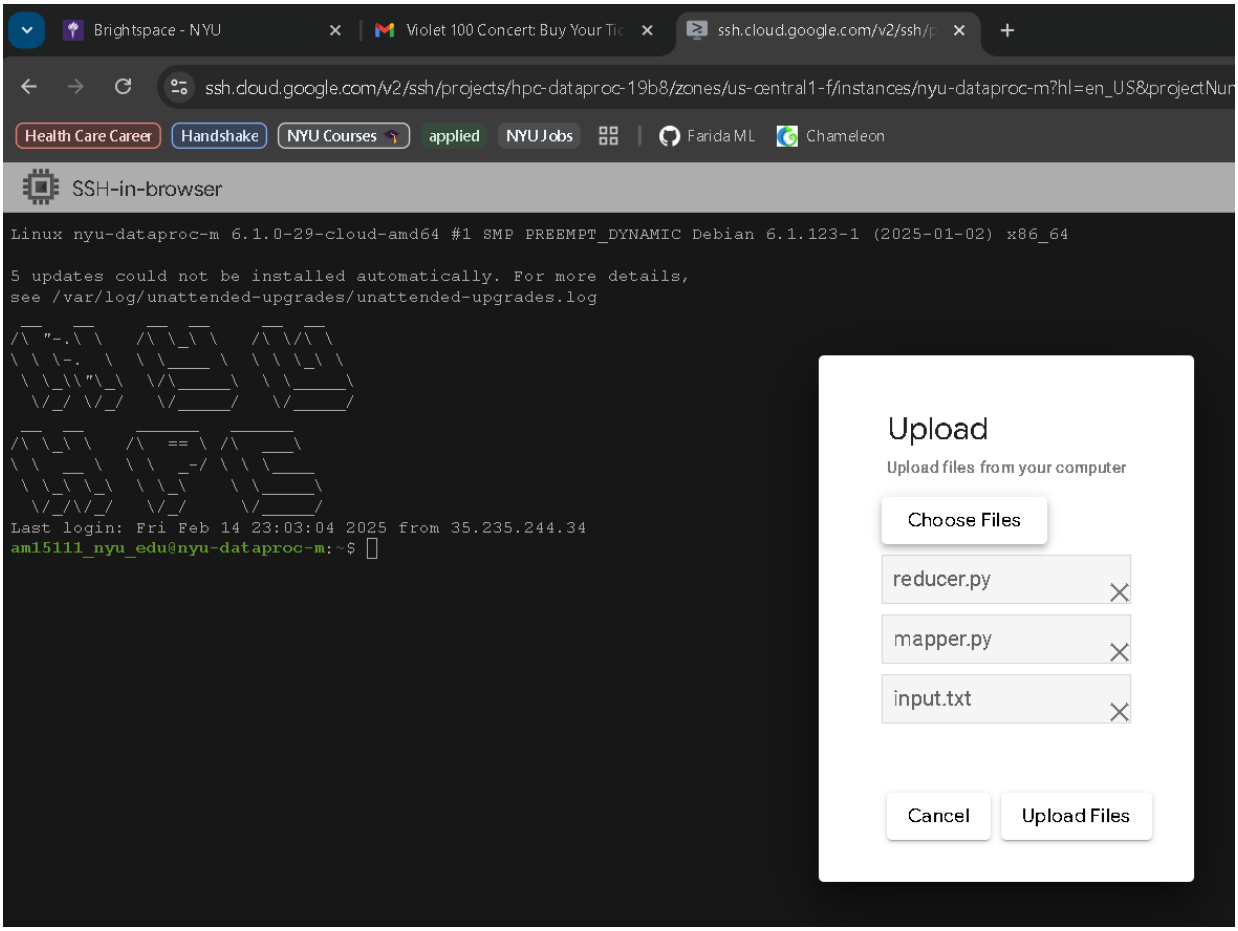
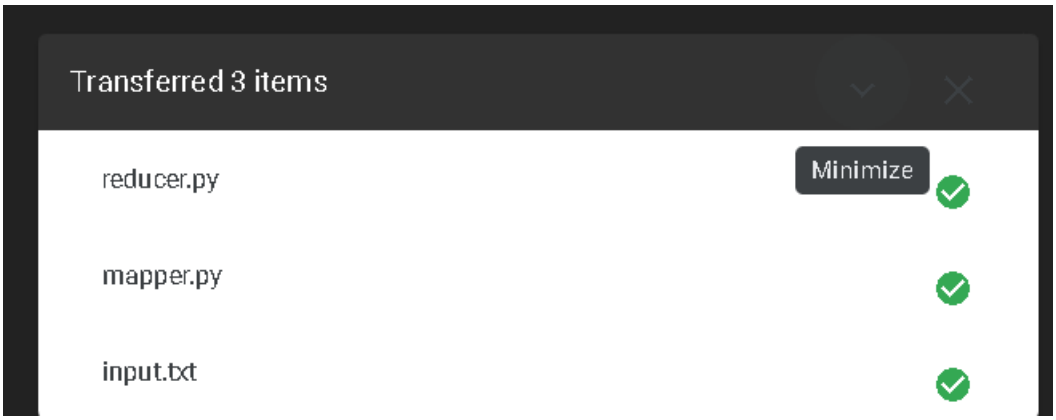
# GHW#1- Big Data Assignment

## Akshat Mishra | netid: am15111 | Spring 2025

**Step 1:** Upload the files to the NYU Dataproc local disk.



**Step 2:** Use “ls” command to check the presence of files in the directory.

```
am15111_nyu_edu@nyu-dataproc-m:~$ ls
input.txt  mapper.py  reducer.py
```

**Step 3:** Move the input to HDFS using the following command:

**hdfs dfs -put input.txt**

Perform “ls” on the HDFS using the following command and we should be getting the input.txt that we just moved.

```
am15111_nyu_edu@nyu-dataproc-m:~$ hdfs dfs -ls
Found 1 items
-rw-r--r--   1 am15111_nyu_edu am15111_nyu_edu    2615571 2025-02-14 23:22 input.txt
```

**Step 4:** Inside Dataproc, all env variables like $JAVA_HOME and $HADOOP_HOME are already set for us. Listing the contents of the $HADOOP_HOME directory, we get

```
am15111_nyu_edu@nyu-dataproc-m:~$ ls $HADOOP_HOME
bin                                 hadoop-azure-datalake.jar           hadoop-extras-3.3.6.jar                                  hadoop-registry.jar
client                              hadoop-azure.jar                    hadoop-extras.jar                                        hadoop-resourceestimator-3.3.6.jar
etc                                 hadoop-client-3.3.6.jar             hadoop-fs2img-3.3.6.jar                                  hadoop-resourceestimator.jar
hadoop-aliyun-3.3.6.jar             hadoop-client.jar                   hadoop-fs2img.jar                                        hadoop-rumen-3.3.6.jar
hadoop-aliyun.jar                   hadoop-common-3.3.6-tests.jar       hadoop-google-secret-manager-credential-provider-3.3.6.jar hadoop-rumen.jar
hadoop-annotations-3.3.6.jar        hadoop-common-3.3.6.jar             hadoop-google-secret-manager-credential-provider.jar     hadoop-shaded-guava.jar
hadoop-annotations.jar              hadoop-common.jar                   hadoop-gridmix-3.3.6.jar                                 hadoop-shaded-protobuf.jar
hadoop-archive-logs-3.3.6.jar       hadoop-datajoin-3.3.6.jar           hadoop-gridmix.jar                                       hadoop-sls-3.3.6.jar
hadoop-archive-logs.jar             hadoop-datajoin.jar                 hadoop-kafka-3.3.6.jar                                   hadoop-sls.jar
hadoop-archives-3.3.6.jar           hadoop-distcp-3.3.6.jar             hadoop-kafka.jar                                         hadoop-streaming-3.3.6.jar
hadoop-archives.jar                 hadoop-distcp.jar                   hadoop-kms-3.3.6.jar                                     hadoop-streaming.jar
hadoop-auth-3.3.6.jar               hadoop-dynamometer-blockgen-3.3.6.jar hadoop-kms.jar                                         lib
hadoop-auth.jar                     hadoop-dynamometer-blockgen.jar     hadoop-minicluster-3.3.6.jar                             libexec
hadoop-aws-3.3.6.jar                hadoop-dynamometer-infra-3.3.6.jar  hadoop-minicluster.jar                                   sbin
hadoop-aws.jar                      hadoop-dynamometer-infra.jar        hadoop-nfs-3.3.6.jar
hadoop-azure-3.3.6.jar              hadoop-dynamometer-workload-3.3.6.jar hadoop-nfs.jar
hadoop-azure-datalake-3.3.6.jar     hadoop-dynamometer-workload.jar     hadoop-registry-3.3.6.jar
```

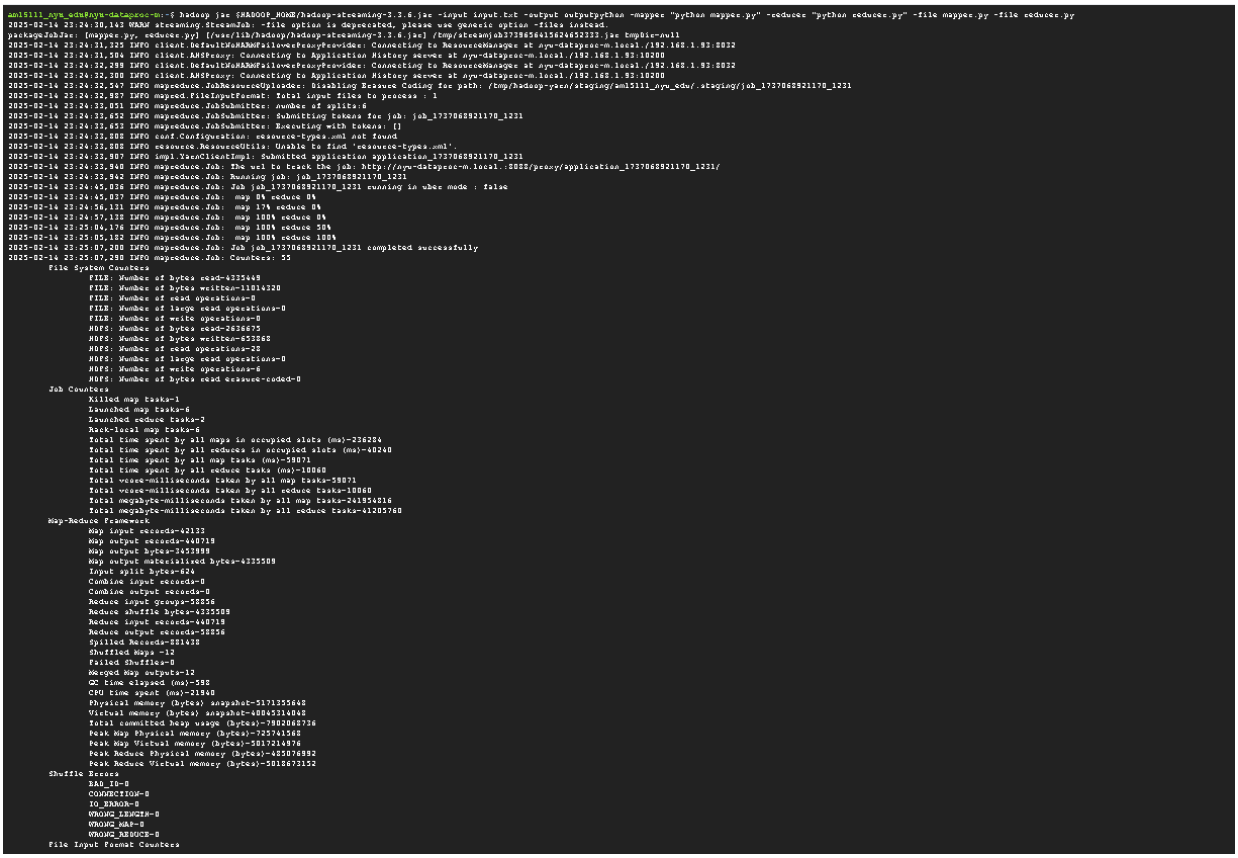
**Step 5:** We will use the hadoop-streaming-3.2.2.jar file present inside this directory to execute a python based MapReduce as show with the below command:

**hadoop jar $HADOOP_HOME/hadoop-streaming-3.3.6.jar -input input.txt -output outputpython -mapper "python mapper.py" -reducer "python reducer.py" -file mapper.py -file reducer.py**

```
am15111_nyu_edu@nyu-dataproc-m:~$ hadoop jar $HADOOP_HOME/hadoop-streaming-3.3.6.jar -input input.txt -output outputpython -mapper "python mapper.py" -reducer "python reducer.py" -file mapp
er.py -file reducer.py
2025-02-14 23:24:30,143 WARN streaming.StreamJob: -file option is deprecated, please use generic option -files instead.
packageJobJar: [mapper.py, reducer.py] [/usr/lib/hadoop/hadoop-streaming-3.3.6.jar] /tmp/streamjob3739656415624652333.jar tmpDir=null
2025-02-14 23:24:31,325 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at nyu-dataproc-m.local./192.168.1.93:8032
2025-02-14 23:24:31,504 INFO client.AHSProxy: Connecting to Application History server at nyu-dataproc-m.local./192.168.1.93:10200
2025-02-14 23:24:32,299 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at nyu-dataproc-m.local./192.168.1.93:8032
2025-02-14 23:24:32,300 INFO client.AHSProxy: Connecting to Application History server at nyu-dataproc-m.local./192.168.1.93:10200
2025-02-14 23:24:32,547 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/am15111_nyu_edu/.staging/job_1737068921170_1231
2025-02-14 23:24:32,987 INFO mapred.FileInputFormat: Total input files to process : 1
2025-02-14 23:24:33,051 INFO mapreduce.JobSubmitter: number of splits:6
2025-02-14 23:24:33,652 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1737068921170_1231
2025-02-14 23:24:33,653 INFO mapreduce.JobSubmitter: Executing with tokens: []
2025-02-14 23:24:33,808 INFO conf.Configuration: resource-types.xml not found
2025-02-14 23:24:33,808 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2025-02-14 23:24:33,907 INFO impl.YarnClientImpl: Submitted application application_1737068921170_1231
2025-02-14 23:24:33,940 INFO mapreduce.Job: The url to track the job: http://nyu-dataproc-m.local.:8088/proxy/application_1737068921170_1231/
2025-02-14 23:24:33,942 INFO mapreduce.Job: Running job: job_1737068921170_1231
2025-02-14 23:24:45,036 INFO mapreduce.Job: Job job_1737068921170_1231 running in uber mode : false
2025-02-14 23:24:45,037 INFO mapreduce.Job:  map 0% reduce 0%
2025-02-14 23:24:56,131 INFO mapreduce.Job:  map 17% reduce 0%
2025-02-14 23:24:57,138 INFO mapreduce.Job:  map 100% reduce 0%
```

**Step 6:** Since the mentioned folder does not exit, This should trigger a job like this:

After successful completion of the job, you should see a 'Job completed successfully' message:




**Step 7:** Listing the contents of the output directory and moving output files from HDFS to Dataproc using:

**hdfs dfs -get outputpython**

```
am15111_nyu_edu@nyu-dataproc-m:~$ hdfs dfs -ls outputpython
Found 3 items
-rw-r--r--   1 am15111_nyu_edu am15111_nyu_edu          0 2025-02-14 22:33 outputpython/_SUCCESS
-rw-r--r--   1 am15111_nyu_edu am15111_nyu_edu     327750 2025-02-14 22:33 outputpython/part-00000
-rw-r--r--   1 am15111_nyu_edu am15111_nyu_edu     326118 2025-02-14 22:33 outputpython/part-00001
```

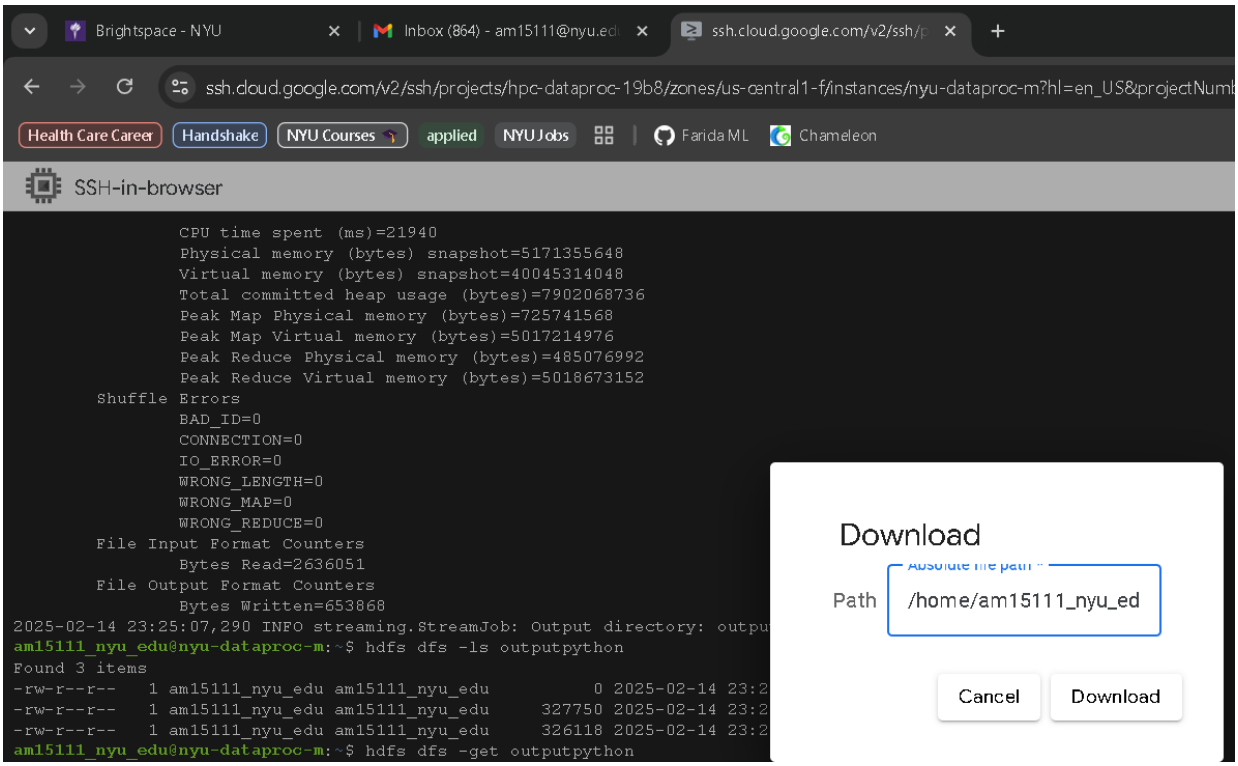
We will see all the files in dataproc local now using "ls outputpython"

```
am15111_nyu_edu@nyu-dataproc-m:~$ ls outputpython
_SUCCESS  part-00000  part-00001
```

**Step 8:** For consolidating all the outputs within a single sorted result file (without using a single reducer), we could use a cat command with a wildcard:

**hdfs dfs -cat outputpython/part* | sort > concatenated_results_new.tx**

**cp concatenated_file.txt .**



**Output file of the concatenated file**

```
am15111_nyu_edu@nyu-dataproc-m:~$ cat concatenated_file.txt
!"ren<:h        1
!'('{)1I('.\.I. 1
!1('Xl  1
!7llllll)I'll('.l.)'    1
!<'Nl]7£'t',    1
!IIfI,'Ct¢',(l  1
!IamI——'1wi1I   1
!Iu'_/1'tfuI.u,m.       1
!WL'7t/'V-(wt)  1
!hc'r0j'ore     1
!ilce_/isft,    1
!l'Ii.r 1
!lIlIlL'C('S.i'aIy      1
!lm'funereal    1
!t'G.)'t)fl     1
!'H$€   1
!'aIa7.:'.n     1
"       159
"'      1
"'1     1
"':vl_v 1
"(.'Izildmrt,.  1
"(x'r.su:zl     1
".      1
".w     1
"/ilrigfzr,     1
"0      1
"0f"mplir,'dIl1zr       1
"1      1
":1     1
":4     1
"A      1
"Abdul, 1
"Ali,   1
```

The Downloaded file is concatenated_file.txt.

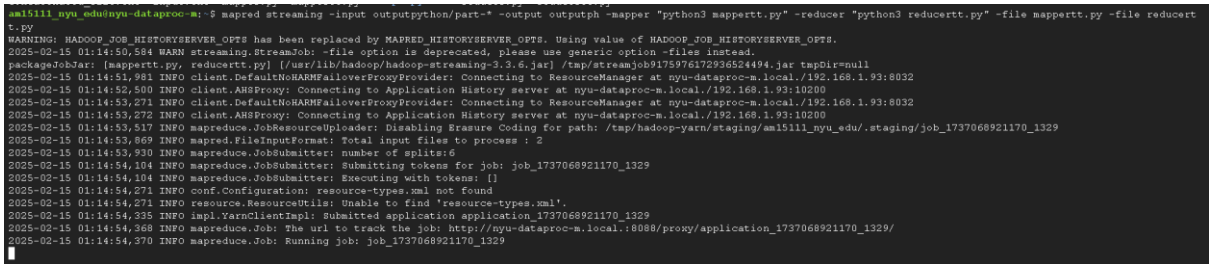
**b. Modify the above code to print the top 10 maximum occurring words along with their frequencies for your text file.**

