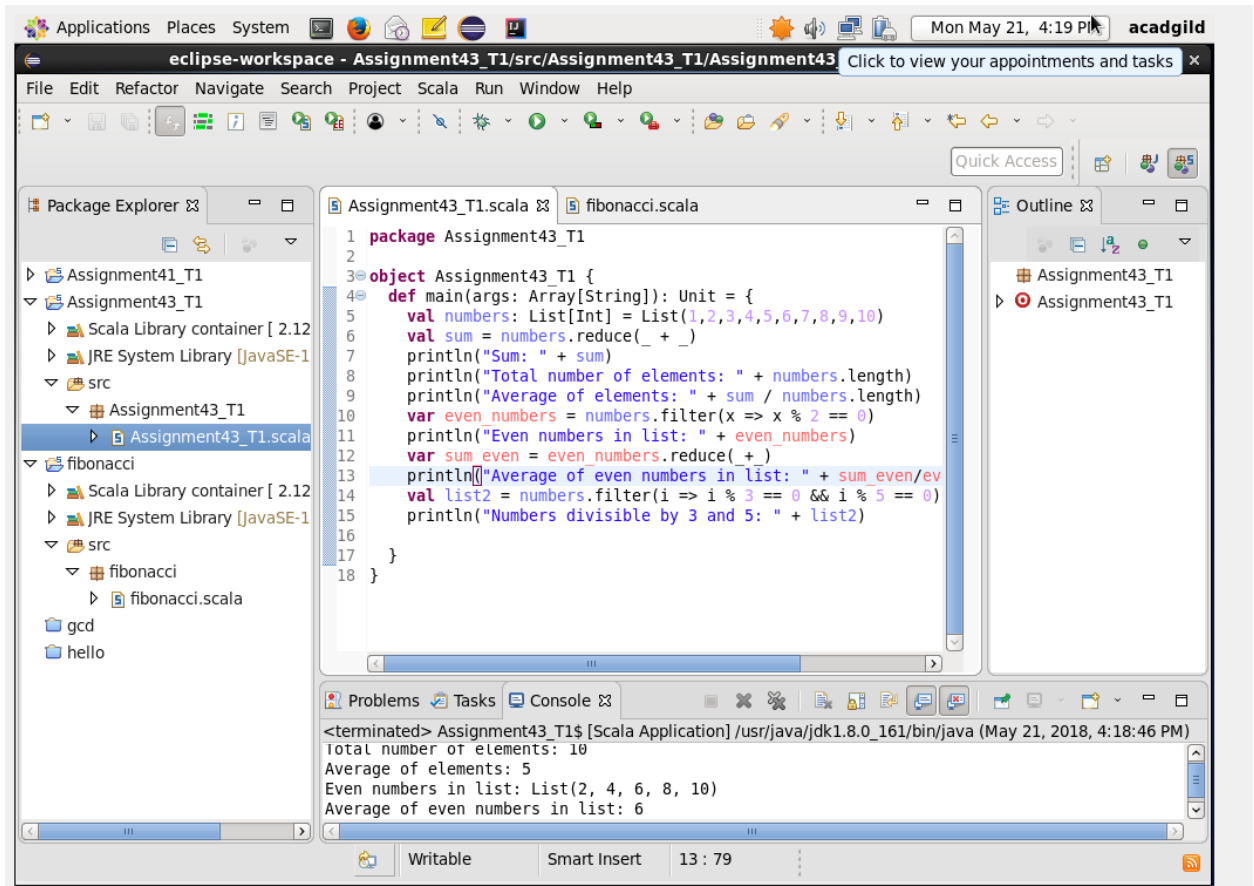


## ASSIGNMENT 43.1

### TASK 1:

In order to find the sum of even and sum of all numbers in the list, use the reduce method, to sum up the elements. To find the length of the list, use the length attribute. Use filter to find even numbers and numbers divisible by 3 and by 5.



```
1 package Assignment43_T1
2
3 object Assignment43_T1 {
4   def main(args: Array[String]): Unit = {
5     val numbers: List[Int] = List(1,2,3,4,5,6,7,8,9,10)
6     val sum = numbers.reduce(_ + _)
7     println("Sum: " + sum)
8     println("Total number of elements: " + numbers.length)
9     println("Average of elements: " + sum / numbers.length)
10    var even_numbers = numbers.filter(x => x % 2 == 0)
11    println("Even numbers in list: " + even_numbers)
12    var sum_even = even_numbers.reduce(_ + _)
13    println("Average of even numbers in list: " + sum_even/ev
14    val list2 = numbers.filter(i => i % 3 == 0 && i % 5 == 0)
15    println("Numbers divisible by 3 and 5: " + list2)
16  }
17 }
18 }
```

<terminated> Assignment43\_T1\$ [Scala Application] /usr/java/jdk1.8.0\_161/bin/java (May 21, 2018, 4:18:46 PM)  
Total number of elements: 10  
Average of elements: 5  
Even numbers in list: List(2, 4, 6, 8, 10)  
Average of even numbers in list: 6

### TASK 2:

#### 1) Drawbacks of MapReduce:

- It is not suitable for real time processing.
- It is not always very easy to implement each task using this paradigm.
- Not suitable when intermediate processes need to talk to each other.
- Not suitable when a lot of data need to be shuffled.
- Not suitable to handle streaming data.
- Not suitable when the desired result is possible with standalone system.
- Not suitable to process graphs.
- Not suitable for iterative processing.
- Not suitable for joining two large data sets with complex conditions.
- Not suitable for stateful operations.

- 2) RDD: Stands for Resilient Distributed Datasets – a basic abstraction of spark which is immutable and logically partitioned to apply parallel operations on them. Some features of RDDs are:
- a) In-memory computation: They have a provision for in-memory computation. It stores intermediate results in distributed RAM memory instead of the hard disk.
  - b) Lazy evaluations: All transformations do not compute their results right away – they remember the transformations and compute them only when an action requires a result for the driver program.
  - c) Fault Tolerance: They rebuild lost data on failure using lineage – each RDD remembers how it was created from other datasets.
  - d) Immutability: It is safe to share across processes and consistency is maintained across computations.
  - e) Partitioning: A partition is one logical division of data which is mutable.
  - f) Persistence: Storage strategy (in-memory or hard disk) can be chosen by the user.
  - g) Coarse-grained operations: Applied to all elements in the dataset through maps, filters, group by operation.
  - h) Location-stickiness: RDDs are capable of defining placement preference to compute partitions. Placement preference refers to the information about the location of the RDD.

3) Spark Operations:

a) Transformations:

RDD Transformations are functions that take an RDD as the input and produce one or many RDDs as the output. Eg: map(), reduce(), filter(). They do not change the input dataset.

1.1) Narrow transformations:

It is the result of map, filter and such that the data is from a single partition only. Only a limited subset of partitions are used to calculate the result. Spark groups narrow transformations as a stage known as pipelining.

1.2) Wide transformations:


It is the result of groupByKey() and reduceByKey() like functions. Data required to compute records in a single partition may live in many partitions of the parent RDD. Also called shuffle transformations.

b) Actions:

Final result of RDD computations – RDD operations that produce non-RDD values. First(), take(), reduce(), collect(), count() are some of the actions in spark.


Task 3:

Use the scala.io.Source package. Read the lines using Source.fromFile.getLines method. Count number of lines using length attribute. Collect contents of file using Source.fromFile.getLines.mkString method, split using delimiter(space and dash) and use length to count number of words.

Applications Places System  Mon May 21, 4:47 PM acadgild

eclipse-workspace - Assignment43\_T3/src/Assignment43\_T3/Assignment43\_T3.scala [Change account settings and status](#)

File Edit Refactor Navigate Search Project Scala Run Window Help

Quick Access 

Package Explorer

- Assignment41\_T1
- Assignment43\_T1
  - Scala Library container [2.10.5]
  - JRE System Library [JVM 1.8.0\_161]
  - src
    - Assignment43\_T1
    - Assignment43\_T1.scala
- Assignment43\_T3
  - Scala Library container [2.10.5]
  - JRE System Library [JVM 1.8.0\_161]
  - src
    - Assignment43\_T3
    - Assignment43\_T3.scala
- fibonacci
  - Scala Library container [2.10.5]
  - JRE System Library [JVM 1.8.0\_161]
  - src
    - fibonacci
    - fibonacci.scala
- gcd
- hello

Assignment43\_T3.scala

```
1 package Assignment43_T3
2
3 import scala.io.Source;
4
5 object Assignment43_T3 {
6   def main(args: Array[String]) {
7     val num_lines = Source.fromFile("/home/acadgild/sample_text.txt").getLines().count()
8     println("Number of lines in the file: " + num_lines)
9     val contents = Source.fromFile("/home/acadgild/sample_text.txt").getLines().mkString("\n")
10    println("Number of words: " + contents.split(" ").length)
11    val contents_dashed = Source.fromFile("/home/acadgild/sample_text.txt").getLines().mkString("\n\n")
12    println("Number of words in dashed doc: " + contents_dashed.split(" ").length)
13  }
14
15 }
```

Outline

- Assignment43\_T3
  - import declarations
  - Assignment43\_T3

Problems Tasks Console

<terminated> Assignment43\_T3\$ [Scala Application] /usr/java/jdk1.8.0\_161/bin/java (May 21, 2018, 4:47:46 PM)  
Number of lines in the file: 3  
Number of words: 22  
Number of words in dashed doc: 22

Writable Smart Insert 9:16

eclipse-worksp... acadgild eclipse-worksp... Assignment43... src Assignment43\_...