Interview Preparation 3
Assignment
Spark Streaming Analysis

- **First Part**
  You have to create a Spark Application which streams data from a file on local directory on your machine and does the word count on the fly. The word should be done by the spark application in such a way that as soon as you drop the file in your local directory, your spark application should immediately do the word count for you.

CODE:

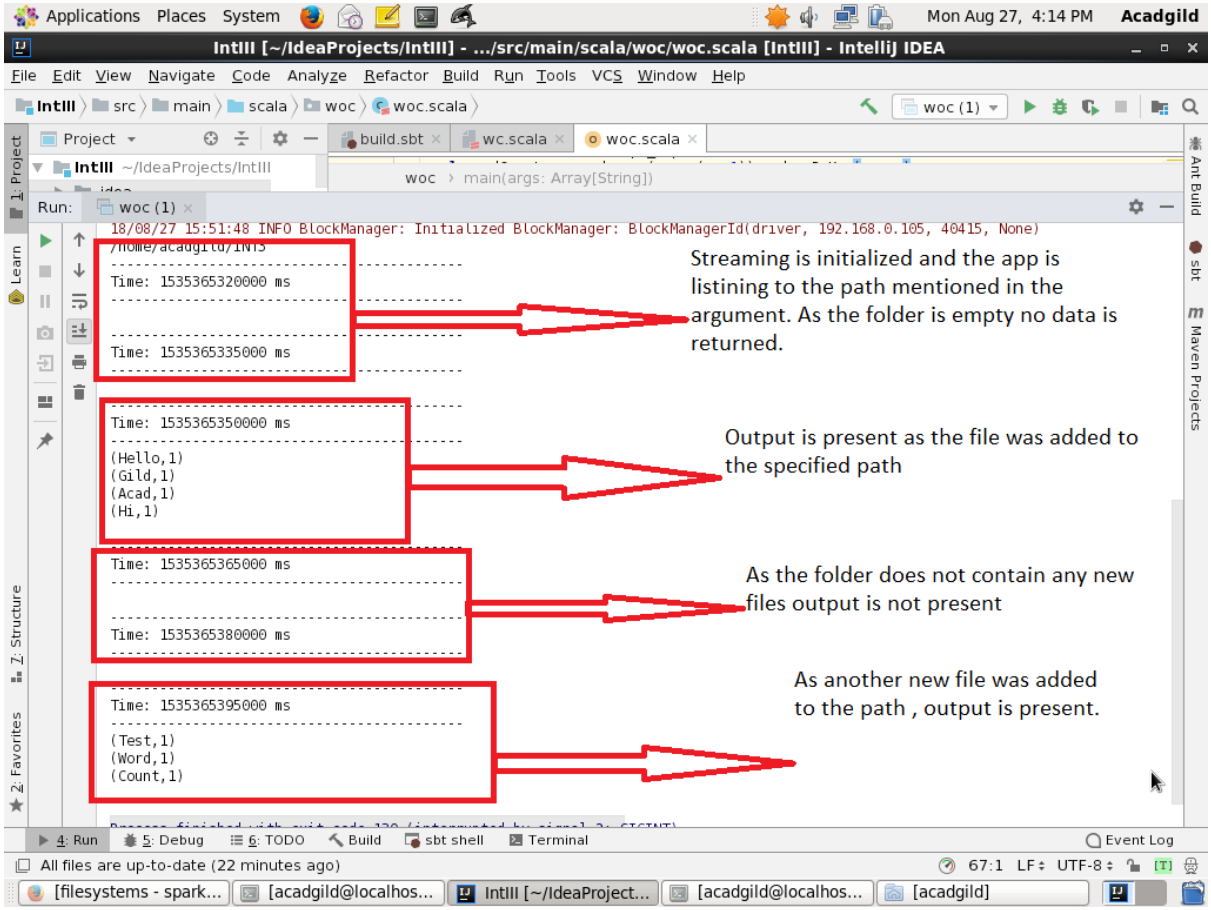
```
package woc
import org.apache.spark.{SparkConf, SparkContext}
import org.apache.spark.streaming.{Seconds, StreamingContext}
import org.apache.log4j.{Level,Logger}

object woc {
  def main(args: Array[String]): Unit = {

//Printing the argument passed
    println(args(0))
// Creating spark configuration
    val conf = new SparkConf().setMaster("local[2]").setAppName("SparkSteamingExample")
//creating spark context
val sc = new SparkContext(conf)
    val rootLogger =Logger.getRootLogger()
    rootLogger.setLevel(Level.ERROR)
//Creating streaming context and specifying wait time
val ssc = new StreamingContext(sc, Seconds(15))
    println(args(0))
//reading the lines from the path specified [ If a file is present ]
    val lines = ssc.textFileStream(args(0))
//split the lines to words using space as seperator
    val words = lines.flatMap(_.split(" "))
//Counting the number of words using map and reduce method
// _+_ is used for summation
    val wordCounts = words.map(x => (x, 1)).reduceByKey(_ + _)
//Print word count
 wordCounts.print()
//start streaming
//all the above process will take place only after streaming is initiated
    ssc.start()
    ssc.awaitTermination()

  }
}
```

