Lab 9 Scala and pySpark

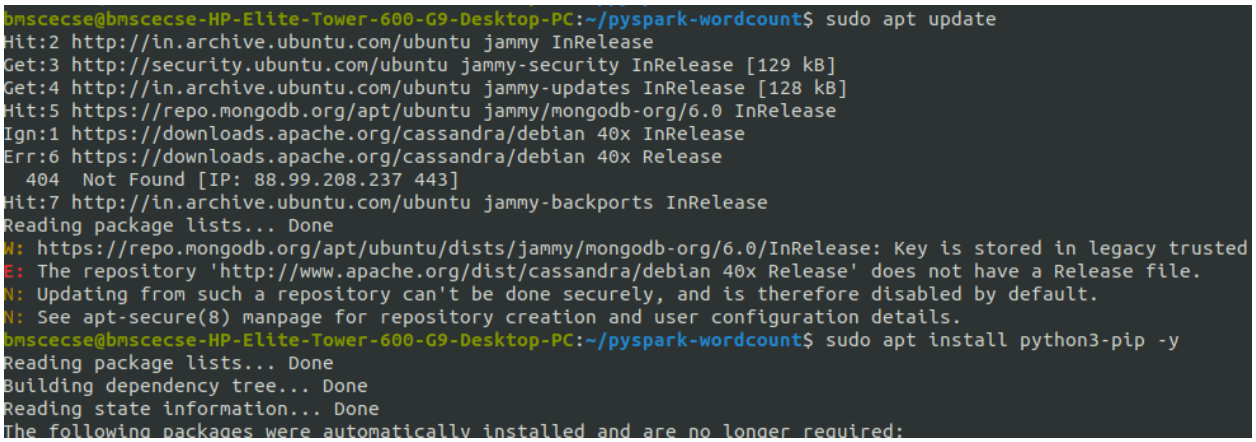
1.Write a Scala program to print numbers from 1 to 100 using for loop.

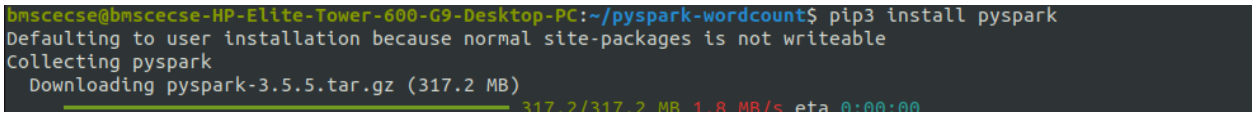
```
scala> for(i <- 1 to 100){
     | println(i)}
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
```

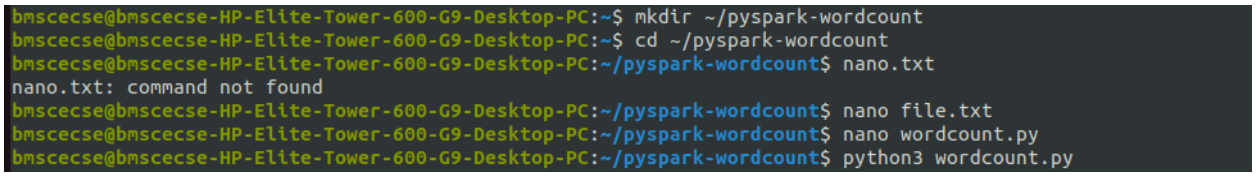
2.Using RDD and FlatMap count how many times each word appears in a file and write out a list of words whose count is strictly greater than 4 using Spark.

