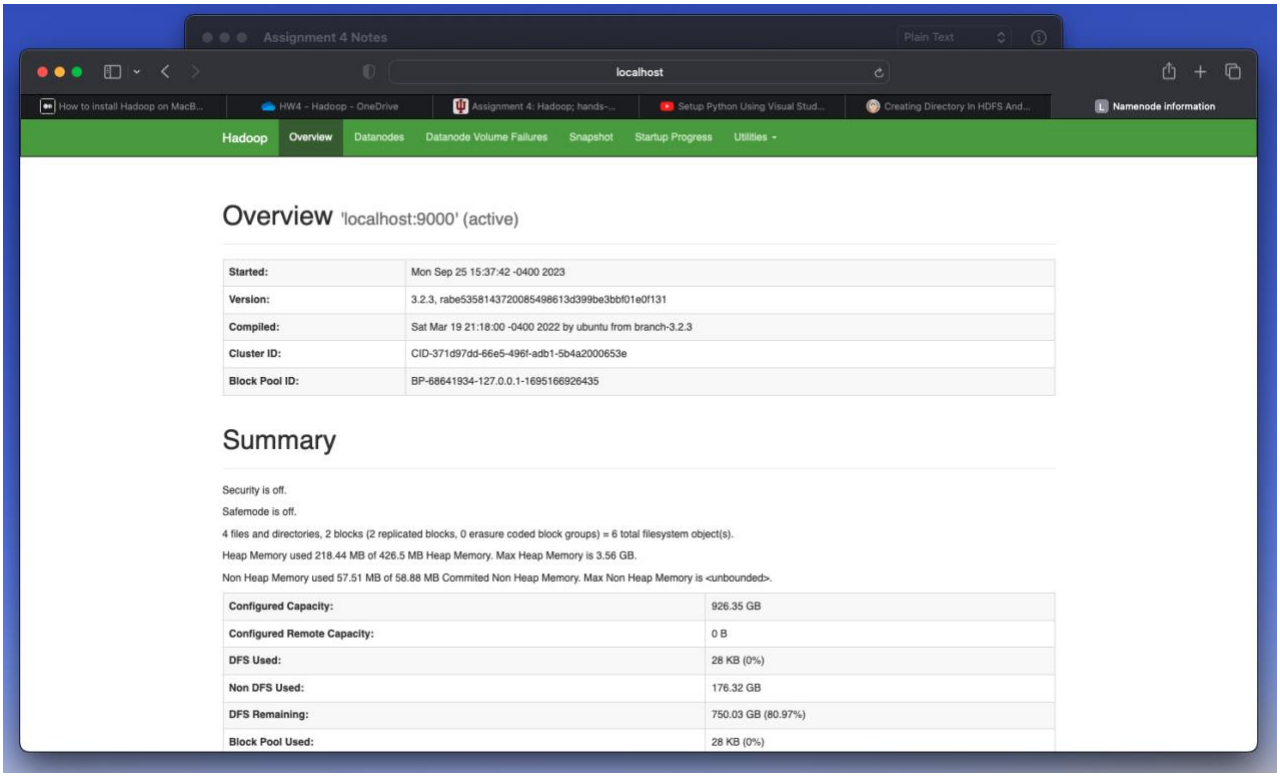
Indiana University

Assignment 4 – HDFS Commands

Part 1: Task 1 to Task 9 HDFS Commands Mac Terminal. I will also upload a text file to final submission, which contains all the commands I used for this section.

All commands came from: https://sparkbyexamples.com/apache-hadoop/hadoop-hdfs-dfs-commands-and-starting-hdfs-dfs-services/

```
1  Assignment 4 by Vicente De Leon
2
3  Present Hadoop notes installation
4
5  start: start-all.sh
6  go to: http://localhost:9870/dfshealth.html#tab-overview
```

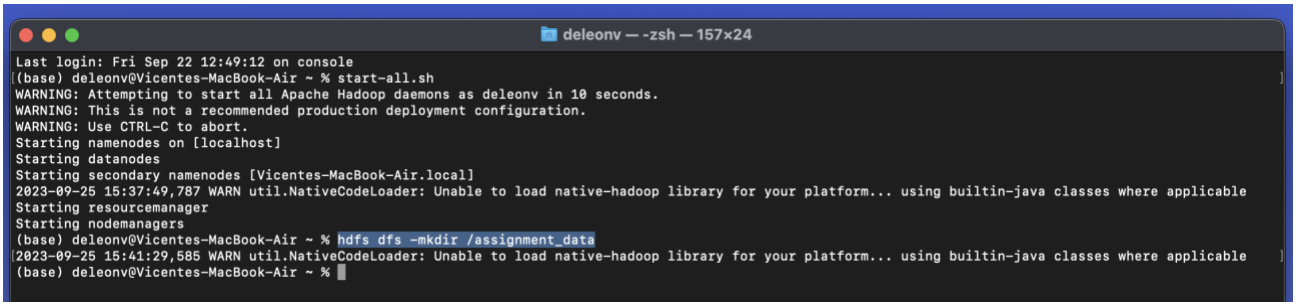


Task1: Create a directory (“assignment_data”) on HDFS using mkdir command:

Source: https://tecadmin.net/working-with-hdfs-file-system/

Source: https://www.quora.com/How-will-you-check-if-a-file-exists-in-HDFS

```
1) Task 1 (Creating directory):
Soruce: https://tecadmin.net/working-with-hdfs-file-system/
Source: https://www.quora.com/How-will-you-check-if-a-file-exists-in-HDFS

type: hdfs dfs -mkdir /assignment_data
to check if it exists: hdfs dfs -ls /
```



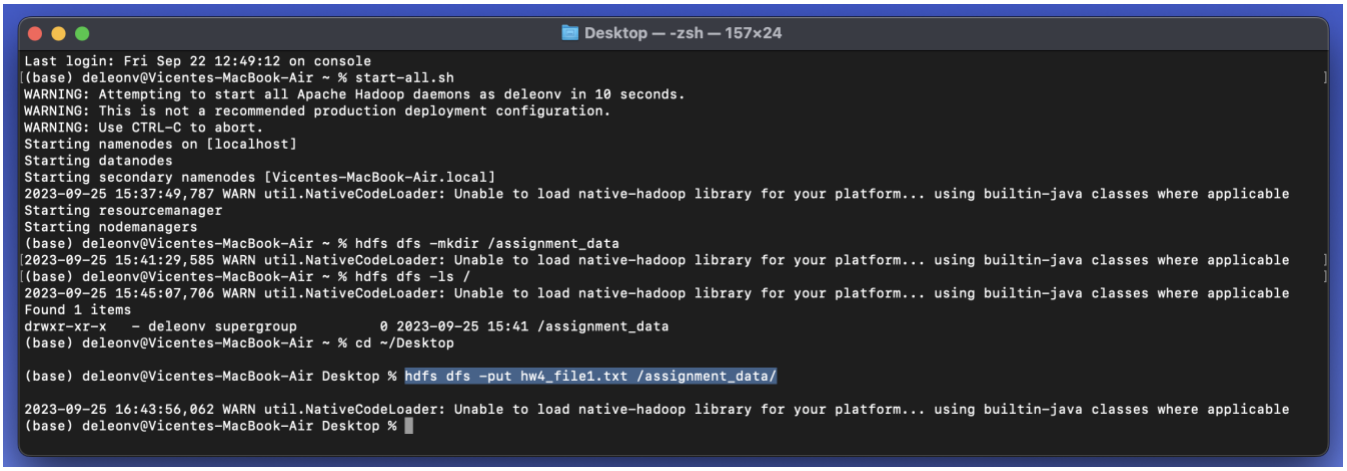
Task 2: Upload files to HDFS (Using HDFS put command:

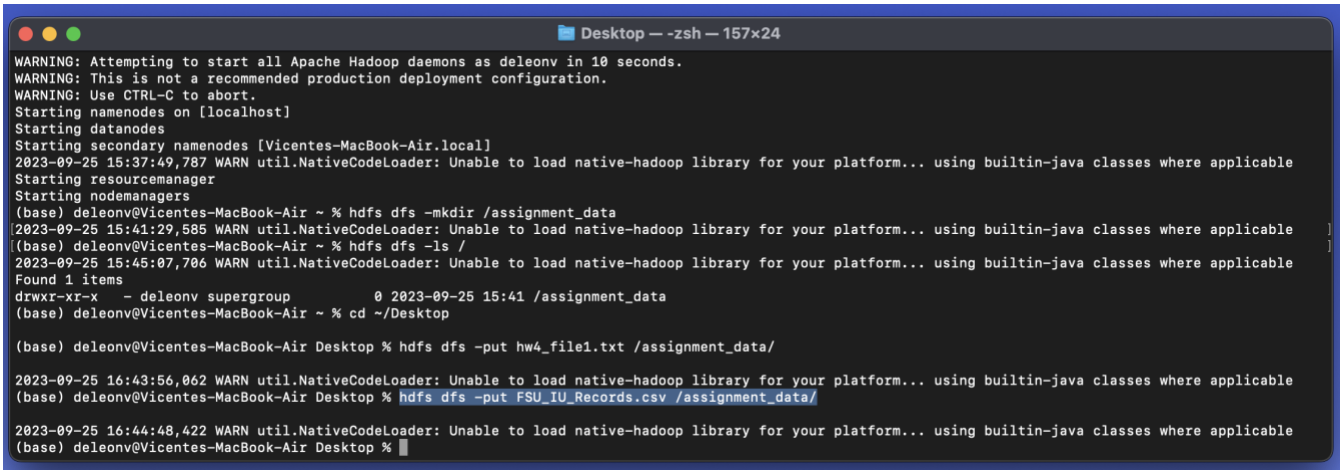
Source: https://tecadmin.net/working-with-hdfs-file-system/

Source: https://sparkbyexamples.com/apache-hadoop/hadoop-how-to-list-files-and-directories-using-hdfs-dfs/

I decided to create 2 random files (1 text file: **“hw4_file1.txt”** and 1 csv file: **“FSU_IU_Records.csv**”) for Task 2. You can see how I created both files by going to the Python Notebook (which I’m also submitting to the final submission) “**Hadoop_HW4.ipynb**”. I’m also submitting both files as well.

```
2) Task 2 (Upload files to HDFS directory)
Source: https://tecadmin.net/working-with-hdfs-file-system/
Source: https://sparkbyexamples.com/apache-hadoop/hadoop-how-to-list-files-and-directories-using-hdfs-dfs/

type: cd ~/Desktop (because I have both sample files in my Desktop)
type: hdfs dfs -put hw4_file1.txt /assignment_data/ (to add txt file into directory)
type: hdfs dfs -put FSU_IU_Records.csv /assignment_data/ (to add csv file into directory)
```

