Session 16

Assignment 1

|  |  |
| --- | --- |
| **Prepared For:** | AcadGild |
|  |  |
| **Document Approval:** | **AcadGild** |
|  |  |
|  |  |
|  |  |
|  |  |
| **Project Title:** | Session 16 – Assignment 1 |
|  |  |
| **Prepared By:** | Duncan Burgess |
|  |  |
|  | dburgess@duncb.com |
|  |  |
| **Primary Engineer:** | Duncan Burgess |
|  |  |
| **Document Reference:** | **Session 16 – Assignment 1** |
|  |  |
| **Start Date:** | 15/10/2017 |
|  |  |
|  |  |



# 

# Contents

[Contents 2](#_Toc495599248)

[Change History 3](#_Toc495599249)

[1. Problem Statement 4](#_Toc495599250)

[2. Solutions 4](#_Toc495599251)

[2.1. Solution 1 4](#_Toc495599252)

[2.2. Solution 2 with loop 5](#_Toc495599253)

# Change History

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Document Revision** | **Date** | **Authored By** | **Authorised By** | **Sections Affected** | **Reason for Change** |
| Rev 01 | 16/10/2017 | Duncan Burgess |  | All | Initial release. |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# Problem Statement

Given a list of numbers - List[Int] (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

* find the sum of all numbers
* find the total elements in the list
* calculate the average of the numbers in the list
* find the sum of all the even numbers in the list
* find the total number of elements in the list divisible by both 5 and 3

# Solution

This was carried out by running commands within the spark shell the following shows the commands run and the associated results.

*Welcome to*

**\_\_\_\_ \_\_**

**/ \_\_/\_\_ \_\_\_ \_\_\_\_\_/ /\_\_**

**\_\ \/ \_ \/ \_ `/ \_\_/ '\_/**

**/\_\_\_/ .\_\_/\\_,\_/\_/ /\_/\\_\ version 2.2.0**

**/\_/**

*Using Scala version 2.11.8 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0\_144)*

*Type in expressions to have them evaluated.*

*Type :help for more information.*

*scala> val num = List[Int](1, 2, 3, 4, 5, 6, 7, 8, 9, 10)*

*num: List[Int] = List(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)*

*scala> val numRDD = sc.parallelize(num)*

*numRDD: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[0] at parallelize*

*at <console>:26*

*scala> val sumRDD = numRDD.reduce{(x, y) => x + y}*

*[Stage 0:> (0 + 0) / 2]*

*sumRDD: Int = 55*

*scala> val cntRDD = numRDD.count()*

*cntRDD: Long = 10*

*scala> val avgRDD: Double = sumRDD.toDouble / cntRDD*

*avgRDD: Double = 5.5*

*scala> val EvnRDD = numRDD.filter(s => ((s%2)==0)).collect*

*EvnRDD: Array[Int] = Array(2, 4, 6, 8, 10)*

*scala> val sumevRDD = EvnRDD.reduce{(x, y) => x + y}*

*sumevRDD: Int = 30*

*scala> val n53RDD = numRDD.filter(s => ((s%5)==0) && (s%3==0)).collect*

*n53RDD: Array[Int] = Array()*

*scala> val totDiv53 = numRDD.filter(s => ((s%5)==0) && (s%3==0)).count()*

*totDiv53: Long = 0*

*scala>*