**Step1:** Repeating what we did in (a) Question.

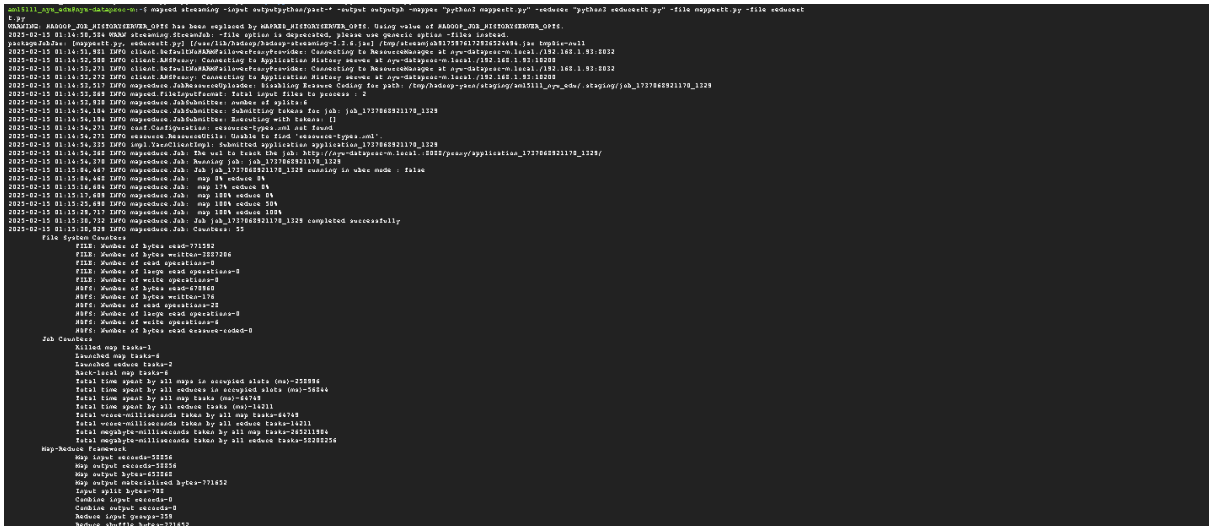
**Step2:** I have created mappertt.py and reducertt.py implements min heap to keep track of top10 occurring words.

**Step3:** We will run command:

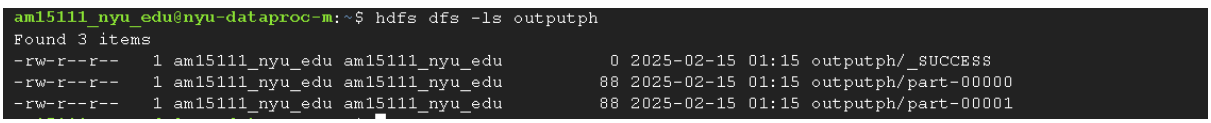
**mapred streaming -input outputpython/part-* -output outputph -mapper "python3 mappertt.py" -reducer "python3 reducertt.py" -file mappertt.py -file reducertt.py**




Complete output of the command.

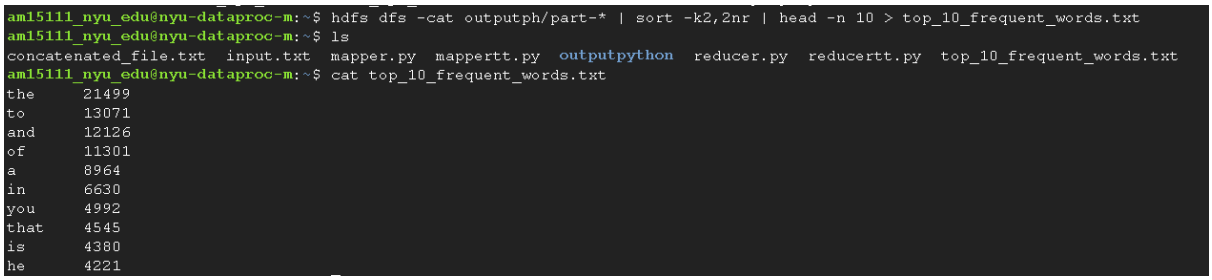


**Step4:** Since there are two files as seen using command: **hdfs dfs -ls outputph** .

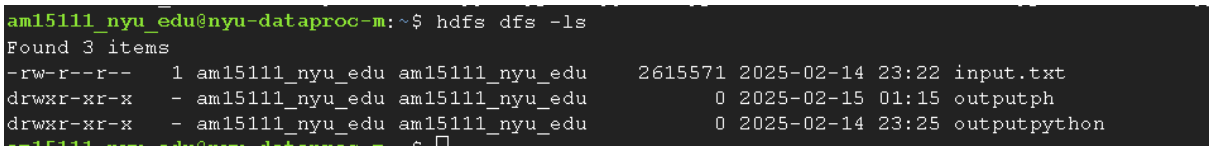


Need to concatenate these two files and take top 10 from both of these files, so running this command gives sorted based on maximum occurring frequency and saving it in file top_10_frequent_words.txt :
**hdfs dfs -cat outputph/part-* | sort -k2,2nr | head -n 10 > top_10_frequent_words.txt**

With the use of : **cat top_10_frequent_words.txt** we can view the file contents.




**Step 5:** Download the top_10_frequent _words.txt and Now directory structure looks like this:




The Downloaded file is present in the Folder. (top_10_frequent_words.txt)

**C. Create a new text file which is somewhere between 12-13mb, but in order to achieve this, copy and paste a small portion of the previously created text file (like around the first 250-300 lines, not more) multiple times till you reach the desired size. Perform the same programs (a) and (b) on this file.**

**Step1:** Uploading the text file named “duplicated_input.txt”. Directory structure after uploading this file:

```
am15111_nyu_edu@nyu-dataproc-m:~$ ls
concatenated_file.txt  duplicated_input.txt  input.txt  mapper.py  mappertt.py  outputpython  reducer.py  reducertt.py  top_10_frequent_words.txt
am15111_nyu_edu@nyu-dataproc-m:~$ hdfs dfs -put duplicated_input.txt
```

**Step 2:** uploading this file to hdfs, command:

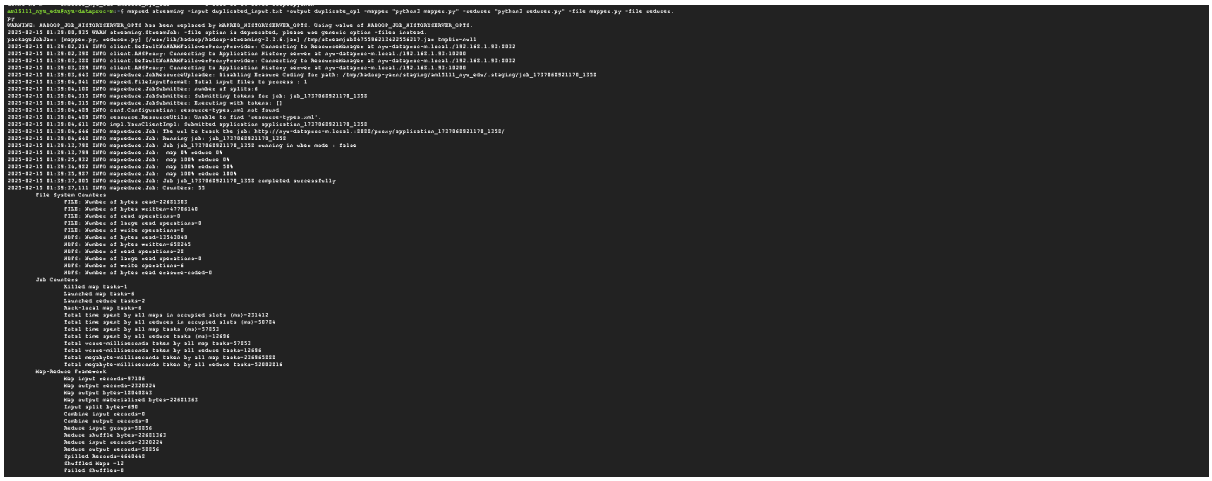
**hdfs dfs -put duplicated_input.txt**

```
am15111_nyu_edu@nyu-dataproc-m:~$ hdfs dfs -put duplicated_input.txt
am15111_nyu_edu@nyu-dataproc-m:~$ hdfs dfs -ls
Found 4 items
-rw-r--r--   1 am15111_nyu_edu am15111_nyu_edu   13521879 2025-02-15 01:36 duplicated_input.txt
-rw-r--r--   1 am15111_nyu_edu am15111_nyu_edu    2615571 2025-02-14 23:22 input.txt
drwxr-xr-x   - am15111_nyu_edu am15111_nyu_edu          0 2025-02-15 01:15 outputph
drwxr-xr-x   - am15111_nyu_edu am15111_nyu_edu          0 2025-02-14 23:25 outputpython
```

**Step 3:** running the mapper.py and reducer.py to get the word count for a. part:

**mapred streaming -input duplicated_input.txt -output duplicate_op1 -mapper "python3 mapper.py" -reducer "python3 reducer.py" -file mapper.py -file reducer.py**

```
am15111_nyu_edu@nyu-dataproc-m:~$ mapred streaming -input duplicated_input.txt -output duplicate_op1 -mapper "python3 mapper.py" -reducer "python3 reducer.py" -file mapper.py -file reducer.
py
WARNING: HADOOP_JOB_HISTORYSERVER_OPTS has been replaced by MAPRED_HISTORYSERVER_OPTS. Using value of HADOOP_JOB_HISTORYSERVER_OPTS.
2025-02-15 01:39:00,935 WARN streaming.StreamJob: -file option is deprecated, please use generic option -files instead.
packageJobJar: [mapper.py, reducer.py] [/usr/lib/hadoop/hadoop-streaming-3.3.6.jar] /tmp/streamjob8475596213422556217.jar tmpDir=null
2025-02-15 01:39:02,214 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at nyu-dataproc-m.local./192.168.1.93:8032
2025-02-15 01:39:02,398 INFO client.AHSProxy: Connecting to Application History server at nyu-dataproc-m.local./192.168.1.93:10200
2025-02-15 01:39:03,388 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at nyu-dataproc-m.local./192.168.1.93:8032
2025-02-15 01:39:03,389 INFO client.AHSProxy: Connecting to Application History server at nyu-dataproc-m.local./192.168.1.93:10200
2025-02-15 01:39:03,643 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/am15111_nyu_edu/.staging/job_1737068921170_1358
2025-02-15 01:39:04,041 INFO mapred.FileInputFormat: Total input files to process : 1
2025-02-15 01:39:04,108 INFO mapreduce.JobSubmitter: number of splits:6
2025-02-15 01:39:04,315 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1737068921170_1358
2025-02-15 01:39:04,315 INFO mapreduce.JobSubmitter: Executing with tokens: []
2025-02-15 01:39:04,489 INFO conf.Configuration: resource-types.xml not found
2025-02-15 01:39:04,489 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2025-02-15 01:39:04,611 INFO impl.YarnClientImpl: Submitted application application_1737068921170_1358
2025-02-15 01:39:04,646 INFO mapreduce.Job: The url to track the job: http://nyu-dataproc-m.local.:8088/proxy/application_1737068921170_1358/
2025-02-15 01:39:04,648 INFO mapreduce.Job: Running job: job_1737068921170_1358
```



