## 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:**

- Creating a list of numbers

```
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>:29

scala>
```

- find the sum of all numbers

```
scala> val sumRDD = numRDD.reduce{(x,y)=>x+y}
sumRDD: Int = 55
```

- find the total elements in the list

```
scala> val countRDD = numRDD.count()
countRDD: Long = 10
```

- calculate the average of the numbers in the list

```
scala> val averageRDD :Double  = sumRDD.toDouble / countRDD
averageRDD: Double = 5.5

scala>
```

- find the sum of all the even numbers in the list

```
scala> val evenRDD = numRDD.filter(s=>s%2==0).collect
evenRDD: Array[Int] = Array(2, 4, 6, 8, 10)

scala> evenRDD.foreach(println)
2
4
6
8
10

scala> val sumEvenRDD = evenRDD.reduce{(x,y)=>x+y}
sumEvenRDD: Int = 30

scala>
```

- find the total number of elements in the list divisible by both 5 and 3

```
scala> val elementsDiv53RDD = numRDD.filter(s=>(s%5) == 0 && (s%3) == 0).collect
elementsDiv53RDD: Array[Int] = Array()

scala> val elementsDiv53RDD = numRDD.filter(s=>(s%5) == 0 && (s%3) == 0).count
elementsDiv53RDD: Long = 0

scala> println(elementsDiv53RDD)
0

scala>
```