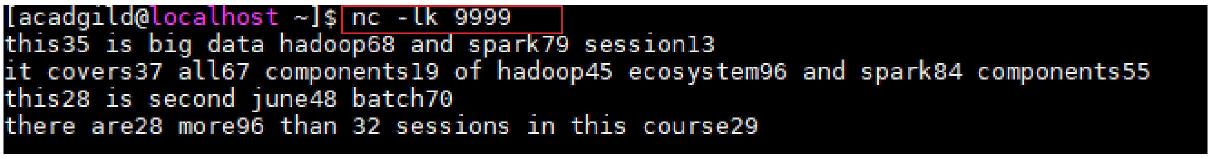
Assignment 24.1
Spark Streaming

<u>Task 1:</u>

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.

First we have 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
this35 is big data hadoop68 and spark79 session13
it covers37 all67 components19 of hadoop45 ecosystem96 and spark84 components55
this28 is second june48 batch70
there are28 more96 than 32 sessions in this course29
```

In above strings, you could see that out of total 4 lines, there are 3 lines in which sum of numbers is odd. Those lines(rows) and their respective sum of numbers are :

it covers37 all67 components19 of hadoop45 ecosystem96 and spark84 components55 **(2nd row,sum = 3+7+6+7+1+9+4+5+9+6+8+4+5+5 = 79 (odd))**

this28 is second june48 batch70 **(3rd row,sum : 29 (odd))**

there are28 more96 than 32 sessions in this course29 **(4th row,sum : 41 (odd))**

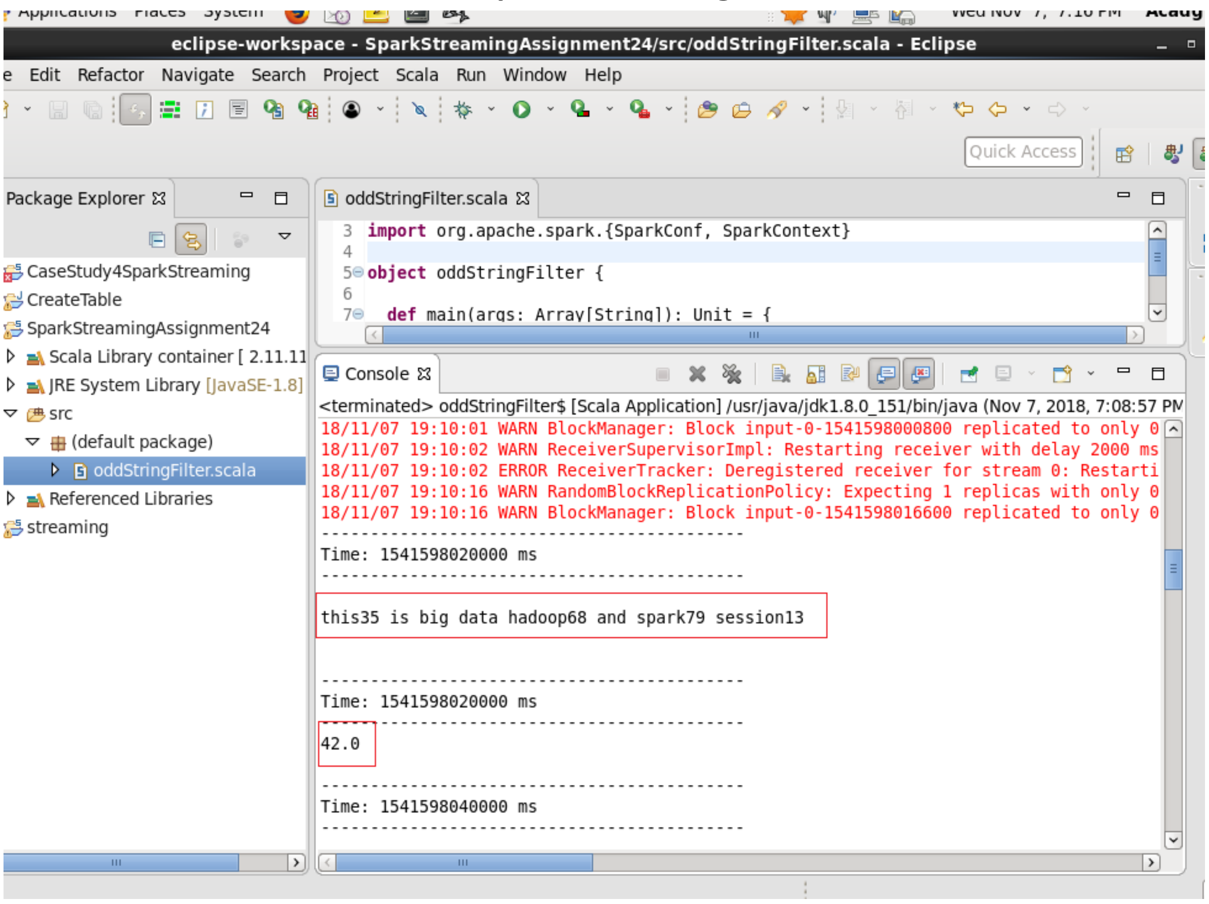
As we have considered only rows in which sum of numbers is even, we have taken only 1 row:

Then we are taking remaining rows (sum of numbers is even), hence out of 4, we are getting only 1 row whose sum of numbers is even.

this35 is big data hadoop68 and spark79 session13 **(1st row,sum : 42 (even))**

Hence we are getting below output:

Assignment 24.1
Spark Streaming



