Spark Training questions

Please answer the questions below

Exercise 1

Q1. Please put your code here:

```
import sys
from pyspark import SparkContext, SparkConf

if __name__ == "__main__":
    if len(sys.argv) != 2:
        print("Usage: wordcount <input_folder>", file=sys.stderr)
        sys.exit(-1)

    conf = SparkConf().setAppName("python-word-count")
    sc = SparkContext(conf=conf)

    text_file = sc.textFile("hdfs://" + sys.argv[1])
    rdd1 = text_file.flatMap(lambda line: line.split(" "))
    rdd2 = rdd1.map(lambda word: (word, 1))
    rdd3 = rdd2.reduceByKey(lambda a, b: a + b)
    rdd4 = rdd3.filter(lambda x: len(x[0]) > 5)

    counts = rdd4.repartition(5)

    print("—RDD1----------------------------------")
    print(rdd1.take(5))
    print("—RDD2----------------------------------")
    print(rdd2.take(5))
    print("—RDD3----------------------------------")
    print(rdd3.take(5))
    print("—counts---------------------------------")
    print(counts.take(5))

    result = counts.takeOrdered(40, key=lambda x: -x[1])
    print("-----------------------------------------")
    print(*result, sep="\n")
    print("-----------------------------------------")
```

Q2. Add print-screen of the stage proving you have 5 tasks

Stage Id	Description	Submitted	Duration	Tasks: Succeeded/Total	Input	Output	Shuffle Read	Shuffle Write
2	takeOrdered at /home/hadoop/course/spark-word-count.py:20 +details	2025/12/15 15:27:54	0.1 s	5/5			1433.1 KiB	
1	reduceByKey at /home/hadoop/course/spark-word-count.py:16 +details	2025/12/15 15:27:53	1.0 s	5/5			7.0 MiB	1433.1 KiB

Exercise 2

Q1. Please put your code here:

```
import sys
from pyspark import SparkContext, SparkConf

if __name__ == "__main__":
    if len(sys.argv) != 2:
        print("Usage: wordcount <input_folder>", file=sys.stderr)
        sys.exit(-1)

    conf = SparkConf().setAppName("python-word-count")
    sc = SparkContext(conf=conf)

    text_file = sc.textFile("hdfs://" + sys.argv[1])
    rdd1 = text_file.flatMap(lambda line: line.split(" ")).cache()
    rdd2 = rdd1.map(lambda word: (word, 1))
    rdd3 = rdd2.reduceByKey(lambda a, b: a + b)

    counts = rdd3.repartition(5)

    print("—RDD1----------------------------------")
    print(rdd1.take(5))
    print("—RDD2----------------------------------")
    print(rdd2.take(5))
    print("—RDD3----------------------------------")
    print(rdd3.take(5))
    print("—counts----------------------------------")
    print(counts.take(5))

    result = counts.takeOrdered(40, key=lambda x: -x[1])
    print("-----------------------------------------")
    print(*result, sep="\n")
    print("-----------------------------------------")

    distinct_wc = rdd1.distinct().count()
    print("-----------------------------------------")
    print(distinct_wc)
    print("-----------------------------------------")
```