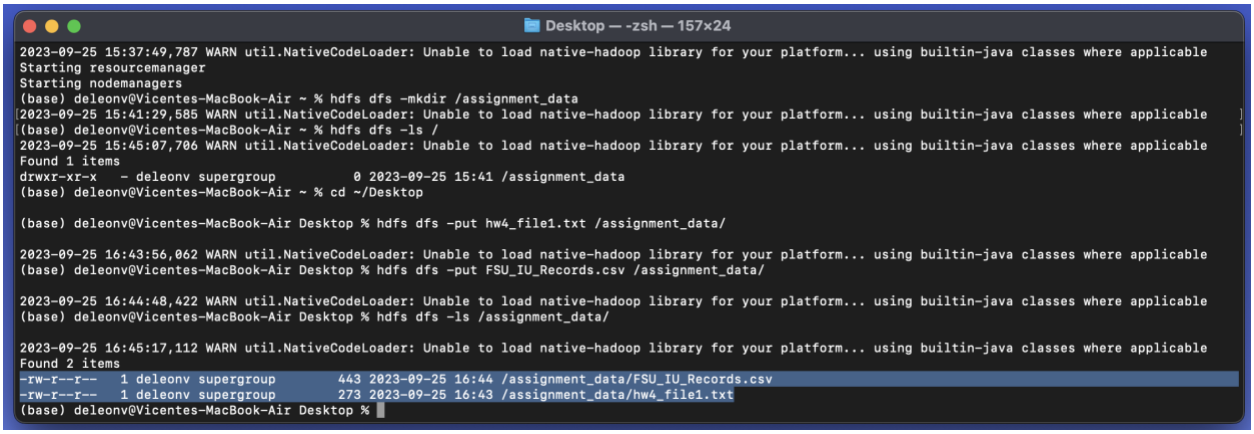




Task 3: List files in HDFS (listing all the files in the HDFS directory using ls command):

Source: https://www.quora.com/How-will-you-check-if-a-file-exists-in-HDFS

```
3) Task 3: Checking if these files exists
Source: Source: https://www.quora.com/How-will-you-check-if-a-file-exists-in-HDFS
type: hdfs dfs -ls /assignment_data/ (to check if those two files exist within directory. Just like when you use -ls command in Mac Terminal
after pwd and cd to acces files within Desktop.)
```

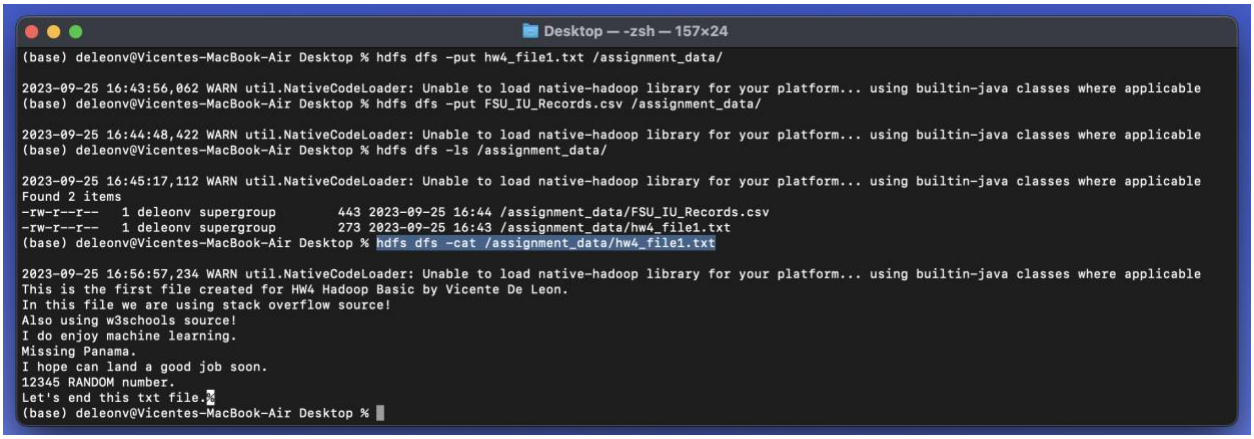


Task 4: View File content (using HDFS cat command)

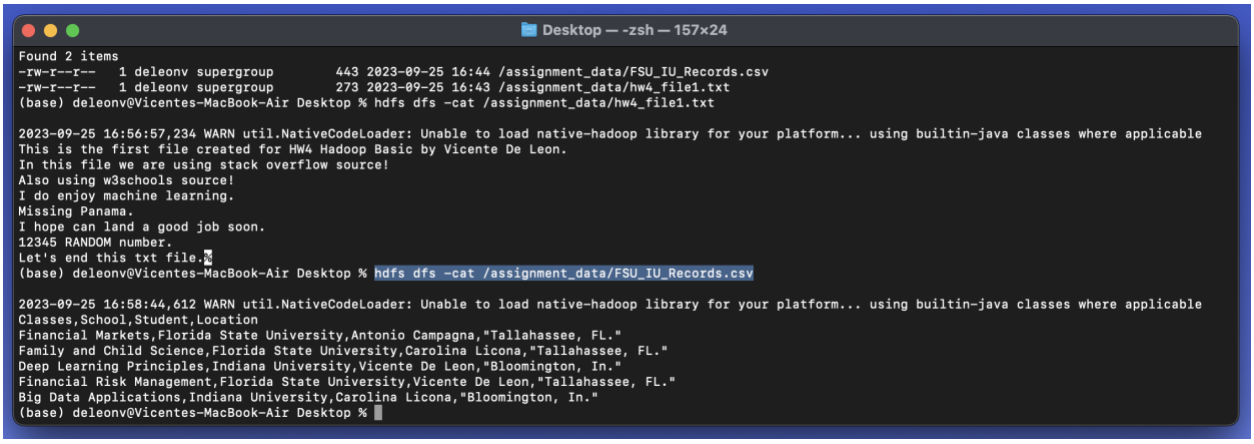
Source https://sparkbyexamples.com/apache-hadoop/hadoop-hdfs-dfs-commands-and-starting-hdfs-dfs-services/

```
4) Task 4: View File Content:
Source: https://sparkbyexamples.com/apache-hadoop/hadoop-hdfs-dfs-commands-and-starting-hdfs-dfs-services/

type: hdfs dfs -cat /assignment_data/hw4_file1.txt (display file 1 in MAC Terminal)
type: hdfs dfs -cat /assignment_data/FSU_IU_Records.csv
```

Inside “hw4_file1.txt” (using HDFS cat command):

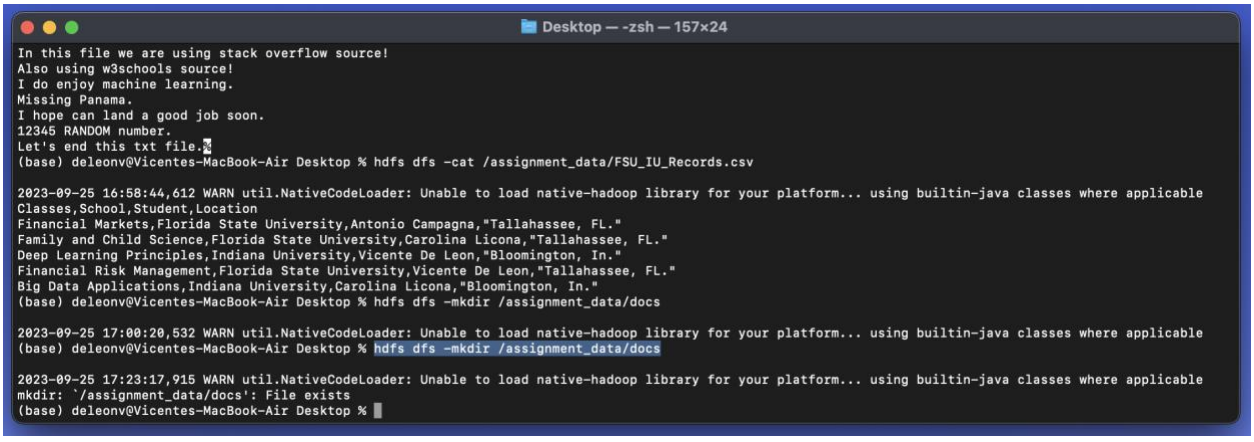


Inside “FSU_IU_Records.csv” (using HDFS cat command):



Task 5: Create a new directory in HDFS (New Subdirectory called “docs” within “assigntment_data” directory using HDFS mkdir command).

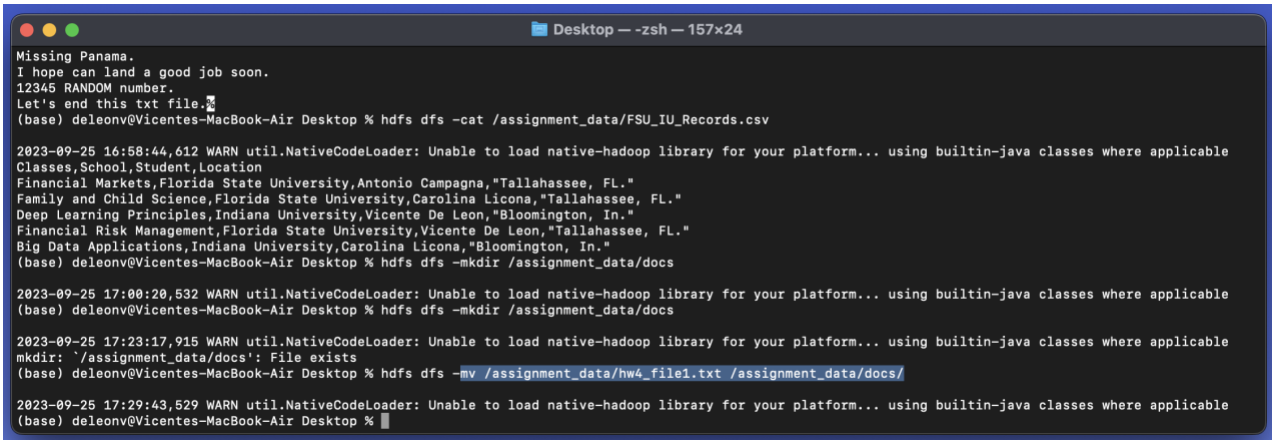
```
37  5) Task 5: Creat subdirectory
38  Source: https://sparkbyexamples.com/apache-hadoop/hadoop-hdfs-dfs-commands-and-starting-hdfs-dfs-services/
39  Source: Soruce: https://tecadmin.net/working-with-hdfs-file-system/
40
41  type: hdfs dfs -mkdir /assignment_data/docs
```