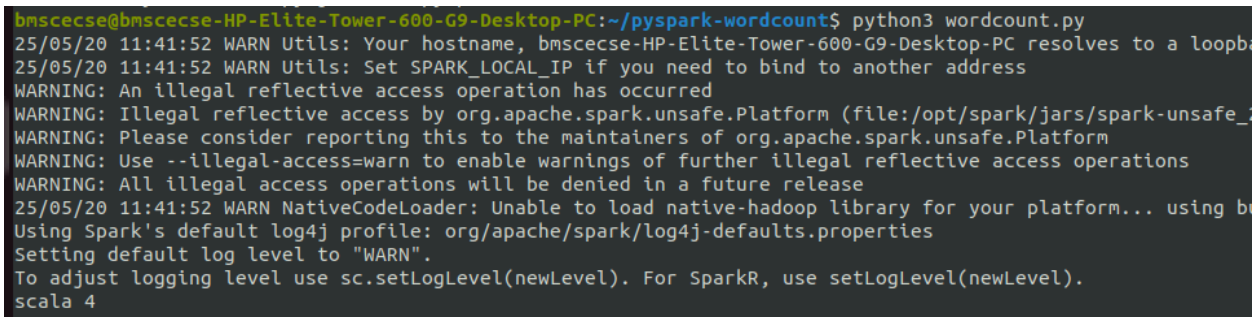
```
  GNU nano 6.2                          wordCount.py *
from pyspark import SparkContext

sc = SparkContext("local", "SimpleWordCount")


rdd = sc.textFile("text1.txt")

counts = (rdd.flatMap(lambda line: line.split())
             .map(lambda word: (word.lower(), 1))
             .reduceByKey(lambda a, b: a + b)
             .filter(lambda x: x[1] > 4))


for word, count in counts.collect():
    print(word, count)

sc.stop()
```

Spark Shell Execution Screenshots









3.Write a simple streaming program in Spark to receive text data streams on a particular port, perform basic text cleaning (like white space removal, stop words removal, lemmatization, etc.), and print the cleaned text on the screen.

```
  GNU nano 6.2                          streaming_cleaner.py *
from pyspark import SparkContext
from pyspark.streaming import StreamingContext
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
import re

# Set up Spark context and streaming context
sc = SparkContext("local[2]", "TextCleanerStreaming")
sc.setLogLevel("ERROR")
ssc = StreamingContext(sc, 5)  # 5-second batch interval

# Set of stop words and lemmatizer
stop_words = set(stopwords.words("english"))
lemmatizer = WordNetLemmatizer()

# Connect to TCP socket on localhost:9999
lines = ssc.socketTextStream("localhost", 9999)

def clean_text(line):
    # Lowercase and remove punctuation
    line = re.sub(r"[^a-zA-Z\s]", "", line.lower())
    words = line.split()
    # Remove stopwords and lemmatize
    cleaned = [lemmatizer.lemmatize(word) for word in words if word not in stop_words]
    return " ".join(cleaned)

# Clean each line and print
lines.map(clean_text).pprint()

# Start streaming
ssc.start()
ssc.awaitTermination()
```

```
bmscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC: ~
bmscecse@bmsce...  bmscecse@bmsce...  bmscecse@bmsce...  bmscecse@bmsce...
scecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~$ pip3 install nltk
faulting to user installation because normal site-packages is not writeable
llecting nltk
Downloading nltk-3.9.1-py3-none-any.whl (1.5 MB)
                                        1.5/1.5 MB 7.6 MB/s eta 0:00:00
```

Installation of Natural Language Toolkit (nltk)

nlt

```
04.5
bmscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~$ python3
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import nltk
>>> nltk.download('stopwords')
[nltk_data] Downloading package stopwords to
[nltk_data]     /home/bmscecse/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
>>> nltk.download('wordnet')
[nltk_data] Downloading package wordnet to /home/bmscecse/nltk_data...
True
>>> exit()
```

```
bmscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~$ nano streaming_cleaner.py
bmscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~$ python3 streaming_cleaner.py
25/05/20 12:05:10 WARN Utils: Your hostname, bmscecse-HP-Elite-Tower-600-G9-Desktop-PC resolv
es to a loopback address: 127.0.1.1; using 10.124.3.71 instead (on interface eno1)
25/05/20 12:05:10 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.spark.unsafe.Platform (file:/opt/spark/jars/
spark-unsafe_2.12-3.0.3.jar) to constructor java.nio.DirectByteBuffer(long,int)
WARNING: Please consider reporting this to the maintainers of org.apache.spark.unsafe.Platfor
```

Executing the streaming_cleaner.py

```
bmscecse@bmsce... × bmscecse@bmsce... × bmscecse@bmsce... × bmscecse@bmsce... ×
bmscecse@bmscecse-HP-Elite-Tower-600-G9-Desktop-PC:~$ nc -lk 9999
Spark is very powerful and fast for big data processing.
```

Starting a TCP server that listens for incoming connections on port 9999

```
-------------------------------------------
Time: 2025-05-20 12:05:55
-------------------------------------------
spark powerful fast big data processing

-------------------------------------------
Time: 2025-05-20 12:06:00
```

Output- cleaned data