**Step 4:** Make sure that file is being saved in the hdfs directory:

```
am15111_nyu_edu@nyu-dataproc-m:~$ hdfs dfs -ls
Found 5 items
drwxr-xr-x   - am15111_nyu_edu am15111_nyu_edu          0 2025-02-15 01:39 duplicate_op1
-rw-r--r--   1 am15111_nyu_edu am15111_nyu_edu   13521879 2025-02-15 01:36 duplicated_input.txt
-rw-r--r--   1 am15111_nyu_edu am15111_nyu_edu    2615571 2025-02-14 23:22 input.txt
drwxr-xr-x   - am15111_nyu_edu am15111_nyu_edu          0 2025-02-15 01:15 outputph
drwxr-xr-x   - am15111_nyu_edu am15111_nyu_edu          0 2025-02-14 23:25 outputpython
```

```
am15111_nyu_edu@nyu-dataproc-m:~$ hdfs dfs -ls duplicate_op1
Found 3 items
-rw-r--r--   1 am15111_nyu_edu am15111_nyu_edu          0 2025-02-15 01:39 duplicate_op1/_SUCCESS
-rw-r--r--   1 am15111_nyu_edu am15111_nyu_edu     329910 2025-02-15 01:39 duplicate_op1/part-00000
-rw-r--r--   1 am15111_nyu_edu am15111_nyu_edu     328335 2025-02-15 01:39 duplicate_op1/part-00001
```

**Step 5:** Now concatenating it and saving the text outside of hdfs and downloading it.

```
am15111_nyu_edu@nyu-dataproc-m:~$ hdfs dfs -cat duplicate_op1/part* | sort > concatenated_file_duplicate.txt
am15111_nyu_edu@nyu-dataproc-m:~$ ls
concatenated_file.txt  concatenated_file_duplicate.txt  duplicated_input.txt  input.txt  mapper.py  mappertt.py  outputpython  reducer.py  reducertt.py  top_10_frequent_words.txt
```

```
am15111_nyu_edu@nyu-dataproc-m:~$ cp concatenated_file_duplicate.txt .
cp: 'concatenated_file_duplicate.txt' and './concatenated_file_duplicate.txt' are the same file
am15111_nyu_edu@nyu-dataproc-m:~$ pwd
/home/am15111_nyu_edu
```

```
am15111_nyu_edu@nyu-dataproc-m:~$ cat concatenated_file_duplicate.txt
!"ren<:h        1
!'('{}1I('.\.I. 1
!1('Xl  1
!7llllll)I'll('.l.)'    1
!<'Nl]7£'t',    1
!IIfI,'Ct$',(l  1
!IamI——'1wi1I   1
!Iu'_/1'tfuI.u,m.       1
!WL'7t/'V-(wt)  1
!hc'r0j'ore     1
!ilce_/isft,    1
!l'Ii.r 1
!lIlIlL'C('S.i'aIy      1
!lm'funereal    1
!t'G.)'t)fl     1
!'H$€   1
!'aIa7.:'.n     1
"       159
"'      1
"'1     1
"':vl_v 1
"(.'Izildmrt,.  1
"(x'r.su:zl     1
".      1
".w     1
"/ilrigfzr,     1
"0      1
"0f"mplir,'dIl1zr       1
"1      1
":1     1
":4     1
"A      1
"Abdul, 1
"Ali,   1
"And    2
"April  2
```

The Downloaded file is concatenated_file_duplicate.txt.

**Now doing b part running the mapper.py and reducertt.py**

**Step 6:** Command:
**mapred streaming -input duplicate_op1/part-* -output duplicate_tt -mapper "python3 mapper.py" -reducer "python3 reducertt.py" -file mapper.py -file reducertt.py**

```
am15111_nyu_edu@nyu-dataproc-m:~$ mapred streaming -input duplicate_op1/part-* -output duplicate_tt -mapper "python3 mappertt.py" -reducer "python3 reducertt.py" -file mappertt.py -file red
ucertt.py
WARNING: HADOOP_JOB_HISTORYSERVER_OPTS has been replaced by MAPRED_HISTORYSERVER_OPTS. Using value of HADOOP_JOB_HISTORYSERVER_OPTS.
2025-02-15 01:55:55,302 WARN streaming.StreamJob: -file option is deprecated, please use generic option -files instead.
packageJobJar: [mappertt.py, reducertt.py] [/usr/lib/hadoop/hadoop-streaming-3.3.6.jar] /tmp/streamjob6645593606706788975.jar tmpDir=null
2025-02-15 01:55:56,462 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at nyu-dataproc-m.local./192.168.1.93:8032
2025-02-15 01:55:56,623 INFO client.AHSProxy: Connecting to Application History server at nyu-dataproc-m.local./192.168.1.93:10200
2025-02-15 01:55:57,377 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at nyu-dataproc-m.local./192.168.1.93:8032
2025-02-15 01:55:57,378 INFO client.AHSProxy: Connecting to Application History server at nyu-dataproc-m.local./192.168.1.93:10200
2025-02-15 01:55:57,639 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/am15111_nyu_edu/.staging/job_1737068921170_1381
2025-02-15 01:55:58,828 INFO mapred.FileInputFormat: Total input files to process : 2
2025-02-15 01:55:58,881 INFO mapreduce.JobSubmitter: number of splits:6
2025-02-15 01:55:59,046 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1737068921170_1381
2025-02-15 01:55:59,046 INFO mapreduce.JobSubmitter: Executing with tokens: []
2025-02-15 01:55:59,195 INFO conf.Configuration: resource-types.xml not found
2025-02-15 01:55:59,195 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2025-02-15 01:55:59,258 INFO impl.YarnClientImpl: Submitted application application_1737068921170_1381
2025-02-15 01:55:59,303 INFO mapreduce.Job: The url to track the job: http://nyu-dataproc-m.local.:8088/proxy/application_1737068921170_1381/
2025-02-15 01:55:59,305 INFO mapreduce.Job: Running job: job_1737068921170_1381
2025-02-15 01:56:13,411 INFO mapreduce.Job: Job job_1737068921170_1381 running in uber mode : false
2025-02-15 01:56:13,412 INFO mapreduce.Job:  map 0% reduce 0%
2025-02-15 01:56:21,493 INFO mapreduce.Job:  map 17% reduce 0%
2025-02-15 01:56:25,516 INFO mapreduce.Job:  map 100% reduce 0%
2025-02-15 01:56:32,566 INFO mapreduce.Job:  map 100% reduce 50%
2025-02-15 01:56:35,584 INFO mapreduce.Job:  map 100% reduce 100%
2025-02-15 01:56:37,603 INFO mapreduce.Job: Job job_1737068921170_1381 completed successfully
2025-02-15 01:56:37,694 INFO mapreduce.Job: Counters: 56
        File System Counters
                FILE: Number of bytes read=775969
                FILE: Number of bytes written=3895600
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=675343
                HDFS: Number of bytes written=191
                HDFS: Number of read operations=28
                HDFS: Number of large read operations=0
```