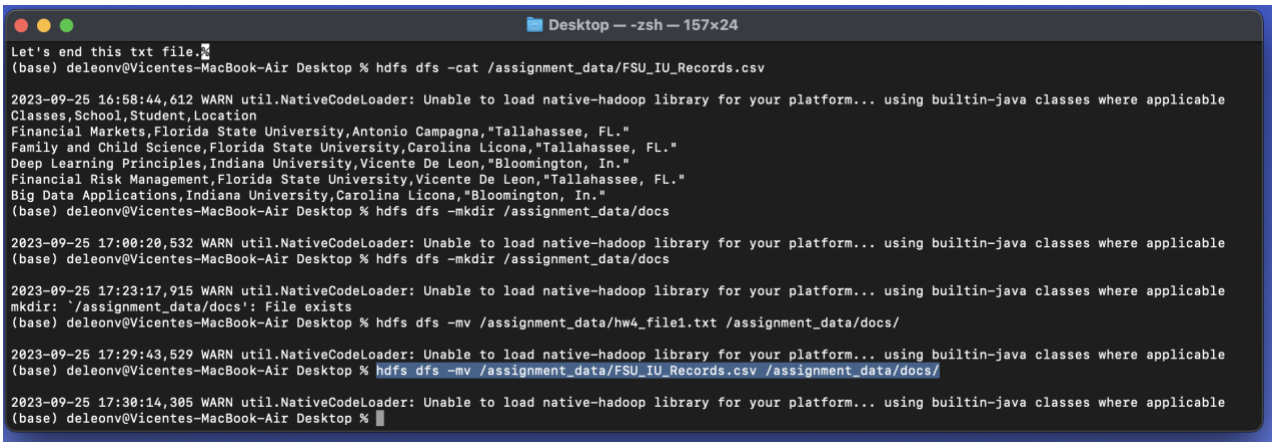


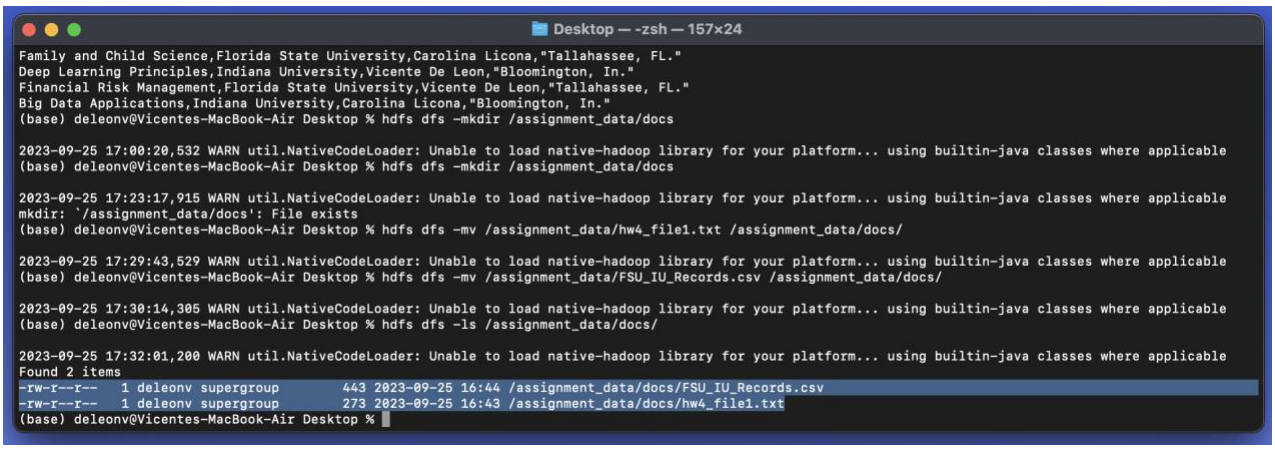
Task 6: Move files to a different directory in HDFS using mv command.

Source: https://sparkbyexamples.com/apache-hadoop/hadoop-hdfs-dfs-commands-and-starting-hdfs-dfs-services/

```
6) Task 6: Move Files
Source: https://sparkbyexamples.com/apache-hadoop/hadoop-hdfs-dfs-commands-and-starting-hdfs-dfs-services/
-mv is used to move files from source to destination

type: hdfs dfs -mv /assignment_data/hw4_file1.txt /assignment_data/docs/
type: hdfs dfs -mv /assignment_data/FSU_IU_Records.csv /assignment_data/docs/
```







Task 7: Delete files from HDFS (Removing all content from directory “docs” using rm command).

```
7) Task 7: Delet Files from HDFS
-rm is used to Remove File or a Directory

type:  hdfs dfs -rm /assignment_data/docs/hw4_file1.txt
type: hdfs dfs -rm /assignment_data/docs/FSU_IU_Records.csv
```

```
Desktop — -zsh — 157×24
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -mkdir /assignment_data/docs

2023-09-25 17:00:20,532 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -mkdir /assignment_data/docs

2023-09-25 17:23:17,915 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
mkdir: `/assignment_data/docs': File exists
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -mv /assignment_data/hw4_file1.txt /assignment_data/docs/

2023-09-25 17:29:43,529 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -mv /assignment_data/FSU_IU_Records.csv /assignment_data/docs/

2023-09-25 17:30:14,305 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -ls /assignment_data/docs/

2023-09-25 17:32:01,200 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r--   1 deleonv supergroup        443 2023-09-25 16:44 /assignment_data/docs/FSU_IU_Records.csv
-rw-r--r--   1 deleonv supergroup        273 2023-09-25 16:43 /assignment_data/docs/hw4_file1.txt
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -rm /assignment_data/docs/hw4_file1.txt

2023-09-25 17:36:00,612 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Deleted /assignment_data/docs/hw4_file1.txt
(base) deleonv@Vicentes-MacBook-Air Desktop %
```

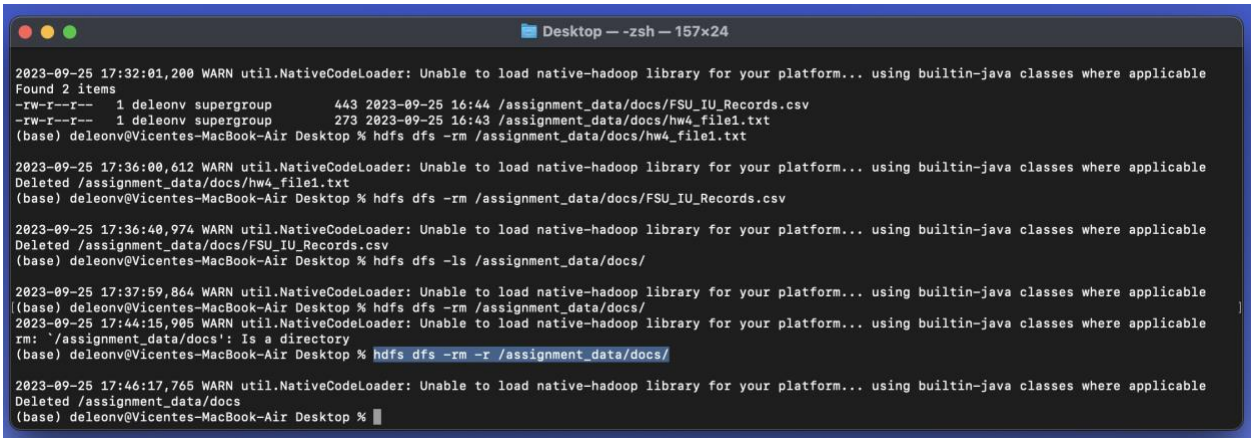
```
Desktop — -zsh — 157×24
2023-09-25 17:23:17,915 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
mkdir: `/assignment_data/docs': File exists
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -mv /assignment_data/hw4_file1.txt /assignment_data/docs/

2023-09-25 17:29:43,529 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -mv /assignment_data/FSU_IU_Records.csv /assignment_data/docs/

2023-09-25 17:30:14,305 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -ls /assignment_data/docs/

2023-09-25 17:32:01,200 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r--   1 deleonv supergroup        443 2023-09-25 16:44 /assignment_data/docs/FSU_IU_Records.csv
-rw-r--r--   1 deleonv supergroup        273 2023-09-25 16:43 /assignment_data/docs/hw4_file1.txt
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -rm /assignment_data/docs/hw4_file1.txt

2023-09-25 17:36:00,612 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Deleted /assignment_data/docs/hw4_file1.txt
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -rm /assignment_data/docs/FSU_IU_Records.csv

2023-09-25 17:36:40,974 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Deleted /assignment_data/docs/FSU_IU_Records.csv
(base) deleonv@Vicentes-MacBook-Air Desktop %
```

```
Desktop — -zsh — 157×24
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -mv /assignment_data/hw4_file1.txt /assignment_data/docs/

2023-09-25 17:29:43,529 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -mv /assignment_data/FSU_IU_Records.csv /assignment_data/docs/

2023-09-25 17:30:14,305 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -ls /assignment_data/docs/

2023-09-25 17:32:01,200 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r--   1 deleonv supergroup        443 2023-09-25 16:44 /assignment_data/docs/FSU_IU_Records.csv
-rw-r--r--   1 deleonv supergroup        273 2023-09-25 16:43 /assignment_data/docs/hw4_file1.txt
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -rm /assignment_data/docs/hw4_file1.txt

2023-09-25 17:36:00,612 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Deleted /assignment_data/docs/hw4_file1.txt
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -rm /assignment_data/docs/FSU_IU_Records.csv

2023-09-25 17:36:40,974 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Deleted /assignment_data/docs/FSU_IU_Records.csv
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -ls /assignment_data/docs/

2023-09-25 17:37:59,864 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
(base) deleonv@Vicentes-MacBook-Air Desktop %
```

Task 8: Delete a directory from HDFS (using HDFS rm and r commands).

Source: https://stackoverflow.com/questions/13529114/how-to-delete-a-directory-from-hadoop-cluster-which-is-having-comma-in-its-na

```
8) Task 8: Deleting Directory from HDFS
Source: https://stackoverflow.com/questions/13529114/how-to-delete-a-directory-from-hadoop-cluster-which-is-having-comma-in-its-na

type: hdfs dfs -rm -r /assignment_data/docs/
```



Task 9: Check HDFS file status.

