ASSIGNMENT 14.1

Problem Statement:

Create a calculator to work with rational numbers.

Requirements:

- It should provide capability to add, subtract, divide and multiply rational numbers
- Create a method to compute GCD (this will come in handy during operations on rational)

Add option to work with whole numbers which are also rational numbers i.e. (n/1)

- achieve the above using auxiliary constructors

- enable method overloading to enable each function to work with numbers and rational.

Solution:

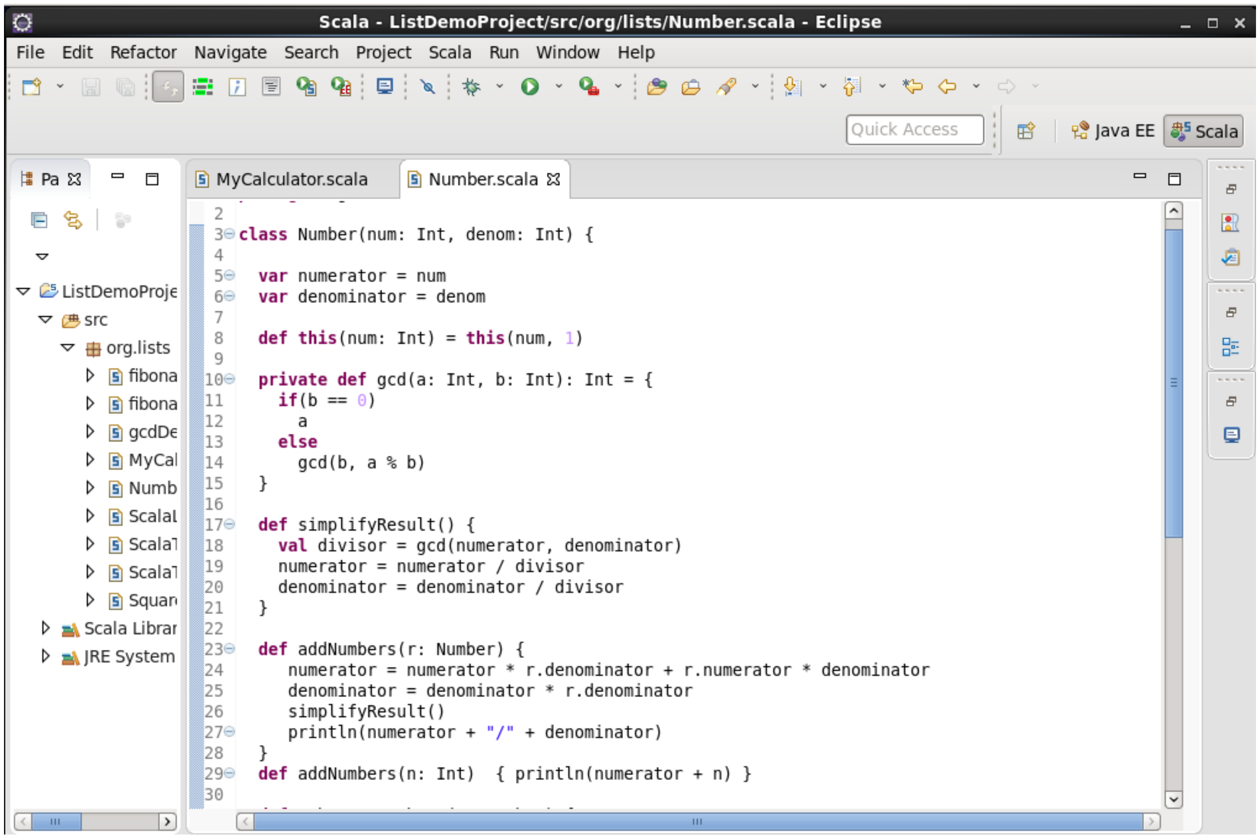
Let’s understand first what rational numbers are.

A **Rational number** is a number that can be in the form **x/y**, where x and y are integers and y is not equal to zero. For example, 3/5 is a rational number where 3 is called ‘numerator’ and 5 is called ‘denominator’.

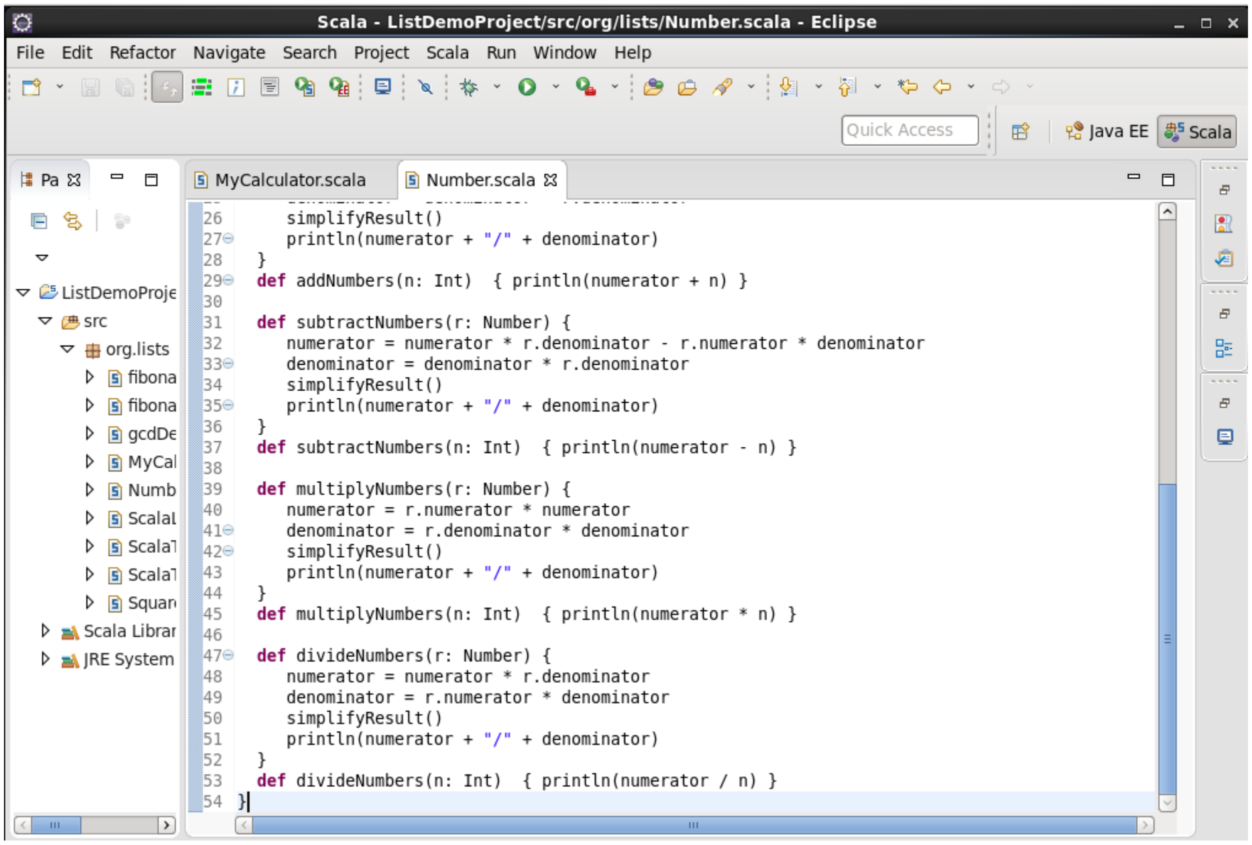
Here is the Scala class code I have written to perform arithmetic operations on rational numbers:

Number.scala

```
class Number(num: Int, denom: Int) {          // Primary constructor that initializes two arguments

  var numerator = num
  var denominator = denom

 def this(num: Int) = this(num, 1)            // An auxiliary constructor that initializes numerator
                                              with value passed on objection creation and
                                              denominator with fixed value ‘1’

  def this() = this(0, 0)                     // Another auxiliary constructor that initializes both
                                              numerator and denominator with value ‘0’

  private def gcd(a: Int, b: Int): Int = {    // A recursive function that returns GCD (Greatest
      if(b == 0)          a                   Common Divisor) of two numbers given
      else                gcd(b, a % b)
  }
  def simplifyResult() {                          // This function is used to reduce fraction
    val divisor = gcd(numerator, denominator)     // result of any arithmetic operation
    numerator = numerator / divisor
    denominator = denominator / divisor
  }

  def addNumbers(r: Number) {                     // routine for addition of two rational numbers
     numerator = numerator * r.denominator + r.numerator * denominator
     denominator = denominator * r.denominator
     simplifyResult()
     println(numerator + "/" + denominator)
  }
  def addNumbers(n: Int)  { println(numerator + n) }      //overloading addNumbers() method by
                                                          // changing datatype of the argument

  def subtractNumbers(r: Number) {             // routine for subtraction of two rational numbers
     numerator = numerator * r.denominator - r.numerator * denominator
     denominator = denominator * r.denominator
     simplifyResult()
     println(numerator + "/" + denominator)
  }
  def subtractNumbers(n: Int)  { println(numerator - n) } // overloading subtractNumbers() method
                                                          // by changing datatype of the argument

  def multiplyNumbers(r: Number) {          // routine for multiplication of two rational numbers
     numerator = r.numerator * numerator
     denominator = r.denominator * denominator
     simplifyResult()
     println(numerator + "/" + denominator)
  }
  def multiplyNumbers(n: Int)  { println(numerator * n) }//overloading multiplyNumbers() method
                                                          // by changing datatype of the argument

  def divideNumbers(r: Number) {                  // routine for division of two rational numbers
     numerator = numerator * r.denominator
     denominator = r.numerator * denominator
     simplifyResult()
     println(numerator + "/" + denominator)
  }
  def divideNumbers(n: Int)  { println(numerator / n) }     // overloading divideNumbers() method
}                                                         // by changing datatype of the argument
```



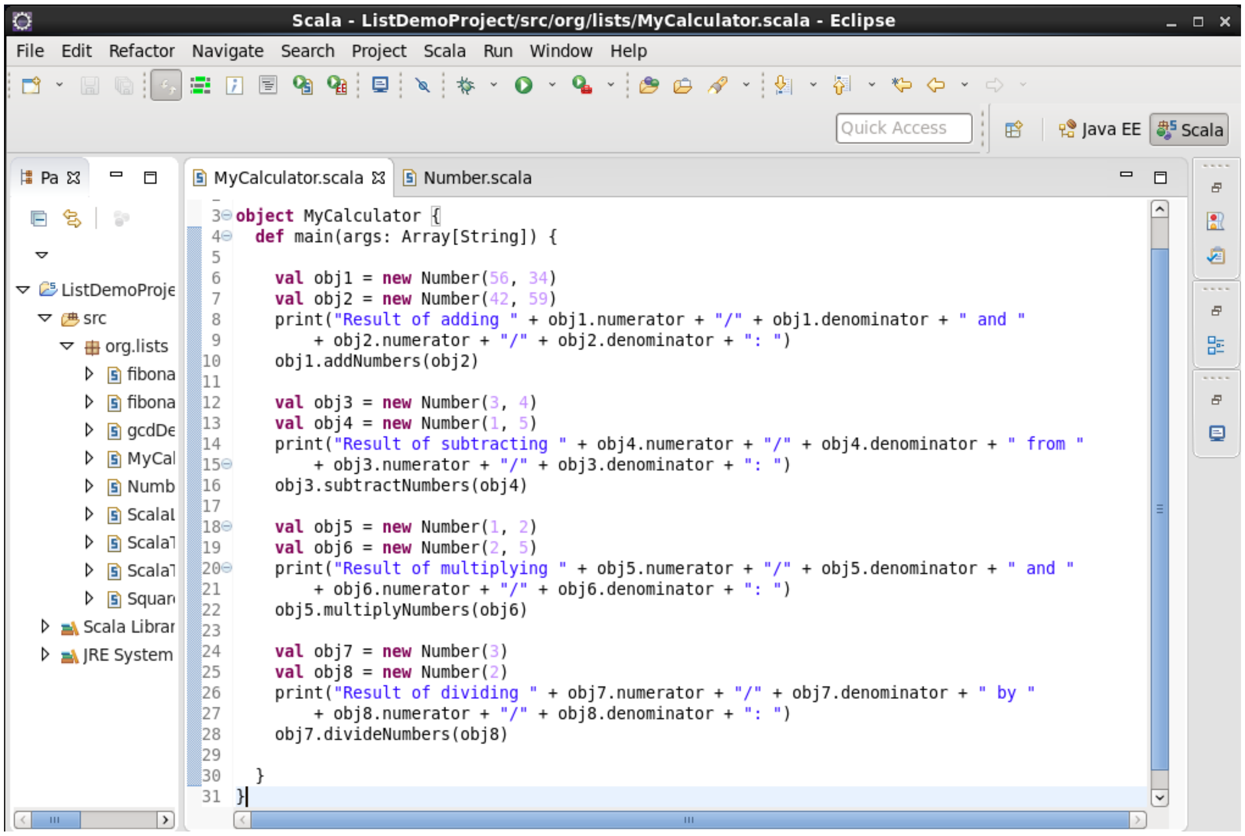
```

class Number(num: Int, denom: Int) {

  var numerator = num
  var denominator = denom

  def this(num: Int) = this(num, 1)

  private def gcd(a: Int, b: Int): Int = {
    if(b == 0)
      a
    else
      gcd(b, a % b)
  }

  def simplifyResult() {
    val divisor = gcd(numerator, denominator)
    numerator = numerator / divisor
    denominator = denominator / divisor
  }

  def addNumbers(r: Number) {
     numerator = numerator * r.denominator + r.numerator * denominator
     denominator = denominator * r.denominator
     simplifyResult()
     println(numerator + "/" + denominator)
  }
  def addNumbers(n: Int)  { println(numerator + n) }

