Session 24: SPARK STREAMING

Assignment 1

Task 1

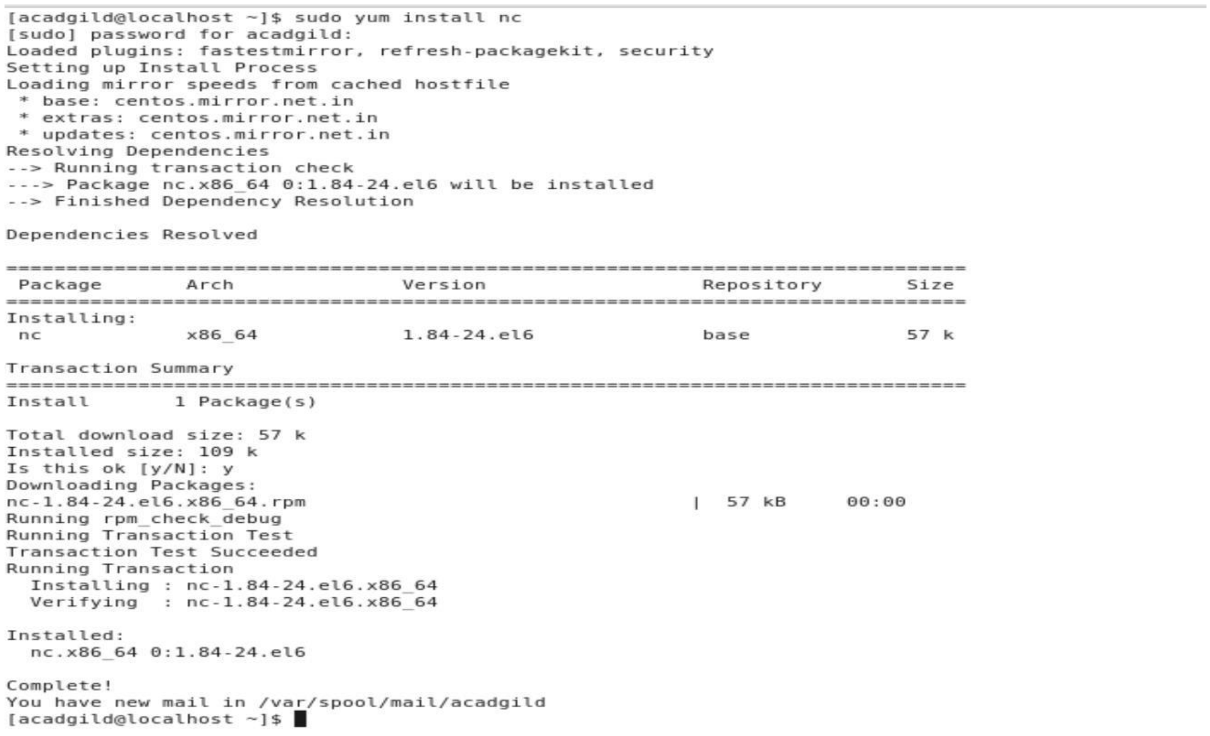
Read a stream of Strings, fetch the words which can be converted to numbers. Filter out the rows, where the sum of numbers in that line is odd.

Provide the sum of all the remaining numbers in that batch.

Solution: 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**

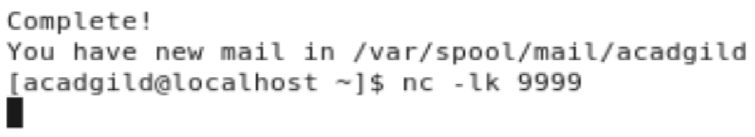
The below screenshot shows the successful installation of **netcat** using **yum**



```
[acadgild@localhost ~]$ sudo yum install nc
[sudo] password for acadgild:
Loaded plugins: fastestmirror, refresh-packagekit, security
Setting up Install Process
Loading mirror speeds from cached hostfile
 * base: centos.mirror.net.in
 * extras: centos.mirror.net.in
 * updates: centos.mirror.net.in
Resolving Dependencies
--> Running transaction check
---> Package nc.x86_64 0:1.84-24.el6 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package       Arch              Version                  Repository      Size
================================================================================
Installing:
 nc            x86_64            1.84-24.el6              base            57 k

Transaction Summary
================================================================================
Install       1 Package(s)

Total download size: 57 k
Installed size: 109 k
Is this ok [y/N]: y
Downloading Packages:
nc-1.84-24.el6.x86_64.rpm                                 |  57 kB     00:00
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : nc-1.84-24.el6.x86_64
  Verifying  : nc-1.84-24.el6.x86_64

Installed:
  nc.x86_64 0:1.84-24.el6

Complete!
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$ █
```