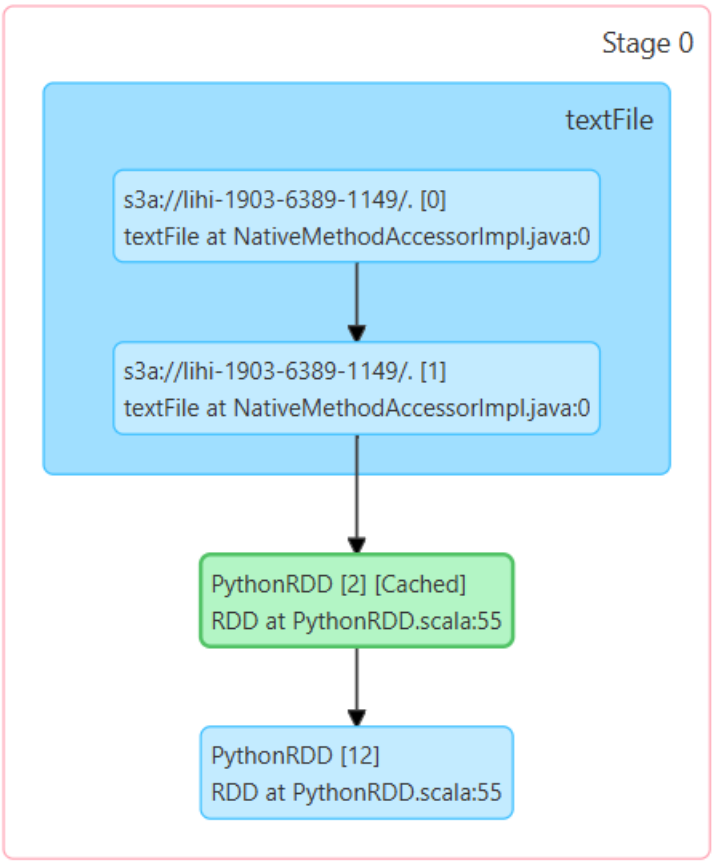
Q2. Write the number of words found

```
25/12/15 15:39:53 INFO YarnScheduler: Removed TaskSet 11.0, whose tasks have all completed, from pool
25/12/15 15:39:53 INFO DAGScheduler: ResultStage 11 (count at /home/hadoop/course/spark-word-count.py:33) finished in 0.
102 s
25/12/15 15:39:53 INFO DAGScheduler: Job 5 is finished. Cancelling potential speculative or zombie tasks for this job
25/12/15 15:39:53 INFO YarnScheduler: Killing all running tasks in stage 11: Stage finished
25/12/15 15:39:53 INFO DAGScheduler: Job 5 finished: count at /home/hadoop/course/spark-word-count.py:33, took 0.525335
s
-----------------------------------------
77928
-----------------------------------------
25/12/15 15:39:53 INFO SparkContext: Invoking stop() from shutdown hook
25/12/15 15:39:53 INFO SparkContext: SparkContext is stopping with exitCode 0.
25/12/15 15:39:53 INFO SparkUI: Stopped Spark web UI at http://ip-172-31-29-215.ec2.internal:4040
25/12/15 15:39:53 INFO YarnClientSchedulerBackend: Interrupting monitor thread
25/12/15 15:39:53 INFO YarnClientSchedulerBackend: Shutting down all executors
25/12/15 15:39:53 INFO YarnSchedulerBackend$YarnDriverEndpoint: Asking each executor to shut down
25/12/15 15:39:53 INFO YarnClientSchedulerBackend: YARN client scheduler backend Stopped
25/12/15 15:39:53 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
25/12/15 15:39:53 INFO MemoryStore: MemoryStore cleared
25/12/15 15:39:53 INFO BlockManager: BlockManager stopped
25/12/15 15:39:53 INFO BlockManagerMaster: BlockManagerMaster stopped
25/12/15 15:39:53 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
25/12/15 15:39:53 INFO SparkContext: Successfully stopped SparkContext
25/12/15 15:39:53 INFO ShutdownHookManager: Shutdown hook called
25/12/15 15:39:53 INFO ShutdownHookManager: Deleting directory /mnt/tmp/spark-6ec74f93-4293-413a-a9a8-6a1ada165d4b
25/12/15 15:39:53 INFO ShutdownHookManager: Deleting directory /mnt/tmp/spark-478cc897-711b-40a0-8411-a73b77949d0a
25/12/15 15:39:53 INFO ShutdownHookManager: Deleting directory /mnt/tmp/spark-6ec74f93-4293-413a-a9a8-6a1ada165d4b/pyspa
rk-3718be6d-e1ba-4a41-a009-a5a7685ba86b
[hadoop@ip-172-31-29-215 course]$
```

Exercise 3

Put a print-scrin with the DAG of the first stage, which shows it reads the files from s3a://<your_bucket_name>

▾ DAG Visualization



Exercise 4

Q1. Please put your code here:

```
import sys
from pyspark import SparkContext, SparkConf

def clean_word(word):
    word = word.rstrip(".,")
    return word

if __name__ == "__main__":
    if len(sys.argv) != 2:
        print("Usage: wordcount <s3_path>", file=sys.stderr)
```

```
    sys.exit(-1)

  conf = SparkConf().setAppName("find-longest-word")
  sc = SparkContext(conf=conf)

  text_file = sc.textFile("s3a://" + sys.argv[1])

  rdd1 = text_file.flatMap(lambda line: line.split(" "))
  cleaned = rdd1.map(lambda word: clean_word(word)).filter(lambda w: w.isalpha())

  # reduce to find the longest word
  longest_word = cleaned.reduce(lambda a, b: a if len(a) > len(b) else b)

  print("-------------------------------------------")
  print(f"The longest word is: {longest_word} (length {len(longest_word)})")
  print("-------------------------------------------")
```

Q2. Put here the printout of the longest word:

```
25/12/15 16:08:19 INFO DAGScheduler: Job 0 finished: reduce at /home/hadoop/course/spark-word-count.py:22, took 2.774899 s
-------------------------------------------
The longest word is: straightforwardness (length 19)
-------------------------------------------
25/12/15 16:08:19 INFO SparkContext: Invoking stop() from shutdown hook
```

Exercise 5

Q1. Please put your code here:

```
import sys
from pyspark import SparkContext, SparkConf

if __name__ == "__main__":
  if len(sys.argv) != 2:
    print("Usage: count_words_in_line <input_folder>", file=sys.stderr)
    sys.exit(-1)

  conf = SparkConf().setAppName("max-words-in-line")
  sc = SparkContext(conf=conf)

  text_file = sc.textFile("s3a://" + sys.argv[1])

  line_word_counts = text_file.map(
    lambda line: (line, len(line.split()))
  )

  max_line = line_word_counts.reduce(
    lambda a, b: a if a[1] >= b[1] else b
  )
```

```
print("----------------------------------------")
print("Line with most words:")
print(max_line[0])
print("Word count:", max_line[1])
print("----------------------------------------")
```

Q2. Put here the printout of the line with the most words: