BIG DATA

SECTION D

Fall’2023

Yogya Sharma

ys5250

Practice Assignment : Jupyter Notebook

Exercise 1:

For 100 strings:

```
list_of_strings = ['abc', 'bigData', 'Students']
large_list_of_strings = list_of_strings*100
%time max_length = print(find_longest_string(list_of_strings))
```

```
Students
CPU times: user 367 µs, sys: 1.02 ms, total: 1.38 ms
Wall time: 1.39 ms
```

For 1000 strings:

```
list_of_strings = ['abc', 'bigData', 'Students']
large_list_of_strings = list_of_strings*1000
%time max_length = max(large_list_of_strings, key=len)
```

```
CPU times: user 65 µs, sys: 0 ns, total: 65 µs
Wall time: 68.9 µs
```

For 1 million strings:

```
large_list_of_strings = list_of_strings*100000000
%time max_length = max(large_list_of_strings, key=len)
```

```
CPU times: user 4.93 s, sys: 56.7 ms, total: 4.98 s
Wall time: 5 s
```

For 1 million strings using map reduce:

In [5]:
```
import functools

list_of_strings = ['abc', 'bigData', 'Students']
large_list_of_strings = list_of_strings*100000000
mapper = len
def reducer(p, c):
    if p[1] > c[1]:
        return p
    return c
mapped = map(mapper, list_of_strings)
mapped = zip(list_of_strings, mapped)
#step 2:
%time reduced = functools.reduce(reducer, mapped)
print(reduced)
```

```
CPU times: user 8 µs, sys: 7 µs, total: 15 µs
Wall time: 18.8 µs
('Students', 8)
```

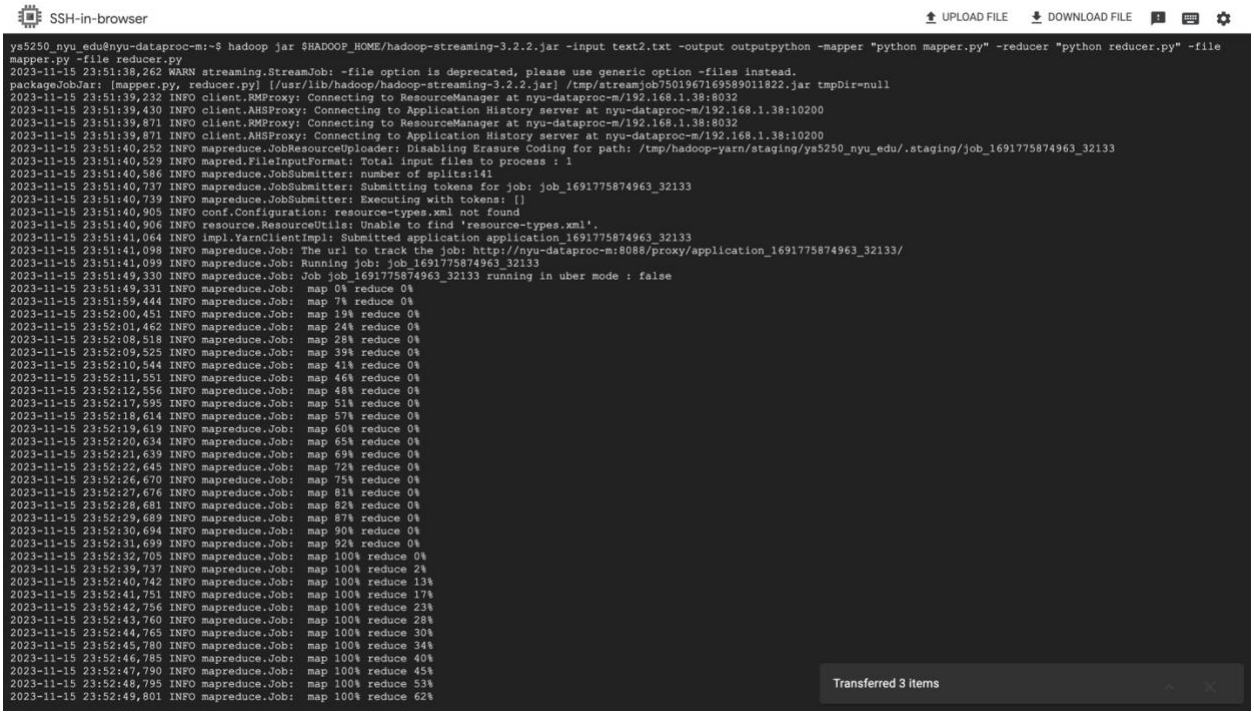
Exercise 2:

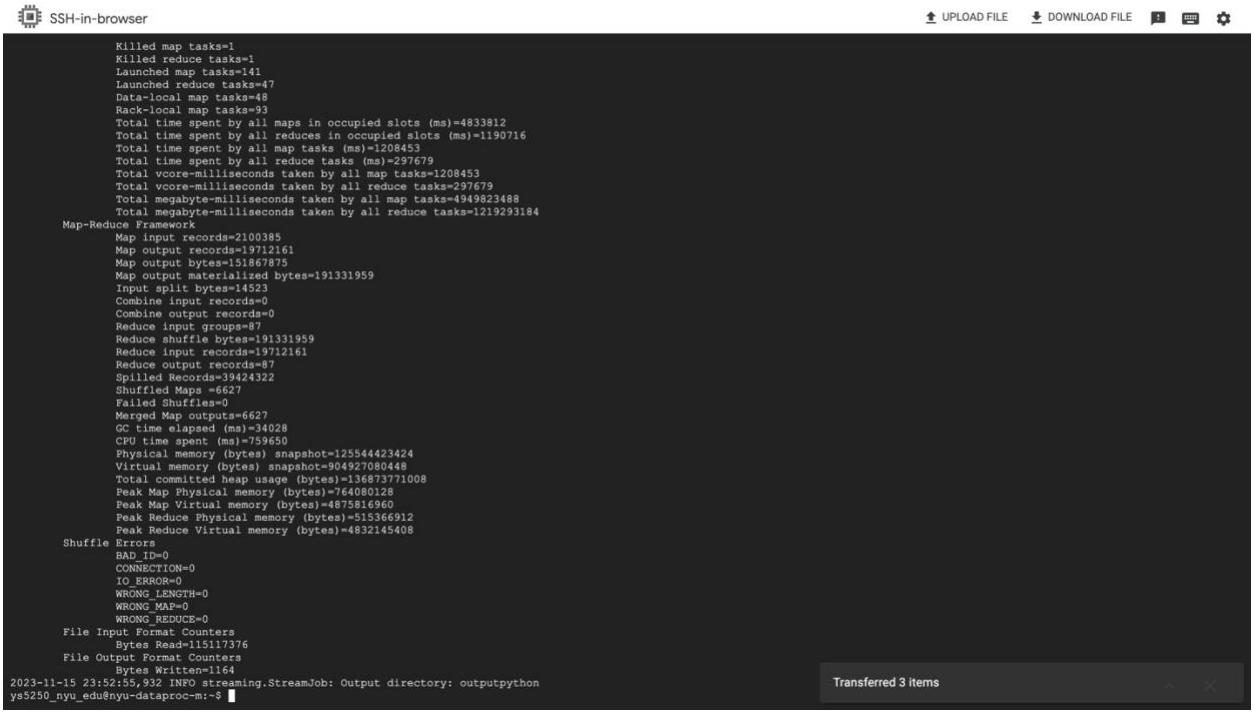
Word count on Jupyter:

In [7]:
```
import functools
from collections import Counter

# Define the mapper function to split text into words
def mapper(text):
    words = text.split()
    return Counter(words)

def reducer(d1, d2):
    return d1 + d2

# Read input text from a file
with open('text2.txt', 'r') as file:
    text = file.read()

%time word_counts = functools.reduce(reducer, map(mapper, text.splitlines()), Counter())

for word, frequency in word_counts.items():
    print(f'Word: {word}, Frequency: {frequency}')
```

```
CPU times: user 54.7 s, sys: 309 ms, total: 55 s
Wall time: 55.7 s
Word: Warning, Frequency: 123553
Word: bells, Frequency: 179712
Word: sounded, Frequency: 179712
Word: fairly, Frequency: 179712
Word: early, Frequency: 179712
Word: with, Frequency: 359424
Word: one, Frequency: 179712
Word: of, Frequency: 718848
Word: the, Frequency: 898560
Word: participants,, Frequency: 179712
Word: who, Frequency: 359424
```

Word count using map reduce Hadoop:





2023.11.15 23:51:40 UTC	2023.11.15 23:51:47 UTC	2023.11.15 23:52:53 UTC	job_1691775874963_32133	streamjob7501967169589011822.jar	ys5250_nyu_edu	nyu-dataproc-m	default	SUCCEEDED	141	141	47	47	00hrs, 01mins, 05sec

Here the elapsed time for Hadoop job is : 1 min 5 seconds

And for JupyterHub is: 56 seconds