Start listening the port 9999 using below command

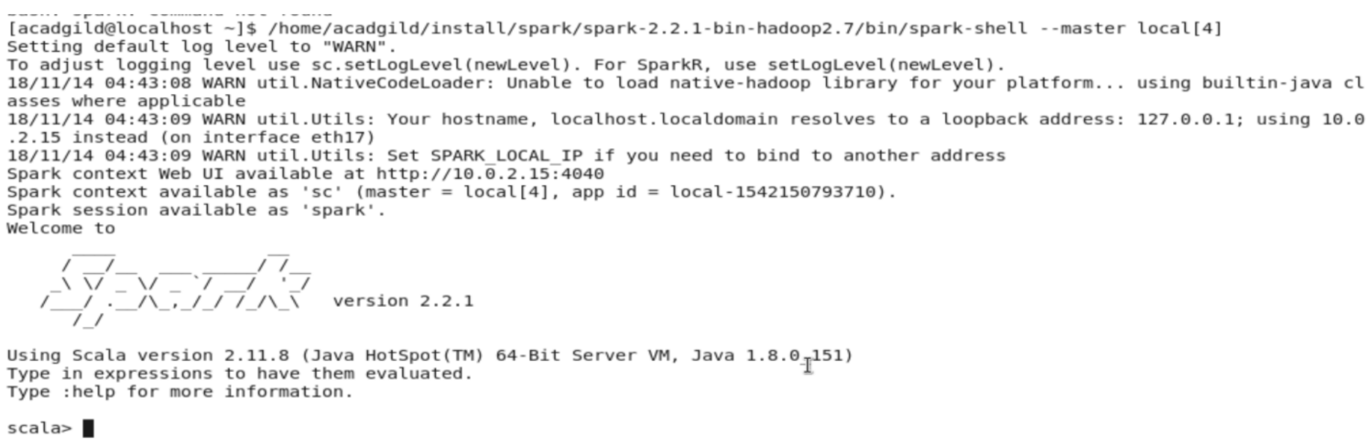
Command: **nc –lk 9999**



```
Complete!
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$ nc -lk 9999
█
```

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

/home/acadgild/spark-2.2.1-bin-hadoop2.7/bin/spark-shell --master local[4]

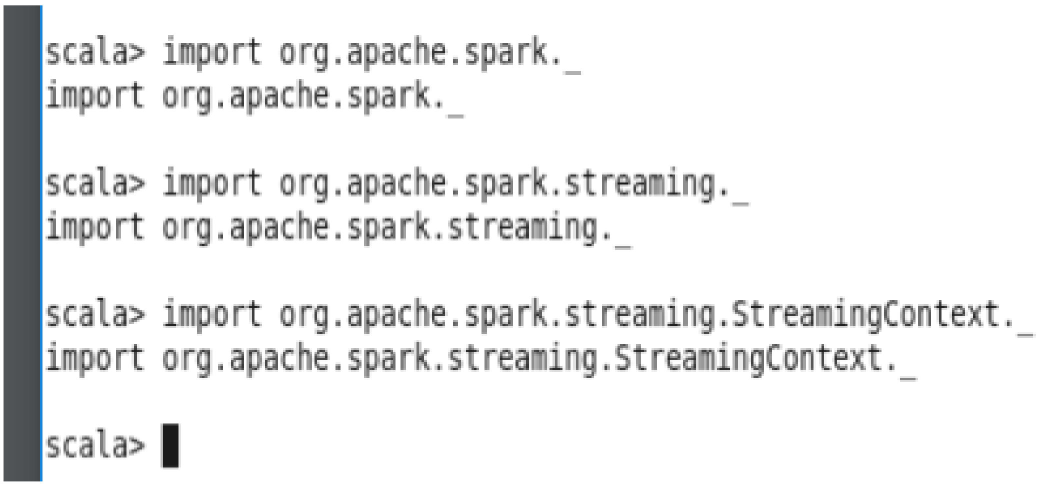


```
[acadgild@localhost ~]$ /home/acadgild/install/spark/spark-2.2.1-bin-hadoop2.7/bin/spark-shell --master local[4]
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
18/11/14 04:43:08 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
18/11/14 04:43:09 WARN util.Utils: Your hostname, localhost.localdomain resolves to a loopback address: 127.0.0.1; using 10.0
.2.15 instead (on interface eth17)
18/11/14 04:43:09 WARN util.Utils: Set SPARK_LOCAL_IP if you need to bind to another address
Spark context Web UI available at http://10.0.2.15:4040
Spark context available as 'sc' (master = local[4], app id = local-1542150793710).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.2.1
      /_/

Using Scala version 2.11.8 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_151)
Type in expressions to have them evaluated.
Type :help for more information.

scala> █
```

Read a stream of Strings, convert it to numbers, sum of Numbers – operations

Step 1: Import the spark streaming libraries

- import org.apache.spark._
- import org.apache.spark.streaming._
- import org.apache.spark.streaming.StreamingContext._

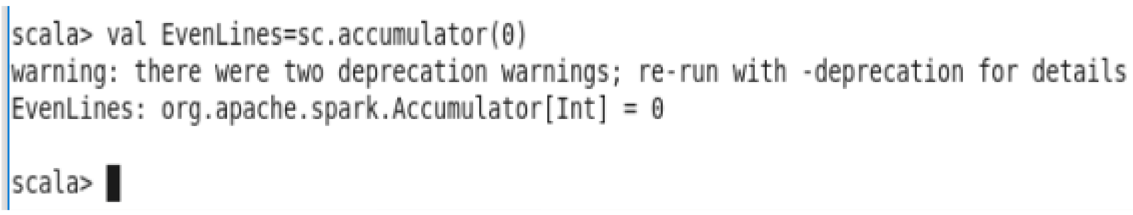


```
scala> import org.apache.spark._
import org.apache.spark._

scala> import org.apache.spark.streaming._
import org.apache.spark.streaming._

scala> import org.apache.spark.streaming.StreamingContext._
import org.apache.spark.streaming.StreamingContext._

scala> █
```

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)



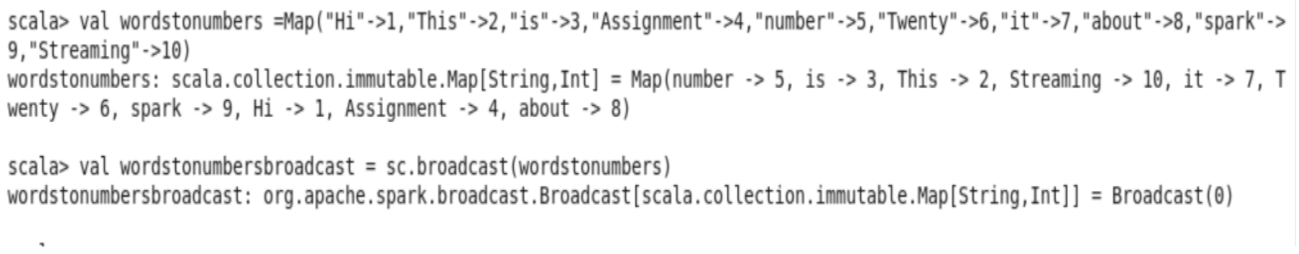
```
scala> val EvenLines=sc.accumulator(0)
warning: there were two deprecation warnings; re-run with -deprecation for details
EvenLines: org.apache.spark.Accumulator[Int] = 0

scala> ▌
```