```
3 import org.apache.spark.{SparkConf, SparkContext}
4
5 object oddStringFilter {
6
7   def main(args: Array[String]): Unit = {
```

```
<terminated> oddStringFilter$ [Scala Application] /usr/java/jdk1.8.0_151/bin/java (Nov 7, 2018, 7:08:57 PM
18/11/07 19:10:01 WARN BlockManager: Block input-0-1541598000800 replicated to only 0
18/11/07 19:10:02 WARN ReceiverSupervisorImpl: Restarting receiver with delay 2000 ms
18/11/07 19:10:02 ERROR ReceiverTracker: Deregistered receiver for stream 0: Restarti
18/11/07 19:10:16 WARN RandomBlockReplicationPolicy: Expecting 1 replicas with only 0
18/11/07 19:10:16 WARN BlockManager: Block input-0-1541598016600 replicated to only 0
-------------------------------------------
Time: 1541598020000 ms
-------------------------------------------

this35 is big data hadoop68 and spark79 session13


-------------------------------------------
Time: 1541598020000 ms
-------------------------------------------
42.0

-------------------------------------------
Time: 1541598040000 ms
-------------------------------------------
```

In screenshot above, we are able to see that lines containing sum of odd numbers are filtered, only line in which sum of numbers as even number is displayed, and sum of the number is displayed in the next line (42).

<u>Scala Code :</u>

<u>Assignment24_Streaming_Part1</u>

```
import org.apache.spark.streaming.{Seconds, StreamingContext}
import org.apache.spark.{SparkConf, SparkContext}

object Assignment24_Streaming_Part1 {

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

      val line = input.split("")
      var number : Double = 0.0
      for (x <- line)
      {
        try{
          val value = x.toDouble
          number = number + value
        }
        catch
          {
            case ex : Exception => {}
          }
      }
      return number
    }
```

Assignment 24.1
Spark Streaming

```
    println("hey, Spark Streaming Session powered by Scala")

    val conf = new SparkConf().setMaster("local[2]").setAppName("EvenLines")

    val sc = new SparkContext(conf)

    sc.setLogLevel("WARN")

    println("Spark Context Created")

    val ssc = new StreamingContext(sc, Seconds(20))
    println("Spark Streaming Context Created ")

    val lines = ssc.socketTextStream("localhost", 9999)
    val lines_filter = lines.filter(x => Get_Lines_Sum(x)%2 == 0)
    val lines_sum = lines_filter.map(x => Get_Lines_Sum(x))

    println("Lines with even sum:")
    lines_filter.print()

    println("Sum of the numbers in even lines:")
    lines_sum.reduce(_+_).print()