Step7: Since there are two files as seen using command: **hdfs dfs -ls duplicate_tt**
Need to concatenate these two files and take top 10 from both of these files, so running this command gives sorted based on maximum occurring frequency and saving it in file top_10_frequent_words.txt :
**hdfs dfs -cat duplicate_tt/part-* | sort -k2,2nr | head -n 10 > duplicate_top_10_frequent_words.txt**

```
am15111_nyu_edu@nyu-dataproc-m:~$ hdfs dfs -ls duplicate_tt
Found 3 items
-rw-r--r--   1 am15111_nyu_edu am15111_nyu_edu          0 2025-02-15 01:56 duplicate_tt/_SUCCESS
-rw-r--r--   1 am15111_nyu_edu am15111_nyu_edu         92 2025-02-15 01:56 duplicate_tt/part-00000
-rw-r--r--   1 am15111_nyu_edu am15111_nyu_edu         99 2025-02-15 01:56 duplicate_tt/part-00001
```

```
am15111_nyu_edu@nyu-dataproc-m:~$ hdfs dfs -cat duplicate_tt/part-* | sort -k2,2nr | head -n 10 > duplicate_top_10_frequent_words.txt
am15111_nyu_edu@nyu-dataproc-m:~$ ls
concatenated_file.txt            duplicate_top_10_frequent_words.txt  input.txt  mappertt.py   reducer.py    top_10_frequent_words.txt
concatenated_file_duplicate.txt  duplicated_input.txt                mapper.py  outputpython  reducertt.py
```

Step7: Seeing the content of top 10 occurring words:

```
am15111_nyu_edu@nyu-dataproc-m:~$ cat duplicate_top_10_frequent_words.txt
the     103132
to      70998
a       63932
of      59982
and     47302
you     41622
is      33059
I       32594
in      26247
your    26059
```

The Downloaded file is duplicated_top_10_frequent_words.txt.

## Q1. What difference do you notice, if any, in the output directory after running the normal word count program on the modified text file as compared to the when you ran it with the originally created text file?

The output directory shows a difference in word frequency when running the normal word count program on the modified text file compared to the original 2 MB file. Since 200-300 lines were duplicated multiple times to generate the 13 MB file, certain words appear more frequently in the modified version. As a result, both directories contain only a few other differences beyond this frequency variation.

## Q2. Why do you think there is a difference in the output while running word count despite the input files being of similar size?

The difference in the output occurs because the modified text file was created by duplicating 200-300 lines multiple times, rather than generating entirely new content. This repetition causes certain words to appear with disproportionately higher frequencies compared to the original file, even though both files are of similar size. The normal word count program processes each word as it appears in the file, leading to a skewed distribution in the modified file, where some words dominate the frequency count.

## Q3. What difference do you notice, if any, among the top 10 most frequently occurring words between the two files? Why do you think this is the case?

The top 10 most frequently occurring words differ between the 3 MB file and the 13 MB file due to the duplication of 200-300 lines multiple times in the larger file. The key differences observed are:

1. Increased Frequency for Common Words – Words like "the", "to", "a", and "of" appear much more frequently in the 13 MB file, reflecting the repeated content.
2. Ranking Changes – Some words, such as "he", "that", and "in", which were in the top 10 for the smaller file, are replaced by "I" and "your" in the larger file. This suggests that the duplicated lines contain these words more often.
3. Skewed Distribution – Since the larger file was created by duplication rather than unique content, certain words dominate the frequency count, altering the natural distribution seen in the 3 MB file.

## Learnings from the Experiment:

1. **Impact of Data Duplication** – Repeating small portions of text inflates word frequency, altering natural distributions despite similar file sizes.
2. **Importance of Unique Data** – Larger file size doesn't mean more diverse data; redundancy skews analysis results.
3. **MapReduce Behavior** – It processes all words as they appear, making input structure crucial for accurate output.
4. **Changes in Word Rankings** – Frequently repeated words dominate the top 10 in the larger file.
5. **Efficient Hadoop Usage** – Reinforced HDFS and MapReduce operations for handling large datasets.
6. **Sorting & Aggregation** – Helps consolidate results for better analysis.