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,

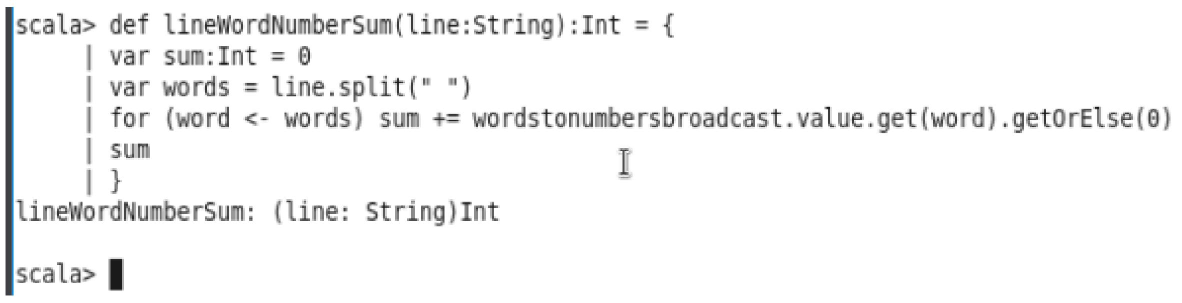
- val wordstonumbers = map(“Hi”->1, “This”->2, “is”->3, “Assignment”->4, “number”->5, “Twenty”->6,”it”->7, “about”->8, “spark”->9, “Streaming”->10)
- val wordstonumbersbroadcast = sc.broadcast(wordstonumbers)



```
scala> val wordstonumbers =Map("Hi"->1,"This"->2,"is"->3,"Assignment"->4,"number"->5,"Twenty"->6,"it"->7,"about"->8,"spark"->
9,"Streaming"->10)
wordstonumbers: scala.collection.immutable.Map[String,Int] = Map(number -> 5, is -> 3, This -> 2, Streaming -> 10, it -> 7, T
wenty -> 6, spark -> 9, Hi -> 1, Assignment -> 4, about -> 8)

scala> val wordstonumbersbroadcast = sc.broadcast(wordstonumbers)
wordstonumbersbroadcast: org.apache.spark.broadcast.Broadcast[scala.collection.immutable.Map[String,Int]] = Broadcast(0)
```

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.

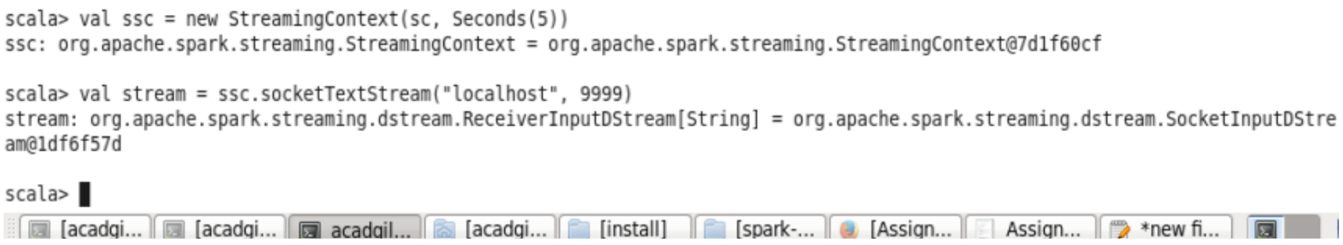


```
scala> def lineWordNumberSum(line:String):Int = {
     | var sum:Int = 0
     | var words = line.split(" ")
     | for (word <- words) sum += wordstonumbersbroadcast.value.get(word).getOrElse(0)
     | sum
     | }
lineWordNumberSum: (line: String)Int

scala> ▌
```

Step 5:

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)



```
scala> val ssc = new StreamingContext(sc, Seconds(5))
ssc: org.apache.spark.streaming.StreamingContext = org.apache.spark.streaming.StreamingContext@7d1f60cf

scala> val stream = ssc.socketTextStream("localhost", 9999)
stream: org.apache.spark.streaming.dstream.ReceiverInputDStream[String] = org.apache.spark.streaming.dstream.SocketInputDStre
am@1df6f57d

scala> 
```

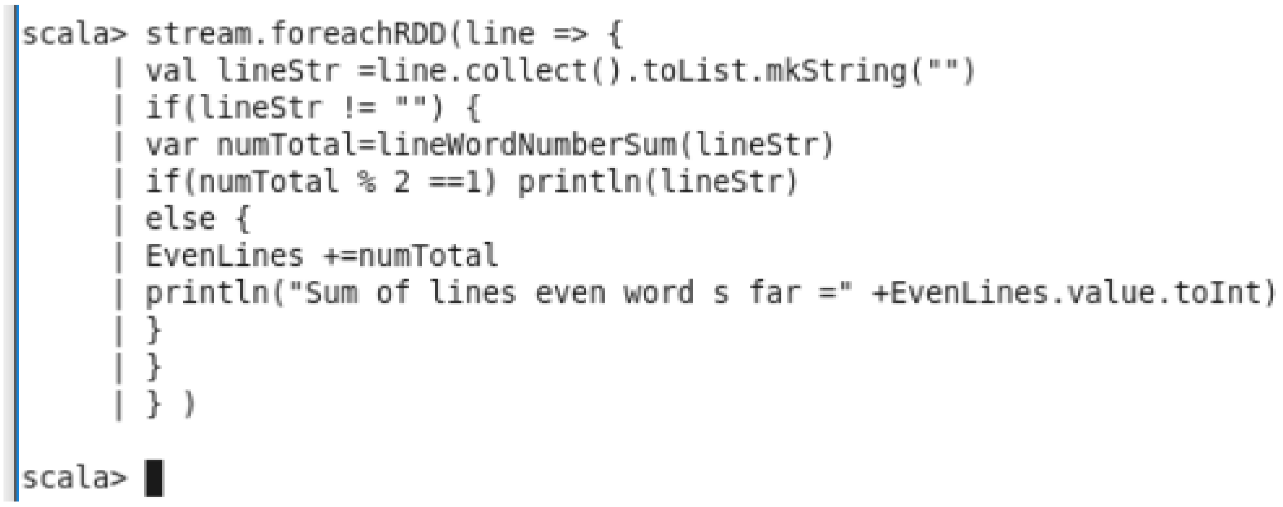
[acadgi... | [acadgi... | acadgil... | [acadgi... | [install] | [spark-... | [Assign... | Assign... | *new fi...