    ssc.start()
    ssc.awaitTermination()
  }

}
```

Assignment 24.1
Spark Streaming

<u>Task 2:</u>

Read two streams

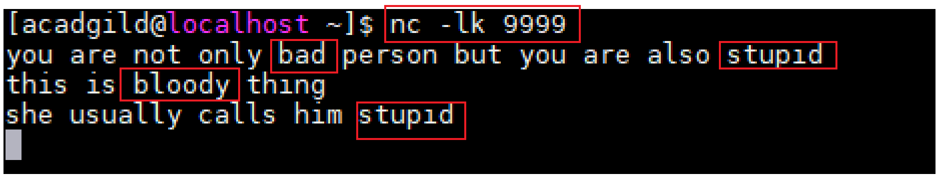
1. List of strings input by user

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.

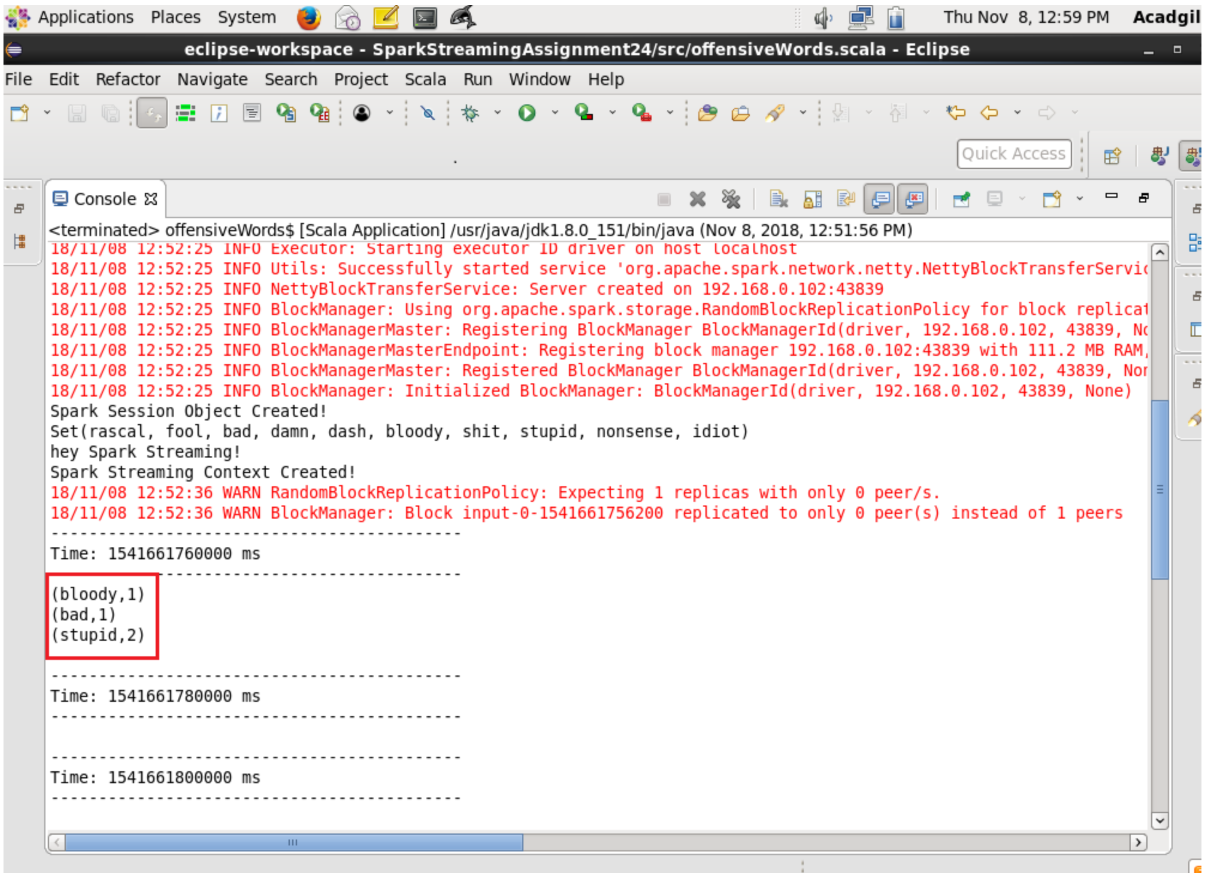
In the spark application, we have a set of words that we considered as offensive words "idiot", "fool", "bad", "nonsense", "shit", "damn", "stupid", "dash", "bloody", "rascal"

First we have 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
-------------------------------------------
```

Assignment 24.1
Spark Streaming

Scala Code :

Assignment_24_Streaming_Part2 (Offensive words) :

```
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()
  }
}
```