```



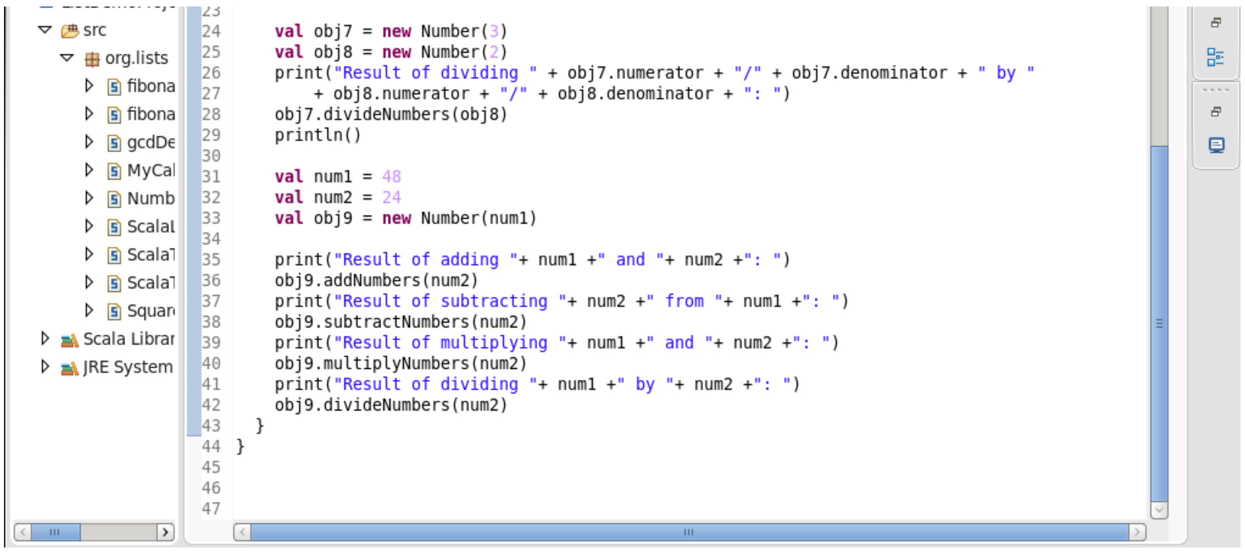
```
     simplifyResult()
     println(numerator + "/" + denominator)
  }
  def addNumbers(n: Int)  { println(numerator + n) }

  def subtractNumbers(r: Number) {
     numerator = numerator * r.denominator - r.numerator * denominator
     denominator = denominator * r.denominator
     simplifyResult()
     println(numerator + "/" + denominator)
  }
  def subtractNumbers(n: Int)  { println(numerator - n) }

  def multiplyNumbers(r: Number) {
     numerator = r.numerator * numerator
     denominator = r.denominator * denominator
     simplifyResult()
     println(numerator + "/" + denominator)
  }
  def multiplyNumbers(n: Int)  { println(numerator * n) }

  def divideNumbers(r: Number) {
     numerator = numerator * r.denominator
     denominator = r.numerator * denominator
     simplifyResult()
     println(numerator + "/" + denominator)
  }
  def divideNumbers(n: Int)  { println(numerator / n) }
}
```

Main class: Mycalculator.scala

```
object MyCalculator {
  def main(args: Array[String]) {

    val obj1 = new Number(56, 34)              // Input objects with rational numbers for addition
    val obj2 = new Number(42, 59)
    print("Result of adding " + obj1.numerator + "/" + obj1.denominator + " and "
      + obj2.numerator + "/" + obj2.denominator + ": ")
    obj1.addNumbers(obj2)                                  // addNumbers() method invocation

    val obj3 = new Number(3, 4)               // Input objects with rational numbers for subtraction
    val obj4 = new Number(1, 5)
    print("Result of subtracting " + obj4.numerator + "/" + obj4.denominator + " from "
      + obj3.numerator + "/" + obj3.denominator + ": ")
    obj3.subtractNumbers(obj4)                           // subtractNumbers() method invocation

    val obj5 = new Number(1, 2)           // Input objects with rational numbers for multiplication
    val obj6 = new Number(2, 5)
    print("Result of multiplying " + obj5.numerator + "/" + obj5.denominator + " and "
      + obj6.numerator + "/" + obj6.denominator + ": ")
    obj5.multiplyNumbers(obj6)                          // multiplyNumbers() method invocation

    val obj7 = new Number(3)                  // Input objects with rational numbers for division
    val obj8 = new Number(2)                  // which internally calls an auxiliary constructor
    print("Result of dividing " + obj7.numerator + "/" + obj7.denominator + " by "
      + obj8.numerator + "/" + obj8.denominator + ": ")
    obj7.divideNumbers(obj8)                                // divideNumbers() method invocation
    println()

    val num1 = 48                                           // Inputs for whole number arithmetic
    val num2 = 24
    val obj9 = new Number(num1)                             // Invokes the auxiliary constructor

    print("Result of adding "+ num1 +" and "+ num2 +": ")
    obj9.addNumbers(num2)                                   // method call to sum up two integers
    print("Result of subtracting "+ num2 +" from "+ num1 +": ")
    obj9.subtractNumbers(num2)                           // method call for subtraction of  integers
    print("Result of multiplying "+ num1 +" and "+ num2 +": ")
    obj9.multiplyNumbers(num2)                             // method call for multiply two integers
    print("Result of dividing "+ num1 +" by "+ num2 +": ")
    obj9.divideNumbers(num2)                    // method call for division two integers
  }
}
```



```
object MyCalculator {
  def main(args: Array[String]) {

    val obj1 = new Number(56, 34)
    val obj2 = new Number(42, 59)
    print("Result of adding " + obj1.numerator + "/" + obj1.denominator + " and "
        + obj2.numerator + "/" + obj2.denominator + ": ")
    obj1.addNumbers(obj2)

    val obj3 = new Number(3, 4)
    val obj4 = new Number(1, 5)
    print("Result of subtracting " + obj4.numerator + "/" + obj4.denominator + " from "
        + obj3.numerator + "/" + obj3.denominator + ": ")
    obj3.subtractNumbers(obj4)

    val obj5 = new Number(1, 2)
    val obj6 = new Number(2, 5)
    print("Result of multiplying " + obj5.numerator + "/" + obj5.denominator + " and "
        + obj6.numerator + "/" + obj6.denominator + ": ")
    obj5.multiplyNumbers(obj6)

    val obj7 = new Number(3)
    val obj8 = new Number(2)
    print("Result of dividing " + obj7.numerator + "/" + obj7.denominator + " by "
        + obj8.numerator + "/" + obj8.denominator + ": ")
    obj7.divideNumbers(obj8)

  }
}
```



```

    val obj7 = new Number(3)
    val obj8 = new Number(2)
    print("Result of dividing " + obj7.numerator + "/" + obj7.denominator + " by "
        + obj8.numerator + "/" + obj8.denominator + ": ")
    obj7.divideNumbers(obj8)
    println()

    val num1 = 48
    val num2 = 24
    val obj9 = new Number(num1)

    print("Result of adding "+ num1 +" and "+ num2 +": ")
    obj9.addNumbers(num2)
    print("Result of subtracting "+ num2 +" from "+ num1 +": ")
    obj9.subtractNumbers(num2)
    print("Result of multiplying "+ num1 +" and "+ num2 +": ")
    obj9.multiplyNumbers(num2)
    print("Result of dividing "+ num1 +" by "+ num2 +": ")
    obj9.divideNumbers(num2)
  }
}
```

<u>Output:</u>

Result of adding 56/34 and 42/59: **2366/1003**

Result of subtracting 1/5 from 3/4: **11/20**

Result of multiplying 1/2 and 2/5: **1/5**

Result of dividing 3/1 by 2/1: **3/2**

Result of adding 48 and 24: **72**

Result of subtracting 24 from 48: **24**

Result of multiplying 48 and 24: **1152**

Result of dividing 48 by 24: **2**

Computation:

56/34 + 42/59 = (56/34 * 59/59) + (42/59 * 34/34)

= 4732/2006

= **2366/1003** - reduced result with the help of gcd() function (divisor: 2)

3/4 – 1/5 = (3/4 * 5/5) – (1/5 * 4/4)

= 15 – 4/2

= 11/20

1/2 * 2/5 = (1*2) / (2*5)

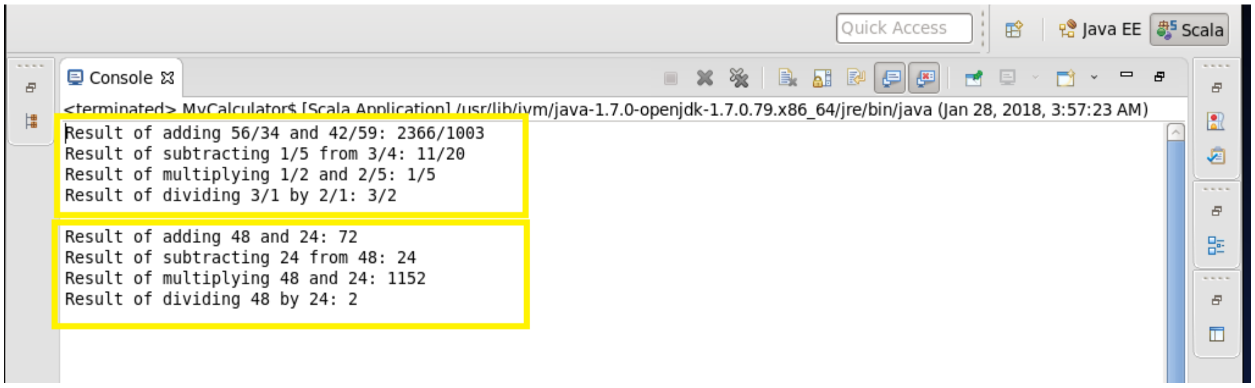
= 2/10

= **1/5** - reduced result with the help of gcd() function (divisor: 2)

3/1 / 2/1 = 3/1 / 1/2

= 3*1 / 2*1

= **3/2**



```
<terminated> MyCalculator$ [Scala Application] /usr/lib/jvm/java-1.7.0-openjdk-1.7.0.79.x86_64/jre/bin/java (Jan 28, 2018, 3:57:23 AM)
Result of adding 56/34 and 42/59: 2366/1003
Result of subtracting 1/5 from 3/4: 11/20
Result of multiplying 1/2 and 2/5: 1/5
Result of dividing 3/1 by 2/1: 3/2

Result of adding 48 and 24: 72
Result of subtracting 24 from 48: 24
Result of multiplying 48 and 24: 1152
Result of dividing 48 by 24: 2
```