Session 16: SCALA BASICS 3 Assignment 1

TASK 1:

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.

EXECUTION:

```
🔳 workspace - ScalaAssignment/src/scala/assignment16/Training.scala - Scala IDE
<u>File Edit Refactor Navigate Search Project Run Window Help</u>
₽ Project Explorer 🛭 🗀
                                     package scala.assignment16
   > 🎏 demoproject
   class Rational(num1:Int, denom1:Int, num2:Int, denom2:Int) {
      > 🚵 Scala Library container [ 2.12.3 ]
      > March JRE System Library [JavaSE-1.8]
                                         // Arbitrary Constructor to handle whole numbers
                                         def this(num1:Int,num2:Int) {
         > # scala.assignment14
                                          this(num1,1,num2,1) // Call to primary constructor
         scala.assignment15
           > 🖺 Task1.scala
           S Task2_1.scala
                                         // Addition of 2 rational numbers
                                         def addRational() : Float = {
           > 🖺 Task2_2.scala
                                          var gcdDenom : Float = gcd(denom1,denom2)
           > Task3.scala
                                          var sumNum : Float = num1 + num2
        scala.assignment16
                                          return (sumNum / gcdDenom)
           > SampleScala.scala
           > S Training.scala
    > 👸 ScalaHello
                                         // Subtraction of 2 rational numbers
                                         def subtractRational(): Float = {
                                          var gcdDenom : Float = gcd denom1,denom2)
                                          var diffNum : Float = num1 - num2
                                           return (diffNum / gcdDenom)
                                         // Multiplication of 2 rational numbers
                                          def multiplyRational() : Float = {
                                          var multiplyNum : Float = num1 * num2
                                           var multiplyDenom: Float = denom1 * denom2
                                           return (multiplyNum/multiplyDenom)
```

```
workspace - ScalaAssignment/src/scala/assignment16/Training.scala - Scala IDE
File Edit Refactor Navigate Search Project Run Window Help

    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □ → □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □
    □</td
₽ Project Explorer 🛭
                                                                               🖺 Training.scala 🖂 🖺 Task3.scala
                                                    > 🎏 demoproject
                                                                                         // Division of 2 rational numbers
        v 👺 ScalaAssignment
                                                                                         def divideRational() : Float = {
             > 🚵 Scala Library container [ 2.12.3 ]
                                                                                           var divisibleNum : Float = num1 * denom2
             > A JRE System Library [JavaSE-1.8]
                                                                                           var divisibleDenom : Float = num2 * denom1
                                                                                           return (divisibleNum/divisibleDenom)
                   > # scala.assignment14

    # scala.assignment15

                        > Task1.scala
                                                                                         // Get the GCD of two numbers
                         > S Task2_1.scala
                                                                                         def gcd(a: Int, b: Int): Int = {
                         > Task2_2.scala
                                                                                           if (a == 0) {
                         > Task3.scala
                                                                                             return b

    # scala.assignment16

                         > SampleScala.scala
                                                                                           return gcd(b%a, a)
                         > S Training.scala
        > 🚟 ScalaHello
                                                                                      }
```

```
workspace - ScalaAssignment/src/scala/assignment16/Training.scala - Scala IDE
File Edit Refactor Navigate Search Project Run Window Help
_ _ _
₽ Project Explorer 🏻
                                      // Get the GCD of two numbers
    > 🞏 demoproject
                                            def gcd(a: Int, b: Int): Int = {
    if (a == 0) {
      > Marcola Scala Library container [ 2.12.3 ]
                                              return b
       > A JRE System Library [JavaSE-1.8]
      🗸 进 src
                                             return gcd(b%a, a)
         > # scala.assignment14
         scala.assignment15
            > S Task1.scala
            > Task2_1.scala
                                        object Training {
            > S Task2_2.scala
                                           def main(args : Array[String]) : Unit = {
            > Task3.scala
         scala.assignment16
                                             var numerator1: Int = 5
            > SampleScala.scala
                                             var denominator1: Int = 2
                                             var numerator2: Int = 3
            > S Training.scala
                                             var denominator2 : Int = 4
    > 👸 ScalaHello
                                             var rational1 = new Rational(numerator1,denominator1,numerator2,denominator2)
                                             var addResult : Float = rational1.addRational()
                                             var subtractResult: Float = rational1.subtractRational()
                                             var multiplyResult : Float = rational1.multiplyRational()
                                             var divideResult : Float = rational1.divideRational()
                                             println("INPUT\n----")
                                             println(numerator1 + "/" + denominator1 + " and " + numerator2 + "/" + denominator2)
                                             println("RESULT: \n----")
                                             println("Addition of Rational Numbers: " + addResult)
                                             println("Subtraction of Rational Numbers: " + subtractResult)
                                             println("Multiplication of Rational Numbers: " + multiplyResult)
                                             println("Division of Rational Numbers: " + divideResult)
```

RESULT:

