Lab 7

20/8/25

## Scala

**Print 1 to 100.**

```
nano PrintNumbers.scala.
```

```
object PrintNumbers{
  def main.( args: Array[String]):Unit={
    for (i<- 1 to 100){
      println(i)
  }}}.
```

```
scalac PrintNumbers.scala

scalac PrintNumbers.
```

## Spark

List of words. is strictly greater than 4 using spark.

```
nano wordoccurance.py.

from pyspark import SparkContext

sc = SparkContext ("Local", "WordCountApp")

text rdd=sc.textfile ("input.txt")

words_rdd = text_rdd.flatMap(lambda
    line: line.split()).

pairs rdd = words_rdd.map(lambda word:
        (word, 1))

word counts rdd = pairs rdd.reduceByKey
 (lambda a, b: a+b)

filtered rdd = words counts rdd.filter(
 lambda x: x[1] > 4).

for word, count in filtered rdd.collect():
    print(f"{word}: {count}")

filtered rdd.SaveAsTextFile("output dir

nano input.txt

spark-submit wordoccurance.py nano input.txt
```

output
apple: 6
red: 10,

## Spark - Open Ended question.

TextCleaner.scala:

```
import org.apache.spark.SparkConf
import org.apache.spark.streaming._

object Textcleaner {
    def main(args: Array[String]): Unit = {
        val conf = new SparkConf().setAppName("Text Cleaner").
         setMaster("Local[*]")
        val ssc = new StreamingContext(conf, Seconds(5))

        val stopwords = set("a", "an", "the", "is", "are",
         "on", "in", "with", "to")

        val lines = ssc.socketTextStream("Localhost", 9999)

        val cleaned = lines.flatMap(_.split("\\W+")).
         map(_.toLowerCase.trim).filter(w => w.non
         Empty && !stopwords.contains(w))

        cleaned.foreachRDD(rdd => {
            println("Cleaned output:")
            rdd.collect().foreach(println)
        })
        ssc.start()
        ssc.awaitTermination()
}}
```

```
scalac -classpath "$SPARK HOME/jars/*" TextCleaner.scala

jar cf TextCleaner.jar TextCleaner*.class

spark-submit --class TextCleaner --master local[*]
TextCleaner.jar

nc -lk 9999
This is a example.
```

Output

```
This example
```