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:



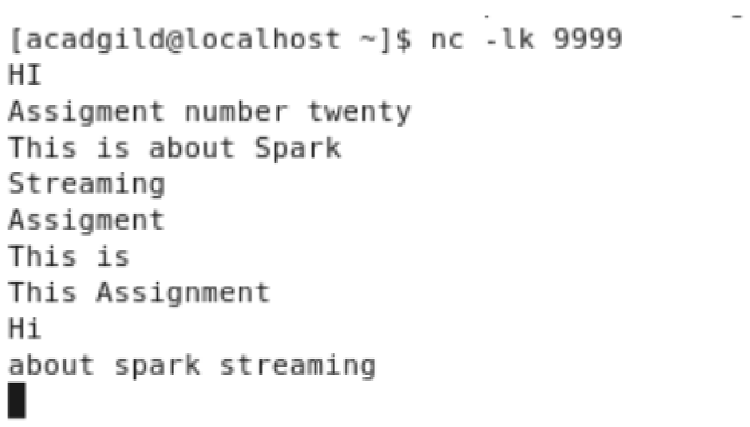
```
scala> stream.foreachRDD(line => {
     | val lineStr =line.collect().toList.mkString("")
     | if(lineStr != "") {
     | var numTotal=lineWordNumberSum(lineStr)
     | if(numTotal % 2 ==1) println(lineStr)
     | else {
     | EvenLines +=numTotal
     | println("Sum of lines even word s far =" +EvenLines.value.toInt)
     | }
     | }
     | } )

scala> 
```

Step 7: Spark Streaming

Now, start the streams and wait until it terminates

- ssc.start()
- ssc.awaitTermination()

