Source Code:

```
import org.apache.spark.SparkContext

var sc = new SparkContext()
var map = sc.textFile("/mnt/spark/bin/new.txt").flatMap(line => line.split(" ")).map(word
=>(word,1));

println(map)


var counts = map.reduceByKey(_ + _);
println("count=", counts.count)

val keysRdd = counts.keys

val sorted = keysRdd.sortBy(x=> x, true)

sorted.collect

val m = counts.max

counts.saveAsTextFile("/mnt/spark/bin/WD_txt")
sorted.saveAsTextFile("/mnt/spark/bin/sort_txt")
```

Output:

part-00001 — /mnt/spark/bin/sort_txt

```
1 it
2 item
3 key
4 keyBy
5 look
6 many
7 mapped
8 may
9 method
10 method.
11 methods
12 number
13 of
14 one
15 operation.
16 or
17 order
18 output
19 pair
20 pair,
21 pairs.
22 parallelism
23 parallelism.
24 parameters.
25 partitions
26 partitions.
27 performance
28 pipe
29 process.
30 program
31 provide
32 randomSplit
33 redistributes
34 reduces
35 reducing
36 repartition
37 returned
```

part-00001 — /mnt/spark/bin/WD_txt

```
1 sortBy,2
2 (increasing,1)
3 (returns,5)
4 (Reducing,1)
5 (For,1)
6 (Spark,1)
7 (it,2)
8 (forked,1)
9 (sorted,1)
10 (The,12)
11 (similar,,1)
12 (number,5)
13 (such,1)
14 (argument,1)
15 (each,2)
16 (while,2)
17 (difference,1)
18 (external,2)
19 (in,10)
20 (provide,1)
21 (weights,1)
22 (look,1)
23 (In,1)
24 (reduces,1)
25 (for,6)
26 (should,1)
27 (used,3)
28 (similar,1)
29 (item,3)
30 (execute,1)
31 (process.,1)
32 (benefit,1)
33 (partitions.,1)
34 (may,2)
35 (the,27)
36 (partitions,6)
37 (splits,2)
```