Text file in Home directory

papergrid

Date: / /

-: LAB : 09 :-

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

=> Spark-shell

```
scala > object PrintNumbers {
            def main(args: Array[String]): Unit = {
      // Loop from 1 to 100 and print each
      //number
      for(i <- 1  to 100) {
          println(i)
        }
      }
    }
  defined object PrintNumbers

scala> PrintNumbers.main(Array())
```

Output:-

```
1
2
3
,
,
,
,
,
,
:
100
```

papergrid

Date: / /

★ Using RDD and FlatMap, count how many times each word time appears in a file and write out a list of words whose count is strictly greater than & using spark

=> textfile (sc.textfile)

```
val data = sc.textFile("sparkdata.txt")
data.collect;
val splitdata = data.flatMap(line => line.
split(" "));

splitdata.collect;

val mapdata = splitdata.map(word => (word,1))

mapdata.collect;

val reducedata = mapdata.reduceByKey(_+_);

reducedata.collect;
```

Output :-

```
Hello: 5
 Tree: 4
```

//

Hadoop program {open ~~ed~~. ended}

→ Mapper.py

```
/usr/bin/env python3
import sys
import re

for line in sys.stdin
    line = line.strip().lower()

words = re.fridall(r' \b[a-z]+\b', line)
  for word in words:
      print(f"{word}\t1")
```

→ Reducer.py

```
/usr/bin/env python3

import sys
from collections improve default dict

word.count. default dist(int)

for line in sys.std in:
    line = line.strip()
    if not line
      continue
    word, count = line.split('\t')
    word.count[word] += int(count)
```

```
top_words = sorted(word_count.items(),
   key=(lambda x: (-x[1], x[0]))[:10]

for word, count in top_words:
   print(f"{word}\t{count}")

→ hadoop jar /home/bmsce/had.jar \
  -mapper mapper.py \
  -reducer reducer.py \
  -file mapper.py \
  -file reducer.py
```

[illegible] 20/5/25