OUTPUT:



- **Second Part**
  In this part, you will have to create a Spark Application which should do the following
  - Pick up a file from the local directory and do the word count
  - Then in the same Spark Application, write the code to put the same file on HDFS.
  - Then in same Spark Application, do the word count of the file copied on HDFS in step 2
  - Lastly, compare the word count of step 1 and 2. Both should match, other throw an error

CODE:

```
package woc
import java.io.File
import org.apache.spark.{SparkConf, SparkContext}
import scala.io.Source._
import org.apache.log4j.{Level,Logger}


object hdwc {
```

```
// Provide Local File Path

  private var localFilePath: File = new File("/home/acadgild/INT3/wc")
//hdfs file path
  private var dfsDirPath: String = "hdfs://localhost:8020/user/acadgild"

  def main(args: Array[String]): Unit = {

    println("Performing local word count")
// readFile function reads the file
//runLocalWordCount function does local wordcount
    val fileContents = readFile(localFilePath.toString())

    val localWordCount = runLocalWordCount(fileContents)

//print Local word count
    println("Local Word Count is ->>"+localWordCount)

   val conf = new
SparkConf().setMaster("local[2]").setAppName("SparkHDFSWordCountComparisonApp")
//create sparkcontext
    val sc = new SparkContext(conf)
    val rootLogger =Logger.getRootLogger()
   rootLogger.setLevel(Level.ERROR)
    println("Writing local file to DFS")
    val dfsFilename = dfsDirPath + "/dfs_read_write_test"
//create the rdd of the file contents
    val fileRDD = sc.parallelize(fileContents)
//save the rdd to the path specified
    fileRDD.saveAsTextFile(dfsFilename)
    println("Reading file from DFS and running Word Count")
//read the contects from hdfs file and count the number of words
    val readFileRDD = sc.textFile(dfsFilename)
    val dfsWordCount = readFileRDD
      .flatMap(_.split(" "))
      .flatMap(_.split("\t"))
      .filter(_.nonEmpty)
      .map(w => (w, 1))
      .countByKey()
      .values
      .sum
//print word count of the hdfs file
    println("DFS word count is "+ dfsWordCount)
//stop the spark context
  sc.stop()

//Comparing local and hdfs word count
    if (localWordCount == dfsWordCount) {
      println(s"Local Word Count and DFS Word Count are same")
    } else {
      println(s"Local Word Count and DFS Word count are not same")
    }

  }
```

//User definesd functions for reading file and performing word count for the file present in local FS

```
  private def readFile(filename: String): List[String] = {
    val lineIter: Iterator[String] = fromFile(filename).getLines()
    val lineList: List[String] = lineIter.toList
    lineList
  }

  def runLocalWordCount(fileContents: List[String]): Int = {
    fileContents.flatMap(_.split(" "))
      .flatMap(_.split("\t"))
      .filter(_.nonEmpty)
      .groupBy(w => w)
      .mapValues(_.size)
      .values
      .sum
  }

}
```

OUTPUT:

```
n:  hdwc
/usr/java/jdk1.8.0_151/bin/java ...
Performing local word count
Local Word Count is ->>4
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
18/08/27 17:33:47 INFO SparkContext: Running Spark version 2.2.1
18/08/27 17:33:49 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform
18/08/27 17:33:51 WARN Utils: Your hostname, localhost.localdomain resolves to a loopback addre
18/08/27 17:33:51 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
18/08/27 17:33:52 INFO SparkContext: Submitted application: SparkHDFSWordCountComparisonApp
18/08/27 17:33:52 INFO SecurityManager: Changing view acls to: acadgild
18/08/27 17:33:52 INFO SecurityManager: Changing modify acls to: acadgild
18/08/27 17:33:52 INFO SecurityManager: Changing view acls groups to:
18/08/27 17:33:52 INFO SecurityManager: Changing modify acls groups to:
```

```
18/08/27 17:33:56 INFO BlockManager: Using org.apache.spark.storage.RandomB
18/08/27 17:33:56 INFO BlockManagerMaster: Registering BlockManager BlockMa
18/08/27 17:33:56 INFO BlockManagerMasterEndpoint: Registering block manage
18/08/27 17:33:56 INFO BlockManagerMaster: Registered BlockManager BlockMan
18/08/27 17:33:56 INFO BlockManager: Initialized BlockManager: BlockManager
Writing local file to DFS
Reading file from DFS and running Word Count
DFS word count is 4
Local Word Count and DFS Word Count are same

Process finished with exit code 0
```