**Session 20: SPARK STREAMING**

**Assignment 20.1**

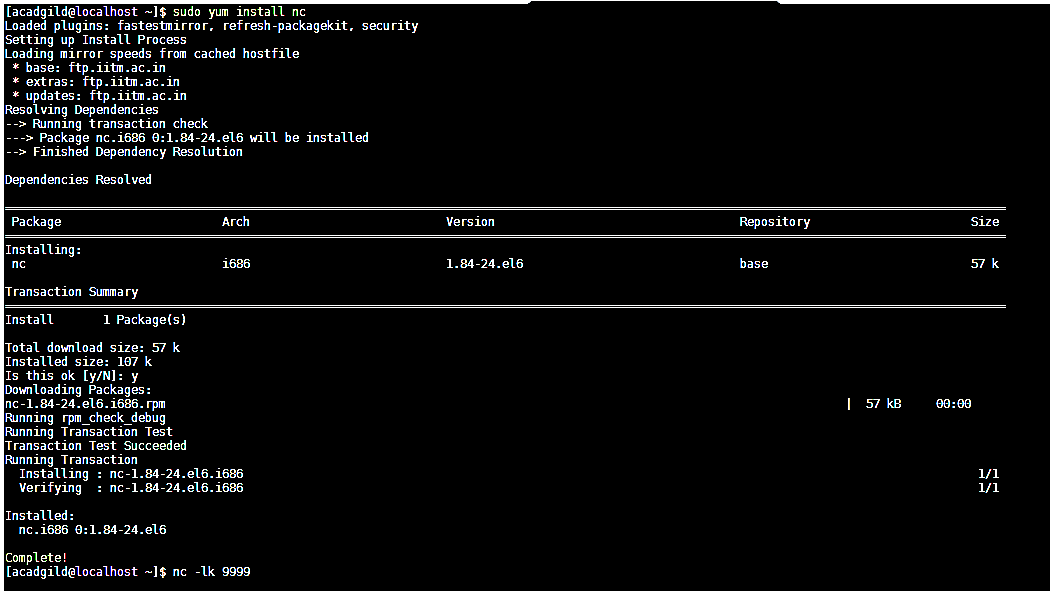
**Problem Statement**

1. Read a stream of Strings, fetch the words which can be converted to numbers. Filter out the rows, where the sum of numbers in that line is odd.
2. Provide the sum of all the remaining numbers in that batch.

**Pre requisite**

In this assignment we are going to read the strings which is captured in a port using **netcat** and hence we are installing the **netcat** in our server,

Command - sudo yum install nc



Start listening the port **9999** using the below command,

Command- **nc –lk 9999**



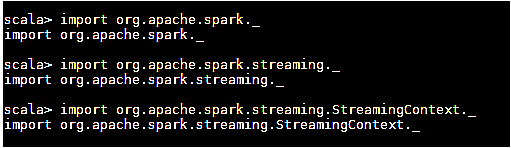
Start Spark shell with multi threads, in this case we are taking 4 multi threads.

Command- ***/home/acadgild/spark-2.2.1-bin-hadoop2.6.bin/spark-shell --master local[4]***

**1.Read a stream of Strings, convert it to numbers, sum of Numbers - operations**

**Step – 1** - Import all the spark streaming packages

* import org.apache.spark.\_
* import org.apache.spark.streaming.\_
* import org.apache.spark.streaming.StreamingContext.\_



**Step – 2** - Define an Accumulator

Defining an accumulator **“EvenLines”** which will keep track of sum of number of word numbers in lines so far,

**val EvenLines = sc.accumulator(0)**

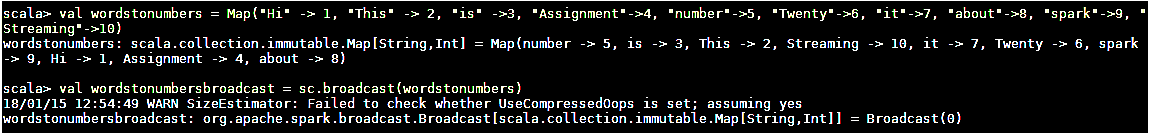


**Step – 3** - convert words to numbers

We are creating a RDD which maps the strings into the corresponding numbers, if we provide any word in the port **9999** which is not mapped, the numerical 0 will be returned.

Broadcast the newly created map,

* **val wordstonumbers = map(“Hi”->1, “This”->2, “is”->3, “Assignment”->4, “number”->5, “Twenty”->6,”it”->7, “about”->8, “spark”->9, “Streaming”->10)**
* **val wordstonumbersbroadcast = sc.broadcast(wordstonumbers)**



**Step – 4** - create a function to return sum of word converted to number in a line

Create a function “***lineWordNumberSum”*** where we are splitting a line based on blank space to get all the words in next. In the lookup value, we are determining corresponding numbers for a word in the **wordstonumbersbroadcast** and we adding the all the numbers.