Let's add "FSU_IU_Recods.csv" back to assignment_data HDFS directory to analyze it using:

- **Stat %b command to check on byte size.**
- **Ls command to check replication factor.**
- **Fsck, files, and blocks command to check on block location.**

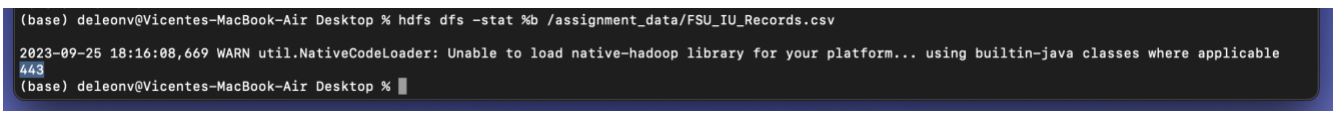
Byte size (answer -> 443):

Source: https://sparkbyexamples.com/apache-hadoop/hadoop-hdfs-dfs-commands-and-starting-hdfs-dfs-services/

Source: https://www.edureka.co/community/6712/hadoop-fs-stat-command

Source: https://community.pivotal.io/s/article/Understanding-format-options-for-hdfs--stat-command?language=en_US

```
9) Task 9: HDFS File status

Just added: hdfs dfs -put FSU_IU_Records.csv /assignment_data/ (to check status size, replication, block location)

Byte size:
Source: https://sparkbyexamples.com/apache-hadoop/hadoop-hdfs-dfs-commands-and-starting-hdfs-dfs-services/
Source: https://www.edureka.co/community/6712/hadoop-fs-stat-command
Source: https://community.pivotal.io/s/article/Understanding-format-options-for-hdfs--stat-command?language=en_US

type: hdfs dfs -stat %b /assignment_data/FSU_IU_Records.csv (returns 443)
```



Replication factor (answer -> replication factor of 1):

Source: https://community.pivotal.io/s/article/Understanding-format-options-for-hdfs--stat-command?language=en_US

Source: https://stackoverflow.com/questions/25166926/how-do-you-retrieve-the-replication-factor-info-in-hdfs-files

Source: https://www.projectpro.io/recipes/run-hdfs-filesystem-checking-utility

```
Replication factor:
Source: https://community.pivotal.io/s/article/Understanding-format-options-for-hdfs--stat-command?language=en_US
Source: https://stackoverflow.com/questions/25166926/how-do-you-retrieve-the-replication-factor-info-in-hdfs-files
Source: https://www.projectpro.io/recipes/run-hdfs-filesystem-checking-utility

type: hdfs dfs -ls /assignment_data/FSU_IU_Records.csv (returns replication fatcor of 1. There is just one copy of the file within HDFS)
```

```
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -ls /assignment_data/FSU_IU_Records.csv

2023-09-25 18:20:41,844 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
-rw-r--r--   1 deleonv supergroup        443 2023-09-25 17:52 /assignment_data/FSU_IU_Records.csv
(base) deleonv@Vicentes-MacBook-Air Desktop %
```

Block Location (/assignment_data/FSU_IU_Records.cv' is HEALTHY**):**

block locations for "FSU_IU_Records.csv": BP-68641934-127.0.0.1-1695166926435:blk_1073741827_1003 len=443 Live_repl=1

Source: https://www.projectpro.io/recipes/run-hdfs-filesystem-checking-utility

Source: https://stackoverflow.com/questions/11168427/viewing-the-number-of-blocks-for-a-file-in-hadoop

You can also use this code to view byte size -> 443 B, replication factor:1, and block locations.

```
Block Location:
Source: https://www.projectpro.io/recipes/run-hdfs-filesystem-checking-utility
Source: https://stackoverflow.com/questions/11168427/viewing-the-number-of-blocks-for-a-file-in-hadoop

type: hdfs fsck /assignment_data/FSU_IU_Records.csv -files -blocks (you can also see byte size: 443 B, replication factor: 1, and block locations.)
block locations for "FSU_IU_Records.csv": BP-68641934-127.0.0.1-1695166926435:blk_1073741827_1003 len=443 Live_repl=1
```

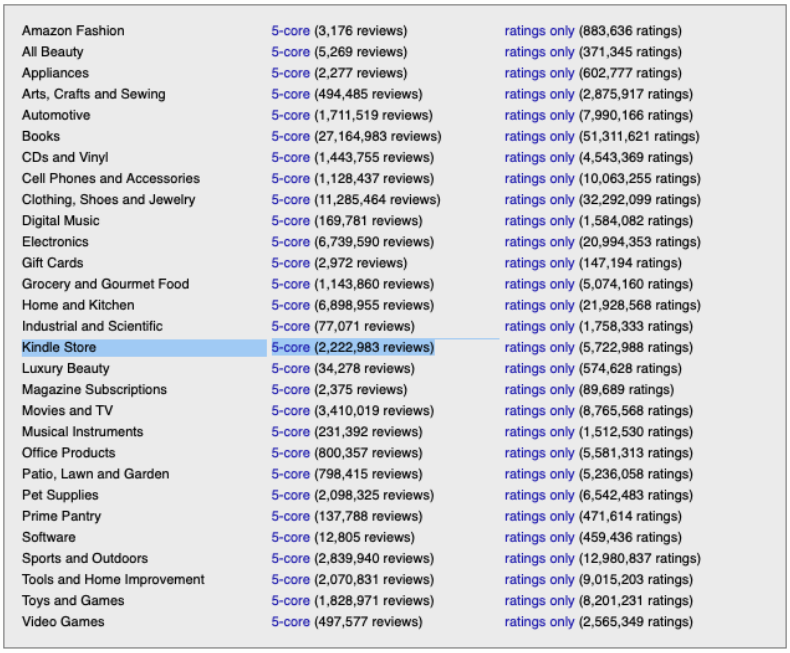
```
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs fsck /assignment_data/FSU_IU_Records.csv -files -blocks

2023-09-25 18:27:24,135 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Connecting to namenode via http://localhost:9870/fsck?ugi=deleonv&files=1&blocks=1&path=%2Fassignment_data%2FFSU_IU_Records.csv
FSCK started by deleonv (auth:SIMPLE) from /127.0.0.1 for path /assignment_data/FSU_IU_Records.csv at Mon Sep 25 18:27:25 EDT 2023

/assignment_data/FSU_IU_Records.csv 443 bytes, replicated: replication=1, 1 block(s):  OK
0. BP-68641934-127.0.0.1-1695166926435:blk_1073741827_1003 len=443 Live_repl=1


Status: HEALTHY
 Number of data-nodes:  1
 Number of racks:               1
 Total dirs:                    0
 Total symlinks:                0

Replicated Blocks:
 Total size:    443 B
 Total files:   1
 Total blocks (validated):      1 (avg. block size 443 B)
 Minimally replicated blocks:   1 (100.0 %)
 Over-replicated blocks:        0 (0.0 %)
 Under-replicated blocks:       0 (0.0 %)
 Mis-replicated blocks:         0 (0.0 %)
 Default replication factor:    1
 Average block replication:     1.0
 Missing blocks:                0
 Corrupt blocks:                0
 Missing replicas:              0 (0.0 %)

Erasure Coded Block Groups:
 Total size:    0 B
 Total files:   0
 Total block groups (validated):        0
 Minimally erasure-coded block groups:  0
 Over-erasure-coded block groups:       0
 Under-erasure-coded block groups:      0
 Unsatisfactory placement block groups: 0
 Average block group size:      0.0
 Missing block groups:          0
 Corrupt block groups:          0
 Missing internal blocks:       0
FSCK ended at Mon Sep 25 18:27:25 EDT 2023 in 9 milliseconds


The filesystem under path '/assignment_data/FSU_IU_Records.csv' is HEALTHY
```

Part 2: Task 10 to Task 12 MapReduce jobs to process and analyze data. These are the scripts needed:

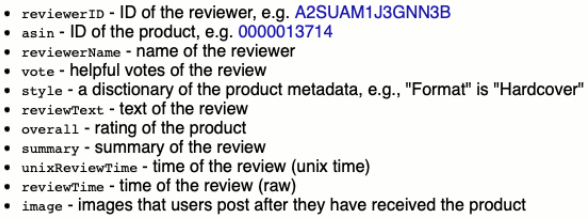
These commands are the HDFS commands I need to successfully do this homework: https://sparkbyexamples.com/apache-hadoop/hadoop-hdfs-dfs-commands-and-starting-hdfs-dfs-services/

Task 10: Dataset Review (brief description, file format size, and the type of data it contains)

- Source: https://cseweb.ucsd.edu/~jmcauley/datasets/amazon_v2/
- In the above source you will find a section that says, "Small Subsets for Experimentation." This section advises us to work with a smaller dataset in case we are working on a class project or school related experimentations. So, for the purpose of this homework, I will be using "k-cores" dataset to tackle task 10, task 11, and task 12. The data selected was "Kindle_Store_5.json" a 5-core data containing 2.223M reviews in total. Since "assignment_data" HDFS directory was created for Tasks 1 to 9 let's use it again.



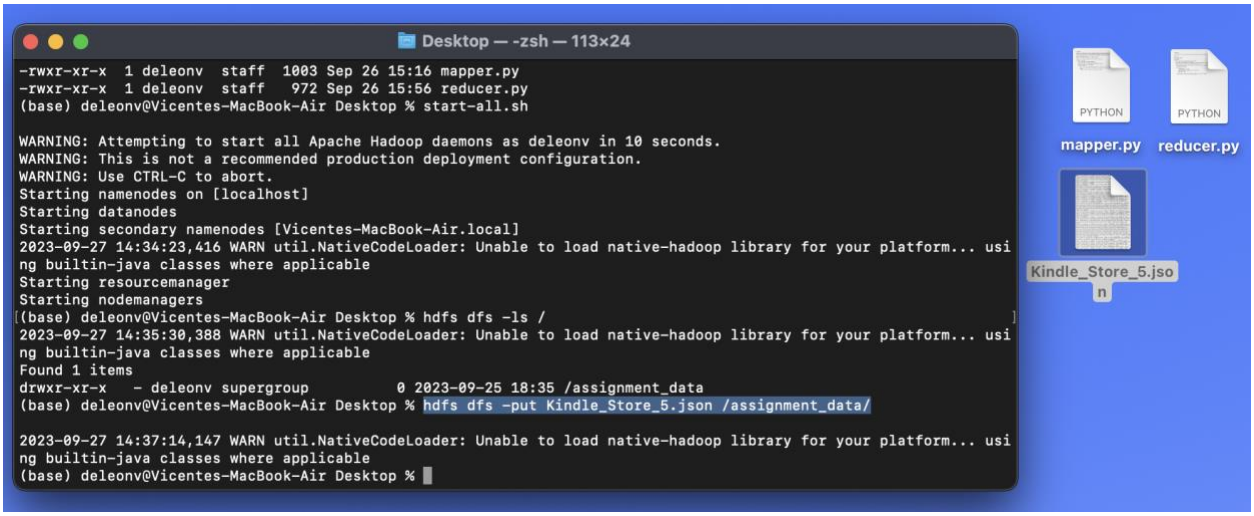
Amazon Fashion	5-core (3,176 reviews)	ratings only (883,636 ratings)
All Beauty	5-core (5,269 reviews)	ratings only (371,345 ratings)
Appliances	5-core (2,277 reviews)	ratings only (602,777 ratings)
Arts, Crafts and Sewing	5-core (494,485 reviews)	ratings only (2,875,917 ratings)
Automotive	5-core (1,711,519 reviews)	ratings only (7,990,166 ratings)
Books	5-core (27,164,983 reviews)	ratings only (51,311,621 ratings)
CDs and Vinyl	5-core (1,443,755 reviews)	ratings only (4,543,369 ratings)
Cell Phones and Accessories	5-core (1,128,437 reviews)	ratings only (10,063,255 ratings)
Clothing, Shoes and Jewelry	5-core (11,285,464 reviews)	ratings only (32,292,099 ratings)
Digital Music	5-core (169,781 reviews)	ratings only (1,584,082 ratings)
Electronics	5-core (6,739,590 reviews)	ratings only (20,994,353 ratings)
Gift Cards	5-core (2,972 reviews)	ratings only (147,194 ratings)
Grocery and Gourmet Food	5-core (1,143,860 reviews)	ratings only (5,074,160 ratings)
Home and Kitchen	5-core (6,898,955 reviews)	ratings only (21,928,568 ratings)
Industrial and Scientific	5-core (77,071 reviews)	ratings only (1,758,333 ratings)
Kindle Store	5-core (2,222,983 reviews)	ratings only (5,722,988 ratings)
Luxury Beauty	5-core (34,278 reviews)	ratings only (574,628 ratings)
Magazine Subscriptions	5-core (2,375 reviews)	ratings only (89,689 ratings)
Movies and TV	5-core (3,410,019 reviews)	ratings only (8,765,568 ratings)
Musical Instruments	5-core (231,392 reviews)	ratings only (1,512,530 ratings)
Office Products	5-core (800,357 reviews)	ratings only (5,581,313 ratings)
Patio, Lawn and Garden	5-core (798,415 reviews)	ratings only (5,236,058 ratings)
Pet Supplies	5-core (2,098,325 reviews)	ratings only (6,542,483 ratings)
Prime Pantry	5-core (137,788 reviews)	ratings only (471,614 ratings)
Software	5-core (12,805 reviews)	ratings only (459,436 ratings)
Sports and Outdoors	5-core (2,839,940 reviews)	ratings only (12,980,837 ratings)
Tools and Home Improvement	5-core (2,070,831 reviews)	ratings only (9,015,203 ratings)
Toys and Games	5-core (1,828,971 reviews)	ratings only (8,201,231 ratings)
Video Games	5-core (497,577 reviews)	ratings only (2,565,349 ratings)



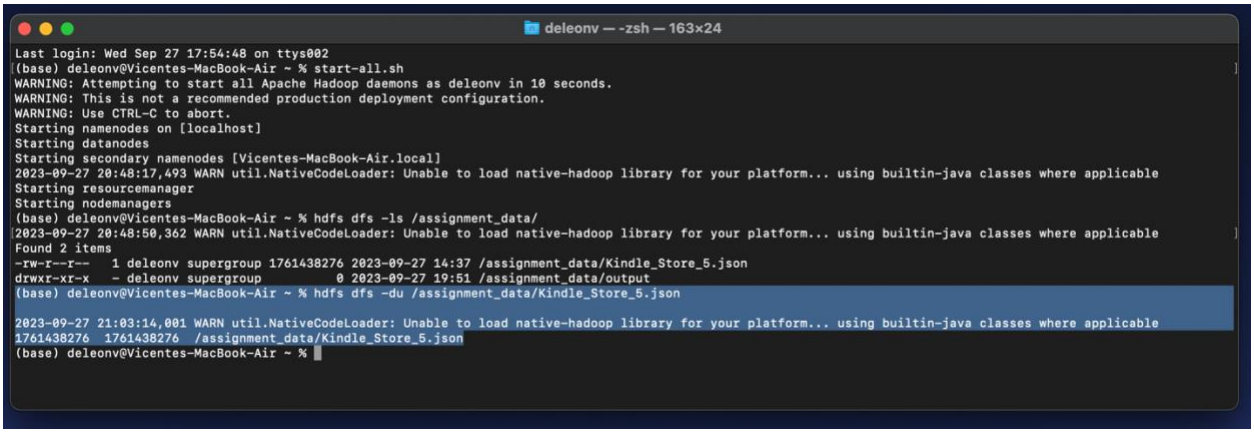
- `reviewerID` - ID of the reviewer, e.g. A2SUAM1J3GNN3B
- `asin` - ID of the product, e.g. 0000013714
- `reviewerName` - name of the reviewer
- `vote` - helpful votes of the review
- `style` - a disctionary of the product metadata, e.g., "Format" is "Hardcover"
- `reviewText` - text of the review
- `overall` - rating of the product
- `summary` - summary of the review
- `unixReviewTime` - time of the review (unix time)
- `reviewTime` - time of the review (raw)
- `image` - images that users post after they have received the product

- I will use the get command "hdfs dfs -du" to get or show the size of the file on HDFS. In this case, the file size is 1.76GB. The above images shows where the JSON data came from. It also shows some of the attributes the JSON file has. Just by looking at the first 5 lines of the Kindle_Store_5.json" file I can see it is a collection of JSON objects. To have a better view of this JSON objects please go to the Hadoop_HW4.ipynb Notebook where I use python to print those first 5 lines. Please, check the image below to view code for the Mac Terminal.

You can see how I started Hadoop services and how initially I added “Kindle_Store_5.json” to HDFS directory (I did Task 11 first before completing Task 10 – Please see highlighted code):



In the below image I checked if Kindle JSON file was within the HDFS directory. To check byte size, I used the -du command (Please see highlighted code):




```
Task 10)
type: start-all.sh (starting hadoop services)
Since assignment_data HDFS directory was created for Tasks 1 to 9 let's use it.
type: hdfs dfs -put Kindle_Store_5.json /assignment_data/ (how I added the JSON file to directory)
type: hdfs dfs -ls /assignment_data/ (to check if "Kindle_Store_5.json" file is within the HDFS directory)
type: hdfs dfs -du /assignment_data/Kindle_Store_5.json (check file size -> 1,761,438,276 = 1.76GB
```

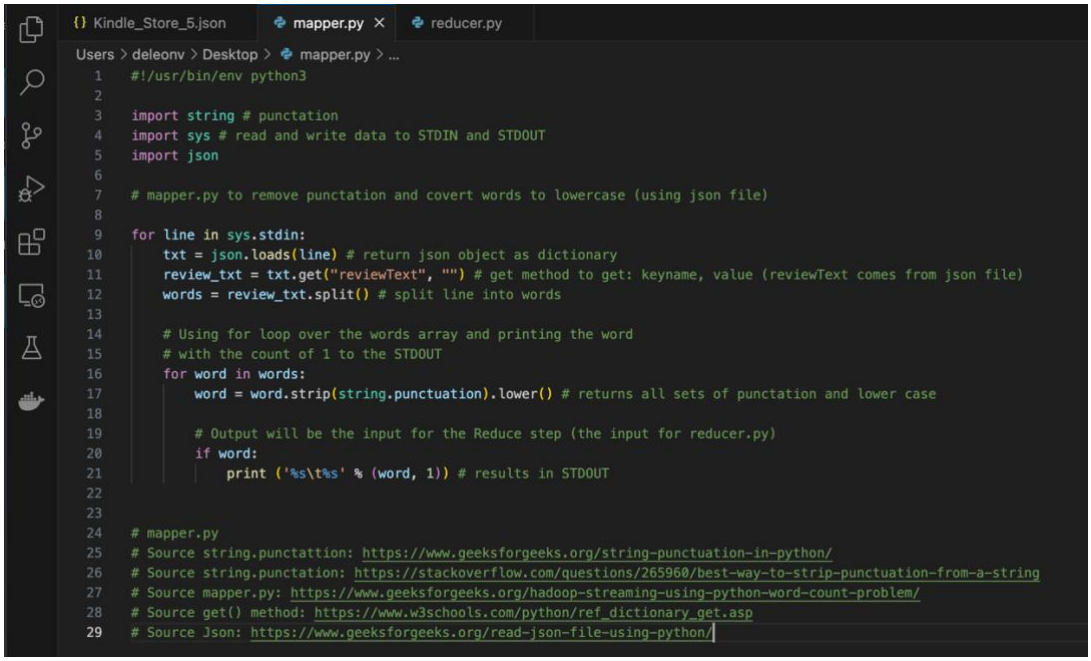
Random line from “Kindle_Store_5.json”:

{'overall': 5.0, 'verified': True, 'reviewTime': '03 3, 2016', 'reviewerID': 'A1E0MODSRYP7O', 'asin': 'B000FA5KK0', 'style': {'Format:': ' Kindle Edition'}, 'reviewerName': 'Jeff', 'reviewText': 'As usual for him, a good book', 'summary': 'a good', 'unixReviewTime': 1456963200}

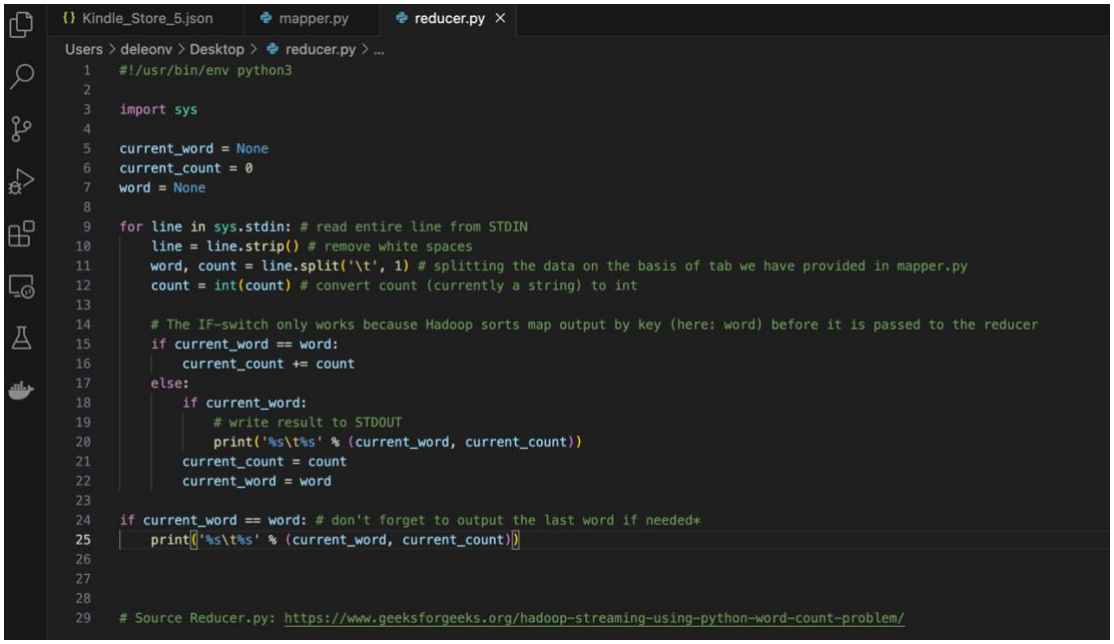
Please Zoom in to see the following image:

Task 11: Word Count

Mapper.py:



Reducer.py:



Observation regarding Mapper.py. The script had a few changes from the original source because the data is in JSON format. I am using the get() method to extract the review text from dictionary txt. This allows me to ensure, even if a review doesn't have a "reviewText" field, that the code won't crash, and it will just use an empty string instead.

The mapper.py and reducer.py came from the these main sources:

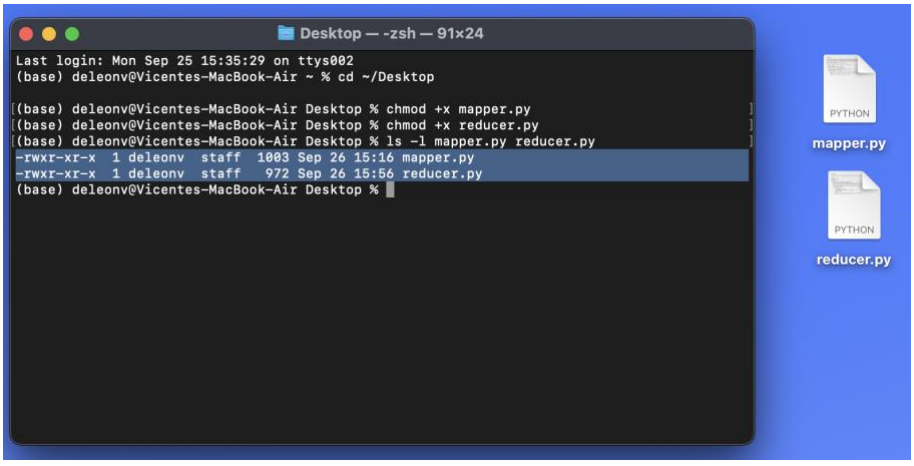
- http://dbversity.com/writing-an-hadoop-mapreduce-program-in-python/
- Mapper.py resources located in “References”.
- Reducer.py resources located in “References”.
- Important: we can’t forget about **“#!/usr/bin/env python3**”. This needs to be within the scripts for them to work with Hadoop.