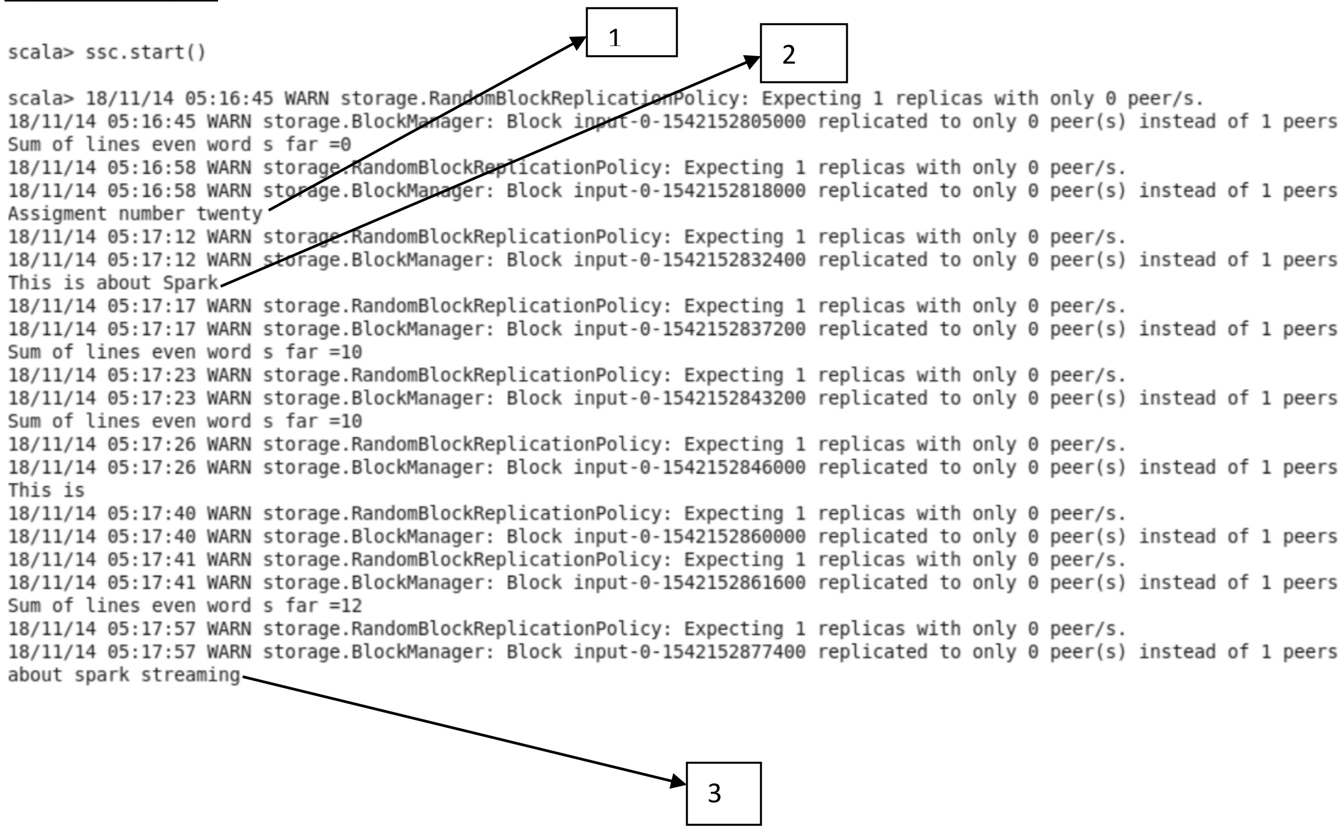
Start netcat in the linux terminal, provide the texts which we determined in the map.



```
[acadgild@localhost ~]$ nc -lk 9999
HI
Assigment number twenty
This is about Spark
Streaming
Assigment
This is
This Assignment
Hi
about spark streaming
█
```

1. Same as, "Assignment number twenty" which has value of 15 which is again ODD and hence it is displayed.
2. Same for "about spark streaming"
3. For lines with even numbered word number, the summation done so far will be displayed. Please see the below screen shot,

Expected Output:



```
scala> ssc.start()

scala> 18/11/14 05:16:45 WARN storage.RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
18/11/14 05:16:45 WARN storage.BlockManager: Block input-0-1542152805000 replicated to only 0 peer(s) instead of 1 peers
Sum of lines even word s far =0
18/11/14 05:16:58 WARN storage.RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
18/11/14 05:16:58 WARN storage.BlockManager: Block input-0-1542152818000 replicated to only 0 peer(s) instead of 1 peers
Assigment number twenty
18/11/14 05:17:12 WARN storage.RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
18/11/14 05:17:12 WARN storage.BlockManager: Block input-0-1542152832400 replicated to only 0 peer(s) instead of 1 peers
This is about Spark
18/11/14 05:17:17 WARN storage.RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
18/11/14 05:17:17 WARN storage.BlockManager: Block input-0-1542152837200 replicated to only 0 peer(s) instead of 1 peers
Sum of lines even word s far =10
18/11/14 05:17:23 WARN storage.RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
18/11/14 05:17:23 WARN storage.BlockManager: Block input-0-1542152843200 replicated to only 0 peer(s) instead of 1 peers
Sum of lines even word s far =10
18/11/14 05:17:26 WARN storage.RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
18/11/14 05:17:26 WARN storage.BlockManager: Block input-0-1542152846000 replicated to only 0 peer(s) instead of 1 peers
This is
18/11/14 05:17:40 WARN storage.RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
18/11/14 05:17:40 WARN storage.BlockManager: Block input-0-1542152860000 replicated to only 0 peer(s) instead of 1 peers
18/11/14 05:17:41 WARN storage.RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
18/11/14 05:17:41 WARN storage.BlockManager: Block input-0-1542152861600 replicated to only 0 peer(s) instead of 1 peers
Sum of lines even word s far =12
18/11/14 05:17:57 WARN storage.RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
18/11/14 05:17:57 WARN storage.BlockManager: Block input-0-1542152877400 replicated to only 0 peer(s) instead of 1 peers
about spark streaming
```

Task 2

Read two streams

1. List of strings input by user

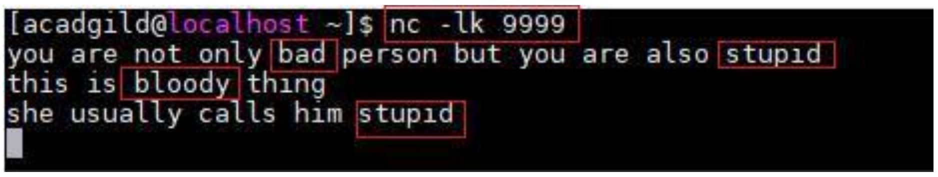
2. Real-time set of offensive words

Find the word count of the offensive words inputted by the user as per the real-time set of offensive words

Solution: In the spark application, we have set of words that we considered as offensive words "idiot", "fool", "bad", "nonsense", "shit", "damn", "stupid", "dash", "bloody", "rascal"

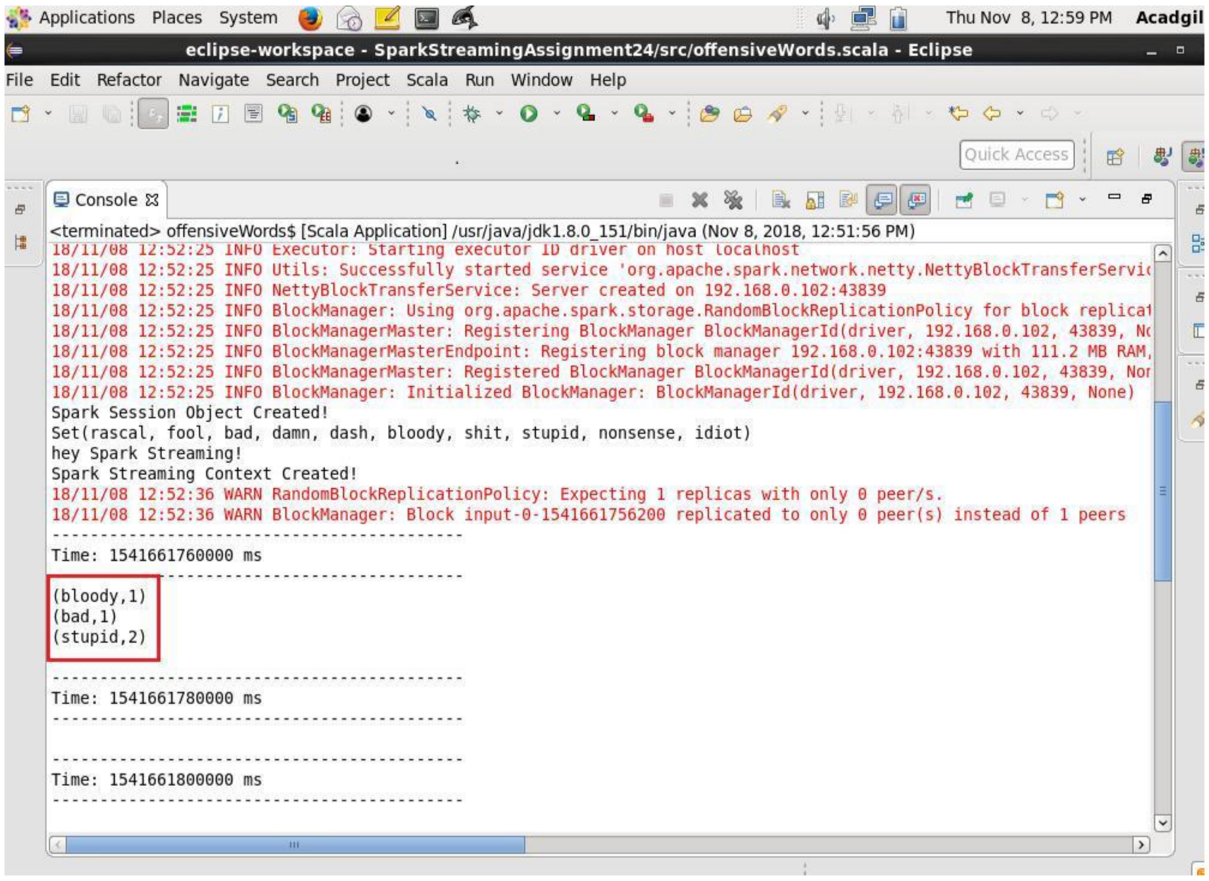
First we started "netcat" in terminal using the command "nc -lk 9999".