Please see the code below,

***def lineWordNumberSum(line:String):Int = {***

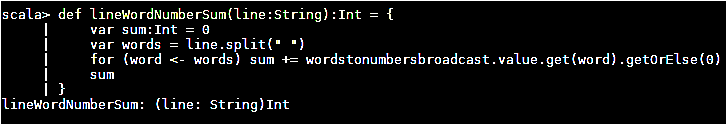
var sum:Int = 0

var words = line.split(" ")

***for (word <- words) sum += wordstonumbersbroadcast.value.get(word).getOrElse(0)***

sum

***}***

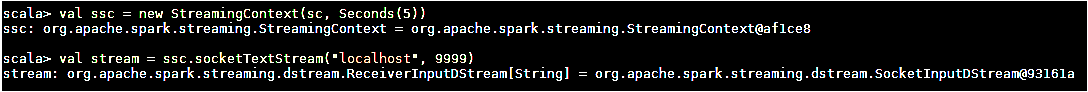


**Step – 5** - Text Streaming

In this step, we are streaming the data as a string in a 5 seconds interval and return the stream. The streams are reading in a port 9999 which is listened

***val ssc = new StreamingContext(sc, Seconds(5))***

***val stream = ssc.socketTextStream("localhost", 9999)***



**Step – 6** - Processing the each RDD

Process each RDD in stream, we are converting the RDD to string. Consider the below scenarios,

If it is not blank calculate corresponding word’s number and sum them using the function ***lineWordNumberSum*** and put as variable **numTotal.**

If **numTotal** is odd, print the provided line in the output, else add **numTotal** to accumulator **EvenLines** and print the sum.

Code,

***stream.foreachRDD(line => {val lineStr = line.collect().toList.mkString("")***

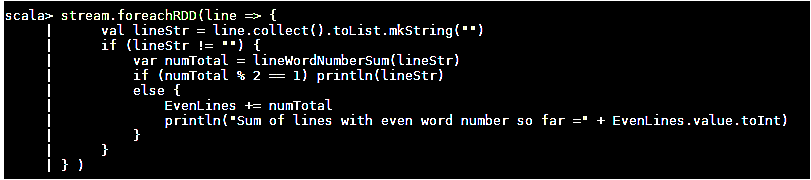
***if (lineStr != "") {var numTotal = lineWordNumberSum(lineStr) if (numTotal % 2 == 1) println(lineStr)***

***else***

***{EvenLines += numTotal***

***println("Sum of lines with even word number so far =" + EvenLines.value.toInt)}}***

***} )***

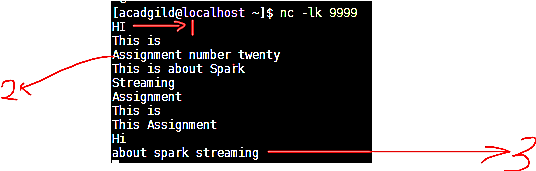


**Step – 7** - Spark Streaming

Now, Start the streams and wait till its termination

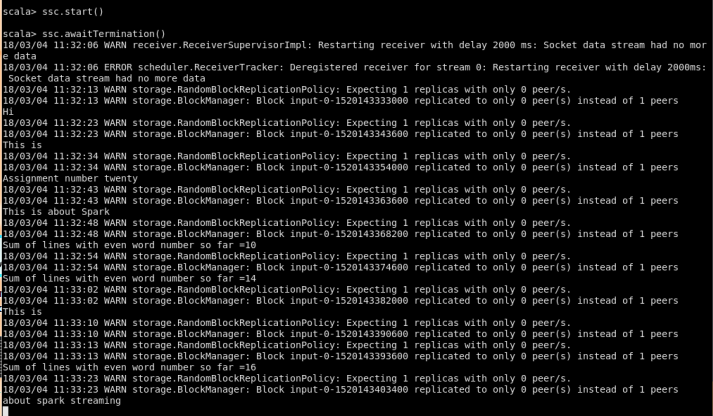
* **ssc.start()**
* **ssc.awaitTermination()**

Start **netcat** in the another terminal, provide the texts which we determined in the map.



1. We know that the string **“Hi”** has a number 1, when we type “**Hi**” in the port which is ODD, hence it will be displayed.
2. Same as, “Assignment number twenty” which has value of 15 which is again ODD and hence it is displayed,
3. For lines with even numbered word number, the **summation done so far** will be displayed. Please see the below screen shot,

**Output**

****