Mapper.py: the purpose of this python script is to preprocess the “reviewText” from the JSON file input, tokenize it into words, and return word-count as output text. This is a huge file, so it will take a couple of minutes for it to work correctly. For it to completely work, we need to use the reducer.py script to achieve the desire results. As I mentioned before, I made a few changes because the data has a JSON format including a collection of JSON objects. I am using the get() method to extract the review text from dictionary txt. This allows me to ensure, even if a review doesn’t have a “reviewText” field for some reason, the code won’t crash, and it will just use an empty string. At the end of the code, if the resulting word is not an empty string, it prints the word and a count of 1. This Python script will be uploaded so you can view my comments and sources from where this code came from.

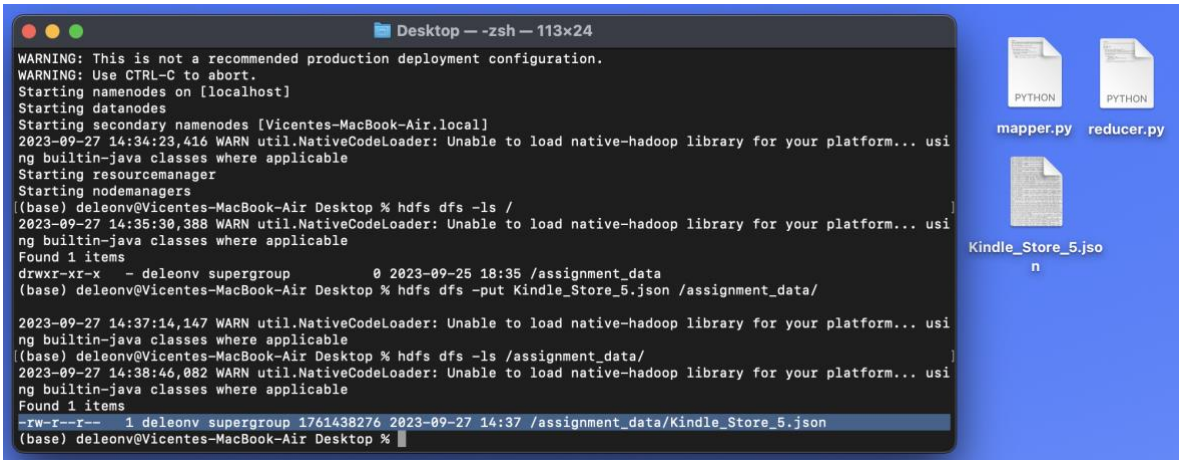
Reducer.py: Now, the purpose of this script is to take the sorted key-value pairs and aggregate the counts for each unique word. It returns the word and its total count as key-value pairs. What is basically does is that for each line it extracts the word and count, it aggregated the count for each unique word, and it returns the word and its total count when there is a new word. This Python script will also be uploaded to final submission for comments and sources purposes.

Both codes are located within my MAC Desktop, and I made the executable and then proceeded to start the Hadoop services. Please see highlighted codes below:

```
1) After writing Python scripts: mapper.py and reducer.py, lets make then available:
type: cd ~/Desktop
type: chmod +x mapper.py (to make the script executable)
type: chmod +x reducer.py (to make the script executable)
type to check if they exist: ls -l mapper.py reducer.py


2) Let's start Hadoop services:
type: start-all.sh (starting Hadoop)
type: hdfs dfs -ls / (let's check if assignment_data directory created for Tasks 1 to 9 still exists. Let's use this smae directory.)
type: hdfs dfs -ls /assignment_data/ (to check if "Kindle_Store_5.json" file is within the HDFS directory)
```

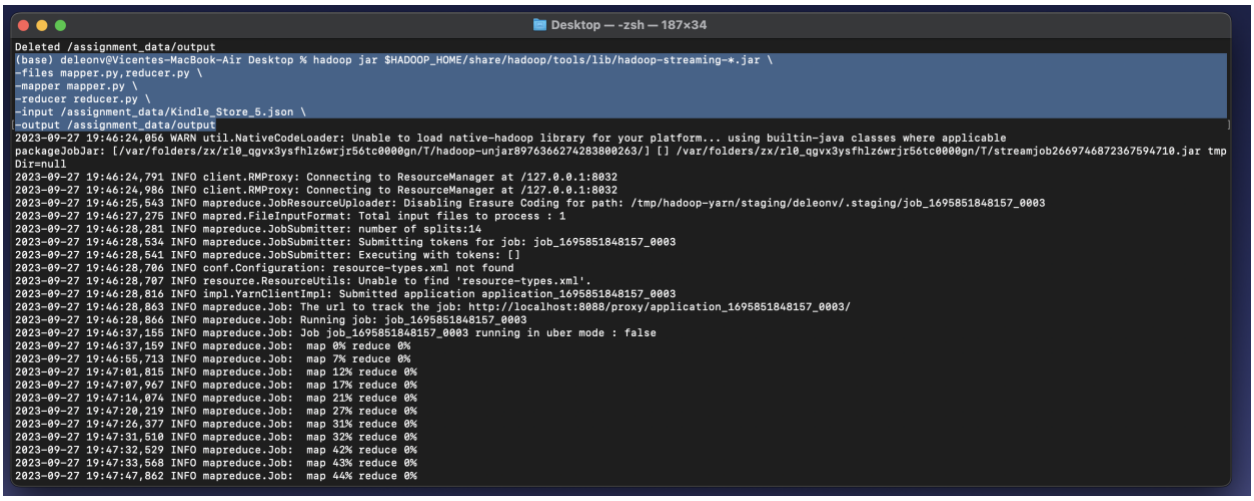


Kindle_Store_5.json is within the HDFS Directory “data_assignment”. Please see highlighted code below:



Now let’s run the MapReduce Job using the Hadoop’s streaming API (Please view References at the end of this document) See highlighted code below to see how the MapReduce job was running within my Mac Terminal.

```
3)
hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming-*.jar \
-files mapper.py,reducer.py \
-mapper mapper.py \
-reducer reducer.py \
-input /assignment_data/Kindle_Store_5.json \
-output /assignment_data/output
```



I was initially having troubles with both Python scripts regarding the JSON implementation and some indentation. Due to this I tried running the MapReduce job and ended up getting an error that said, “System Failed!”. So, I had to delete the “existing failed output” because it seems that Hadoop doesn’t allow overwriting. Just like any other HDFS command, I just -rm command to delete that existing output:

Terminal Code: **hdfs dfs -rm -r /assignment_data/output**

To visualize results (see References at the end of this document) I can use HDFS -cat command. See highlighted code below:

```
Display Results:
Source: https://stackoverflow.com/questions/45316617/how-can-i-view-a-mapreduce-job-hadoop-output-file
Source: https://community.cloudera.com/t5/Support-Questions/How-can-we-see-the-output-in-single-file-if-3-files-are/m-p/114110
Source: https://courses.engr.illinois.edu/cs398acc/sp2018/mps/mp2.html

let's display entire content of output directory:
type: hdfs dfs -ls /assignment_data/output

Just like Part 1, I can use the cat command to display content
type: hdfs dfs -cat /assignment_data/output/part-00000

If I want to save results I can use the get command:
type: hdfs dfs -get /assignment_data/output/part-00000 ~/Desktop/Task11_output.txt
```

```
(base) deleonv@Vicentes-MacBook-Air ~ % hdfs dfs -ls /assignment_data/output
2023-09-29 13:21:30,074 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r--   1 deleonv supergroup          0 2023-09-27 19:51 /assignment_data/output/_SUCCESS
-rw-r--r--   1 deleonv supergroup   12171842 2023-09-27 19:51 /assignment_data/output/part-00000
(base) deleonv@Vicentes-MacBook-Air ~ %
```

```
Desktop — -zsh — 187×34
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -cat /assignment_data/output/part-00000
```

Results:

```
Desktop — -zsh — 187×34
hecka   6
heckava 1
heckel  1
heckenbach      3
heckenbach's    2
heckes  2
hecki   2
heckitty        1
heckl   1
heckle  27
hecklebee       1
heckled 6
heckleena       1
heckler 10
hecklers        3
heckles 6
heckling        6
heckman 13
heckman's       2
heckmans        1
heckno  1
heckof  1
heckoh..ahhh    1
hecks   9
heckthese       1
heckthis        1
hecktor 2
heckuva 37
heckuvalot      1
hecky   3
heckyeah        1
heckyou 1
hecotor's       1
hecould 2
```

I decided to find a way on how to save the text output so I could upload in the final submission as evidence. As you can see after running the last code below on the terminal, you can see a new file located on my Desktop named: Task11_output.txt. See highlighted code below:

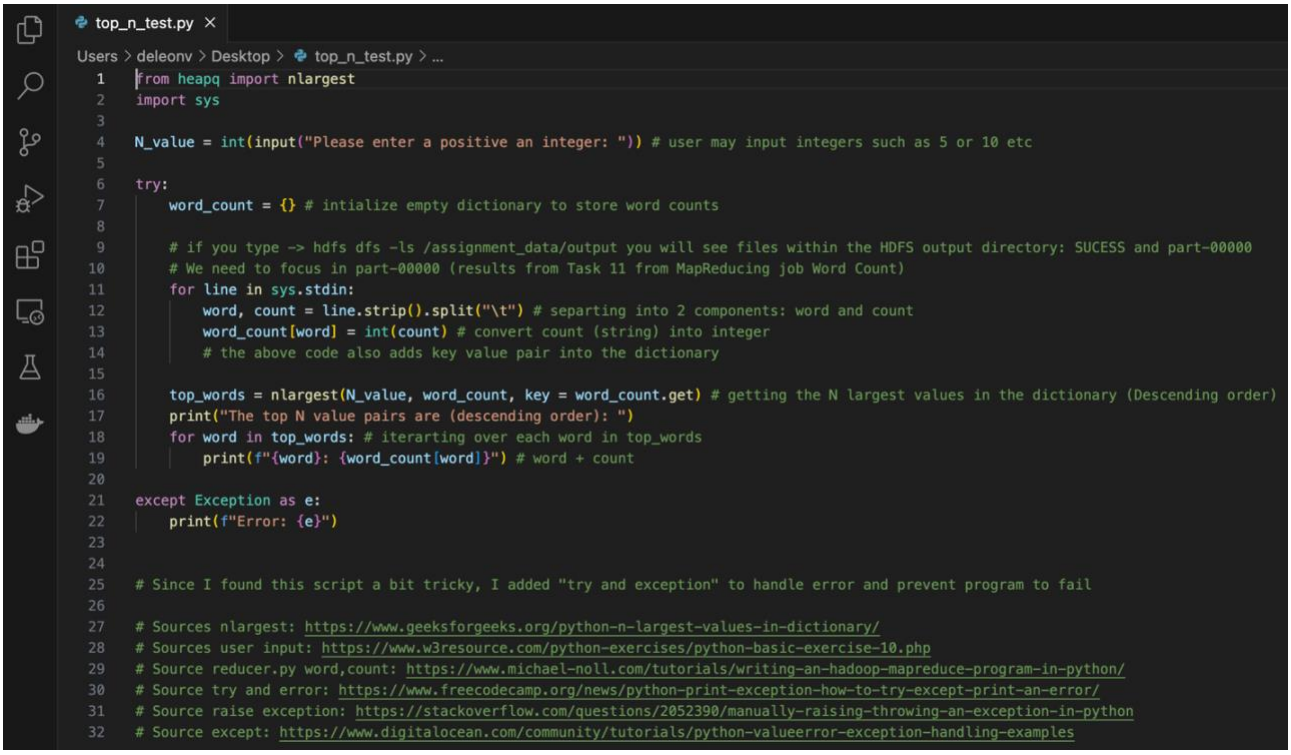
```
(base) deleonv@Vicentes-MacBook-Air Desktop % hdfs dfs -get /assignment_data/output/part-00000 ~/Desktop/Task11_output.txt

2023-09-27 20:19:42,839 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
(base) deleonv@Vicentes-MacBook-Air Desktop %
```

It seems that my MapReduce programs worked. Even though it returned a huge text file, you can still see that it returned what was asked: **lower case words, tokenized words with no punctation, and finally word-count pairs.**

Task 12: Top N Words

Let's begin this task by creating the top_n_words.py. Before having the final script ready to run for Task 12, I had to make many multiple changes to the script for it to successfully run. One of things or features I tried to implement was the use of "user input". I was having so much trouble trying to pit out together. But then, I understood and remember that it seems that Hadoop streaming expects scripts to read from STDIN and not to wait for some user input. The following image is an early script test I had in mind to allow the user to specify the value of N.

Test file: top_n_test.py



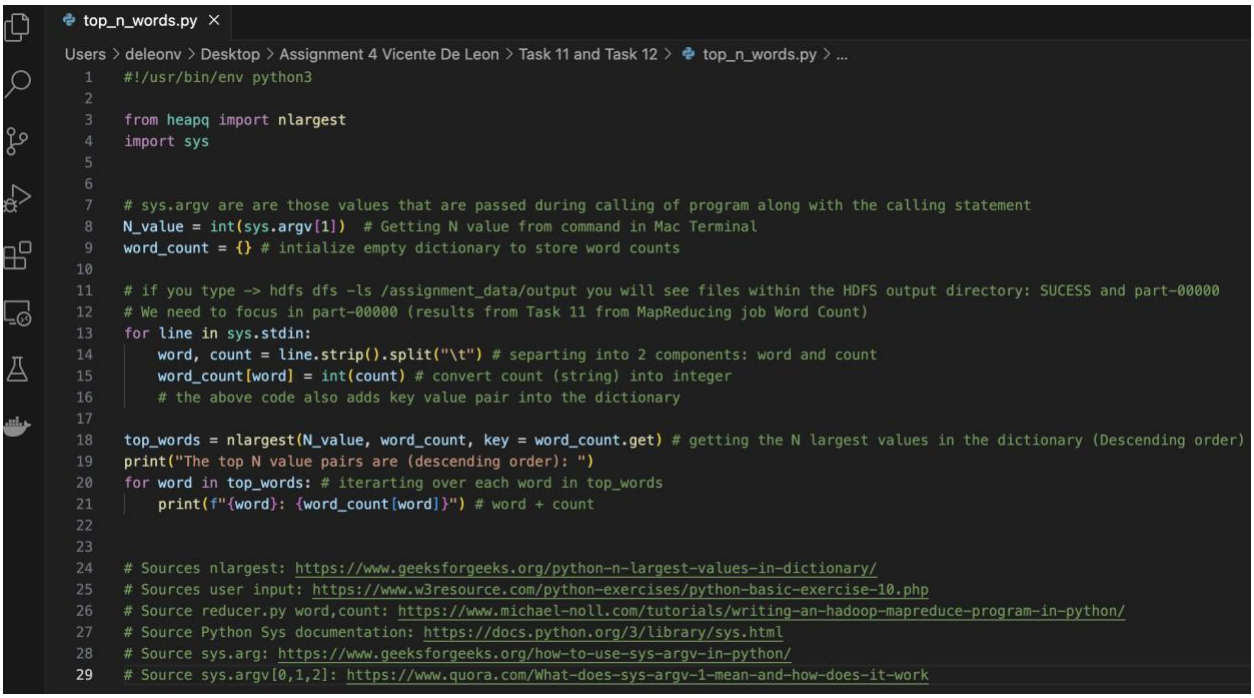
As you can see, I had sources and many features for error handling within the code. However, this code failed multiple times.

The following code is the code I put together from multiple sources to successfully run the MapReduce Job using Python. As you can see, the N value variable uses sys.argv[] to get the N value from command in the terminal. Sys.argv[0} -> will take on the N value the user specifies. In my case, I tried 2 different values: 5 and 10 to get both the top 5 words and the top 10 words.

```
important, we can view this in source:
sys.argv[0,1,2]: https://www.quora.com/What-does-sys-argv-1-mean-and-how-does-it-work
sys.argv[0] -> top_n_words.py
sys.argv[1] -> 5 (or any other number that the user likes top 5, top 10 etc)
```

Final top_n_words.py:

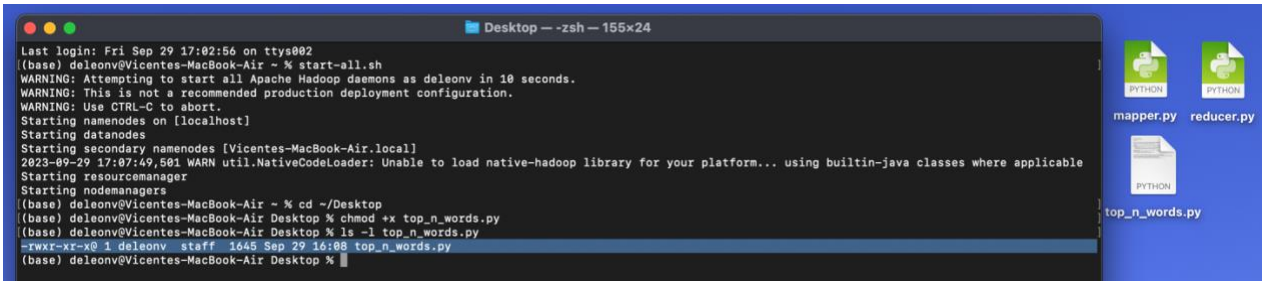
It is important to never forget about the -> #!/usr/bin/env python3 (we needed for it run).



We initialize an empty dictionary named -> word_count, then the script focuses on what's inside the HDFS Task 11 output directory. If we use the -ls command to view files within output directory, we will find 2 files: SUCCESS and part-00000. We want to focus on "part-00000" (results from the Word Count program) while the SUCCESS file is empty file we don't really need. We proceed to do a for loop (each line) and separate them into 2 components: word and count. Then we proceed to convert the string into integer and add the key value pair into the dictionary we initially created. Finally, we use "nlargest" (Geeks for Geeks source) to get the N largest values in the dictionary in descending order. We iterate over each word and prints the word + the count. So, to answer one of the Task 12 questions, I didn't want to modify any script or include any new code into existing code. I just went ahead and created a new script for this Task. Everything will be submitted within the final submission.

So, we make this script executable by:

```
58  Task 12)
59  1) Write new top_n_words.py script and make it executable:
60  type: cd ~/Desktop
61  type: chmod +x top_n_words.py (to make the script executable)
62  type: ls -l top_n_words.py (to check if it exists)
```



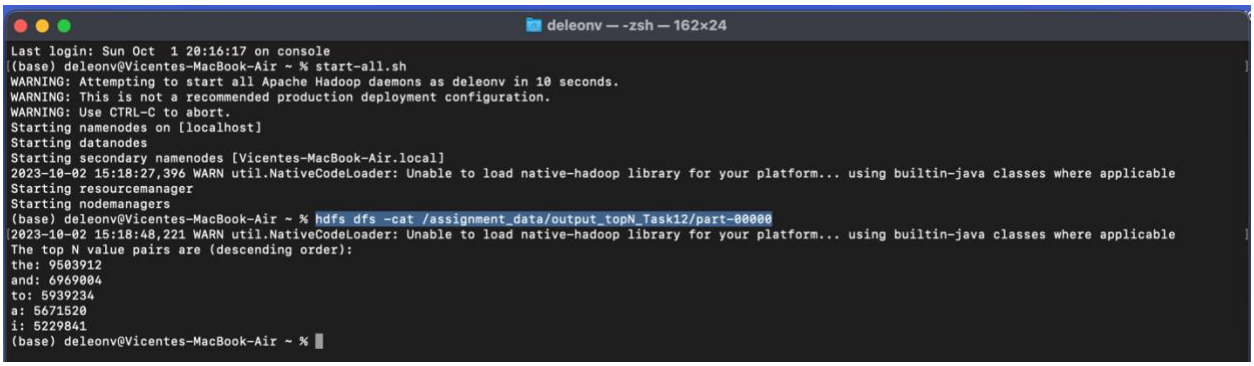
We need to run this command just like I did Task 11. I am using /bin/cat as the mapper because it reads the files and outputs lines. Why? I just want to get the word counts as they are and let the reducer handle the top n procedure. It is extremely important to specify N value right next to top_n_words.py (In this case, I used 5 to get the top 5 words) for it to return the n value you specified. As a I mentioned before, the input will be part-00000 (results task 11) and the output will be named “output_topN_Task12”.

output_topN_Task12 -> contains Top 5 N words.

```
67  2) To run Task 12 to run hadoop job (part-00000 is the result from Task 11):
68  hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming-*.jar \
69  -files top_n_words.py \
70  -mapper "/bin/cat" \
71  -reducer "top_n_words.py 5" \
72  -input /assignment_data/output/part-00000 \
73  -output /assignment_data/output_topN_Task12
74
75  view output directory:
76  type: hdfs dfs -ls /assignment_data/output_topN_Task12 (top 5 words)
```

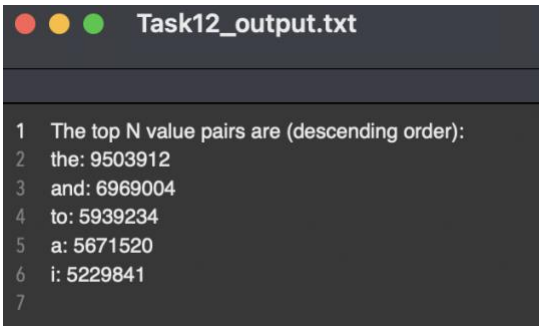
Visualize results (Top 5N Descending Order):

type: hdfs dfs -cat /assignment_data/output_topN_Task12/part-00000



Saving results Top 5N (Task12_output.txt results will be submitted in final submission):

```
89
90  If I want to save results I can use the get command:
91  type: hdfs dfs -get /assignment_data/output_topN_Task12/part-00000 ~/Desktop/Task12_output.txt
```



Let's try again, but this time by doing the job with Top 10N. As you can see in the image below, I specified N value to 10 instead of 5 (as seen highlighted green). Just like explained above, I proceed to run this command in my Mac Terminal to get the desired results. This time output name will be "output_topN10_Task12".

output_topN10_Task12 -> contains Top 10 N words.

```
95  To run Task 12 to run hadoop job (part-00000 is the result from Task 11):
96  hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming-*.jar \
97  -files top_n_words.py \
98  -mapper "/bin/cat" \
99  -reducer "top_n_words.py 10" \
100 -input /assignment_data/output/part-00000 \
101 -output /assignment_data/output_topN10_Task12
```

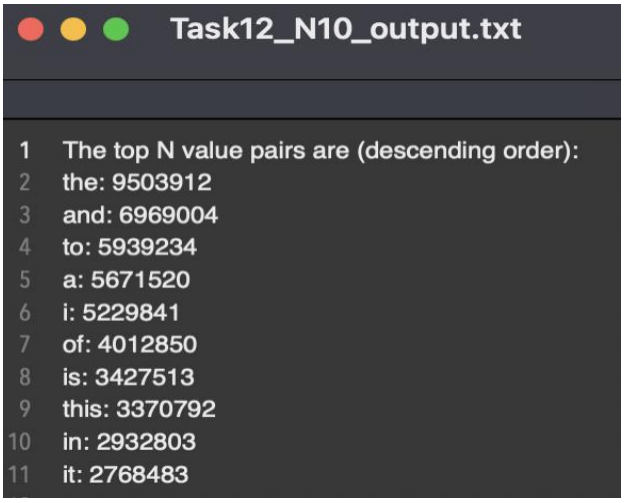
Visualize results (Top 10N Descending Order):

type: hdfs dfs -cat /assignment_data/output_topN10_Task12/part-00000

```
(base) deleonv@Vicentes-MacBook-Air ~ % hdfs dfs -cat /assignment_data/output_topN10_Task12/part-00000
2023-10-02 15:25:43,457 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
The top N value pairs are (descending order):
the: 9503912
and: 6969004
to: 5939234
a: 5671520
i: 5229841
of: 4012850
is: 3427513
this: 3370792
in: 2932803
it: 2768483
```

Saving results Top 10N (Task12_N10_output.txt results will also be submitted in final submission):

```
Results:
type: hdfs dfs -cat /assignment_data/output_topN10_Task12/part-00000

If I want to save results I can use the get command:
type: hdfs dfs -get /assignment_data/output_topN10_Task12/part-00000 ~/Desktop/Task12_N10_output.txt
```



Combiner Section:

In this section, we needed to repeat section 1 while using "combiner" in reducer and compare the results with the results in section 1. What's basically going on here is that the combiner takes on the mapper's output and performs aggregation on it. this reduces the amount of data that needs to be shuffled and sent to reducers. The reducer is just adding up counts. Due to this it can be used as combiner as well. We are taking the original JSON data: Kindle_Store_5.json as our input and the output will be named "output_with_combiner_Task12". We are repeating the same step we initially did in Task 11 while using combiner to compare results later on.

output_with_combiner_Task12 -> contains results just like Task 11 output.

```
Combiner comparison:
hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming-*.jar \
-files mapper.py,reducer.py \
-mapper mapper.py \
-combiner reducer.py \
-reducer reducer.py \
-input /assignment_data/Kindle_Store_5.json \
-output /assignment_data/output_with_combiner_Task12
```

Commands to visualize and save combiner results (will be in final submission):

Task12_output_combiner.txt -> stored results in text file format.

```
Results:
type: hdfs dfs -cat /assignment_data/output_with_combiner_Task12/part-00000

If I want to save results I can use the get command:
type: hdfs dfs -get /assignment_data/output_with_combiner_Task12/part-00000 ~/Desktop/Task12_output_combiner.txt
```

We proceed to the run following command on terminal and as you can see I specified N value 5 to get combiner top 5 words (highlighted in green).

output_topN_Task12_combiner -> contains Top 5 N Combiner words. This file will also be in final submission.

```
COMBINER 5N ########################
hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming-*.jar \
-files top_n_words.py \
-mapper "/bin/cat" \
-reducer "top_n_words.py 5" \
-input /assignment_data/output_with_combiner_Task12/part-00000 \
-output /assignment_data/output_topN_Task12_combiner

view output directory:
type: hdfs dfs -ls /assignment_data/output_topN_Task12_combiner
```

I use the HDFS cat command to visualize results:

```

Results:
type: hdfs dfs -cat /assignment_data/output_topN_Task12_combiner/part-00000
```

```
(base) deleonv@Vicentes-MacBook-Air ~ % hdfs dfs -cat /assignment_data/output_topN_Task12_combiner/part-00000
2023-10-02 15:54:41,963 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
The top N value pairs are (descending order):
the: 9503912
and: 6969004
to: 5939234
a: 5671520
i: 5229841
```

To store Combiner top 5N words:

Task12_topN_output.txt -> contains top 5N combiner words (this will be in final submission).

```

If I want to save results I can use the get command:
type: hdfs dfs -get /assignment_data/output_topN_Task12_combiner/part-00000 ~/Desktop/Task12_topN_output.txt
```

Task12_topN_output.txt

```
1  The top N value pairs are (descending order):
2  the: 9503912
3  and: 6969004
4  to: 5939234
5  a: 5671520
6  i: 5229841
7
```

Comparison Top 5N:

Task12_output.txt (No Combiner) vs Task12_topN_output.txt (Combiner) No change at all.

Task12_output.txt

```
1 The top N value pairs are (descending order):
2 the: 9503912
3 and: 6969004
4 to: 5939234
5 a: 5671520
6 i: 5229841
```

Task12_topN_output.txt

```
1 The top N value pairs are (descending order):
2 the: 9503912
3 and: 6969004
4 to: 5939234
5 a: 5671520
6 i: 5229841
```

We do the same process all over again using combiner data (N value 10):

```
160 COMBINER 10N ##############################
161 hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming-*.jar \
162 -files top_n_words.py \
163 -mapper "/bin/cat" \
164 -reducer "top_n_words.py 10" \
165 -input /assignment_data/output_with_combiner_Task12/part-00000 \
166 -output /assignment_data/output_topN10_Task12_combiner
```

```
176 Results:
177 type: hdfs dfs -cat /assignment_data/output_topN10_Task12_combiner/part-00000
```

```
(base) deleonv@Vicentes-MacBook-Air ~ % hdfs dfs -cat /assignment_data/output_topN10_Task12_combiner/part-00000
2023-10-02 16:04:08,909 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
The top N value pairs are (descending order):
the: 9503912
and: 6969004
to: 5939234
a: 5671520
i: 5229841
of: 4012850
is: 3427513
this: 3370792
in: 2932803
it: 2768483
```

To store Top 10N text file:

Task12_topN10_output.txt -> contains top 10N combiner words (this will also be in final submission).

```
179  If I want to save results I can use the get command:
180  type: hdfs dfs -get /assignment_data/output_topN10_Task12_combiner/part-00000 ~/Desktop/Task12_topN10_output.txt
```

Task12_topN10_output.txt

```
1  The top N value pairs are (descending order):
2  the: 9503912
3  and: 6969004
4  to: 5939234
5  a: 5671520
6  i: 5229841
7  of: 4012850
8  is: 3427513
9  this: 3370792
10 in: 2932803
11 it: 2768483
```

Comparison Top 10N:

Task12_N10_output.txt (No Combiner) vs Task12_topN10_output.txt (Combiner) No change at all.

Task12_N10_output.txt

```
1  The top N value pairs are (descending order):
2  the: 9503912
3  and: 6969004
4  to: 5939234
5  a: 5671520
6  i: 5229841
7  of: 4012850
8  is: 3427513
9  this: 3370792
10 in: 2932803
11 it: 2768483
```

Task12_topN10_output.txt

```
1  The top N value pairs are (descending order):
2  the: 9503912
3  and: 6969004
4  to: 5939234
5  a: 5671520
6  i: 5229841
7  of: 4012850
8  is: 3427513
9  this: 3370792
10 in: 2932803
11 it: 2768483
```

We end Hadoop Services:

```
185
186  END: stop-all.sh
187
```

References:

Mapper.py:

- string.punctattion: https://www.geeksforgeeks.org/string-punctuation-in-python/
- string.punctation: https://stackoverflow.com/questions/265960/best-way-to-strip-punctuation-from-a-string
- mapper.py: https://www.geeksforgeeks.org/hadoop-streaming-using-python-word-count-problem/
- get() method: https://www.w3schools.com/python/ref_dictionary_get.asp
- Json: https://www.geeksforgeeks.org/read-json-file-using-python/

Reducer.py:

- Reducer.py: https://www.geeksforgeeks.org/hadoop-streaming-using-python-word-count-problem/

Task 11:

- Source: https://www.michael-noll.com/tutorials/writing-an-hadoop-mapreduce-program-in-python/
- Source: https://cseweb.ucsd.edu/~jmcauley/datasets/amazon_v2/
- Source: https://www.tutorialspoint.com/hadoop/hadoop_enviornment_setup.htm
- Source: http://dbversity.com/writing-an-hadoop-mapreduce-program-in-python/
- Source: https://hadoop.apache.org/docs/r1.2.1/streaming.html
- Source: https://hadoop.apache.org/docs/stable/hadoop-streaming/HadoopStreaming.html
- Source: https://courses.engr.illinois.edu/cs398acc/sp2018/mps/mp2.html
- Source: https://stackoverflow.com/questions/45316617/how-can-i-view-a-mapreduce-job-hadoop-output-file
- Source: https://community.cloudera.com/t5/Support-Questions/How-can-we-see-the-output-in-single-file-if-3-files-are/m-p/114110
- Source: https://courses.engr.illinois.edu/cs398acc/sp2018/mps/mp2.html
- Source: https://data-flair.training/blogs/hadoop-combiner-tutorial/

Task 12 (including top_n_words.py):

- Source: https://www.quora.com/What-does-sys-argv-1-mean-and-how-does-it-work
- Source: https://www.geeksforgeeks.org/python-n-largest-values-in-dictionary/
- Source: https://www.w3resource.com/python-exercises/python-basic-exercise-10.php
- Source: https://www.michael-noll.com/tutorials/writing-an-hadoop-mapreduce-program-in-python/
- Source: https://docs.python.org/3/library/sys.html
- Source: https://www.geeksforgeeks.org/how-to-use-sys-argv-in-python/