We are running the spark application and adding some string with numbers on the shell as shown below:



```
[acadgild@localhost ~]$ nc -lk 9999
you are not only bad person but you are also stupid
this is bloody thing
she usually calls him stupid
```

In below screenshot, we could see that spark application has counted the number of offensive words occur based on the input string provided by the user.



Applications Places System
Thu Nov 8, 12:59 PM Acadgil
eclipse-workspace - SparkStreamingAssignment24/src/offensiveWords.scala - Eclipse
File Edit Refactor Navigate Search Project Scala Run Window Help
Quick Access
Console

```
<terminated> offensiveWords$ [Scala Application] /usr/java/jdk1.8.0_151/bin/java (Nov 8, 2018, 12:51:56 PM)
18/11/08 12:52:25 INFO Executor: Starting executor ID driver on host localhost
18/11/08 12:52:25 INFO Utils: Successfully started service 'org.apache.spark.network.netty.NettyBlockTransferServic
18/11/08 12:52:25 INFO NettyBlockTransferService: Server created on 192.168.0.102:43839
18/11/08 12:52:25 INFO BlockManager: Using org.apache.spark.storage.RandomBlockReplicationPolicy for block replicat
18/11/08 12:52:25 INFO BlockManagerMaster: Registering BlockManager BlockManagerId(driver, 192.168.0.102, 43839, No
18/11/08 12:52:25 INFO BlockManagerMasterEndpoint: Registering block manager 192.168.0.102:43839 with 111.2 MB RAM,
18/11/08 12:52:25 INFO BlockManagerMaster: Registered BlockManager BlockManagerId(driver, 192.168.0.102, 43839, Nor
18/11/08 12:52:25 INFO BlockManager: Initialized BlockManager: BlockManagerId(driver, 192.168.0.102, 43839, None)
Spark Session Object Created!
Set(rascal, fool, bad, damn, dash, bloody, shit, stupid, nonsense, idiot)
hey Spark Streaming!
Spark Streaming Context Created!
18/11/08 12:52:36 WARN RandomBlockReplicationPolicy: Expecting 1 replicas with only 0 peer/s.
18/11/08 12:52:36 WARN BlockManager: Block input-0-1541661756200 replicated to only 0 peer(s) instead of 1 peers
-------------------------------------------
Time: 1541661760000 ms
-------------------------------------------
(bloody,1)
(bad,1)
(stupid,2)

-------------------------------------------
Time: 1541661780000 ms
-------------------------------------------

-------------------------------------------
Time: 1541661800000 ms
-------------------------------------------
```

Scala Code:

```
import org.apache.spark.{SparkConf, SparkContext}
import org.apache.spark.streaming.{Seconds, StreamingContext} import
scala.collection.mutable.ArrayBuffer
object Assignment_24_Streaming_Part2 {
//ArrayBuffer to store list of offensive words in memory
val wordList: ArrayBuffer[String] = ArrayBuffer.empty[String]
def main(args: Array[String]) {
println("hey Scala, Streaming Offensive Words!")
//Let us create a spark session object
val conf = new
SparkConf().setMaster("local[2]").setAppName("SparkSteamingOffensiveWords")
val sc = new SparkContext(conf)
sc.setLogLevel("WARN")
println("Spark Session Object Created!")
val OffensiveWordList: Set[String] =
Set("idiot","fool","bad","nonsense","shit","damn","stupid","dash","bloody","rascal
"
)
println(s"$OffensiveWordList")
//Create the context with a 30 second batch size println("hey Spark Streaming!")
val ssc = new StreamingContext(sc, Seconds(20))
println("Spark Streaming Context Created!")
val lines = ssc.socketTextStream("localhost", 9999)
val wordCounts = lines.flatMap(_.split(" "))
.map(x => x)
.filter(x => OffensiveWordList.contains(x.toLowerCase))
.map(x => (x, 1))
.reduceByKey(_ + _)
wordCounts.print()
ssc.start()
ssc.awaitTermination()
}
}
```