**Assignment 14.1:**

Create a calculator to work with rational numbers.

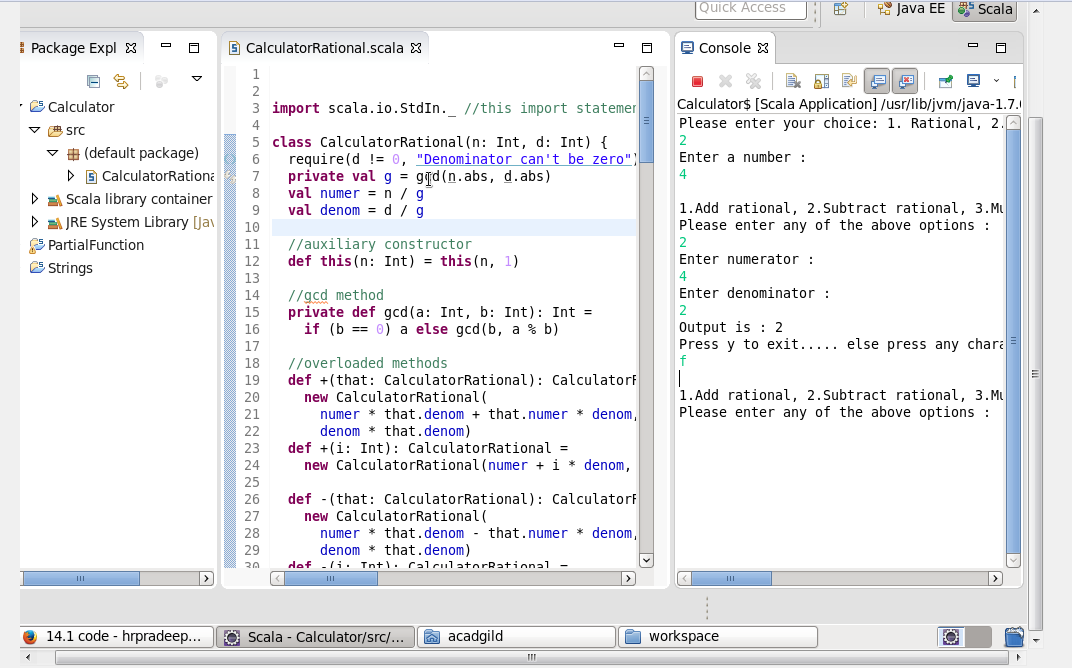
Requirements:

* 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.



**Below is the Scala code:**



import scala.io.StdIn.\_ //this import statement is required to use readInt(),readChar() methods  
  
class CalculatorRational(n: Int, d: Int) {  
  require(d != 0, "Denominator can't be zero")  
  private val g = gcd(n.abs, d.abs)  
  val numer = n / g  
  val denom = d / g  
  
  //auxiliary constructor  
  def this(n: Int) = this(n, 1)  
  
  //gcd method  
  private def gcd(a: Int, b: Int): Int =  
    if (b == 0) a else gcd(b, a % b)  
  
  //overloaded methods  
  def +(that: CalculatorRational): CalculatorRational =  
    new CalculatorRational(  
      numer \* that.denom + that.numer \* denom,  
      denom \* that.denom)  
  def +(i: Int): CalculatorRational =  
    new CalculatorRational(numer + i \* denom, denom)  
  
  def -(that: CalculatorRational): CalculatorRational =  
    new CalculatorRational(  
      numer \* that.denom - that.numer \* denom,  
      denom \* that.denom)  
  def -(i: Int): CalculatorRational =  
    new CalculatorRational(numer - i \* denom, denom)  
  
  def \*(that: CalculatorRational): CalculatorRational =  
    new CalculatorRational(numer \* that.numer, denom \* that.denom)  
  def \*(i: Int): CalculatorRational =  
    new CalculatorRational(numer \* i, denom)  
  
  def /(that: CalculatorRational): CalculatorRational =  
    new CalculatorRational(numer \* that.denom, denom \* that.numer)  
  def /(i: Int): CalculatorRational =  
    new CalculatorRational(numer, denom \* i)  
}  
  
object Calculator {  
  
  //Accepts option from user to work either with rational or with whole (integer) number  
  def NumberChoice() = {  
    println("Please enter your choice: 1. Rational, 2. Whole")  
  }  
  
  //creates a rational number  
  def makeRational(rational: CalculatorRational): CalculatorRational = {  
    println("Enter numerator : ")  
    val p = readInt()  
    println("Enter denominator : ")  
    val q = readInt()  
    rational.+(new CalculatorRational(p, q))  
  }  
  
  //provides option of operations to user  
  def Options() = {  
    println(" ")  
    println("1.Add rational, 2.Subtract rational, 3.Multiply rational, 4.Divide rational, 5.Add integer, 6.Subtract integer, 7.Multiply integer, 8.Divide integer")  
  }  
  
  //depending upon option selected by user calls the overloaded method  
  def Compute(rational: CalculatorRational, input: Int): CalculatorRational = {  
    input match {  
      case 1 =>  
        println("Enter numerator : ")  
        val p = readInt()  
        println("Enter denominator : ")  
        val q = readInt()  
        rational.+(new CalculatorRational(p, q))  
      case 2 =>  
        println("Enter numerator : ")  
        val p = readInt()  
        println("Enter denominator : ")  
        val q = readInt()  
        rational.-(new CalculatorRational(p, q))  
      case 3 =>  
        println("Enter numerator : ")  
        val p = readInt()  
        println("Enter denominator : ")  
        val q = readInt()  
        rational.\*(new CalculatorRational(p, q))  
      case 4 =>  
        println("Enter numerator : ")  
        val p = readInt()  
        println("Enter denominator : ")  
        val q = readInt()  
        rational./(new CalculatorRational(p, q))  
      case 5 =>  
        println("Enter an integer for addition: ")  
        val p = readInt()  
        rational.+(new CalculatorRational(p))  
      case 6 =>  
        println("Enter an integer for subtraction : ")  
        val p = readInt()  
        rational.-(new CalculatorRational(p))  
      case 7 =>  
        println("Enter an integer for multiplication: ")  
        val p = readInt()  
        rational.\*(new CalculatorRational(p))  
      case 8 =>  
        println("Enter a non-zero integer for division: ")  
        val p = readInt()  
        rational./(new CalculatorRational(p))  
      case \_ =>  
        rational  
    }  
  }  
  
  //runs loop until user types 'y' to exit  
  def main(args: Array[String]): Unit = {  
  
    var rationalNumber1: CalculatorRational = new CalculatorRational(0)  
    var rationalNumber2: CalculatorRational = new CalculatorRational(0)  
    var num = 0  
    var choice1 = 0  
    var choice2 = 0  
    var ch = 'y'  
  
    NumberChoice()  
    choice1 = readInt()  
    choice1 match { //matching the user choice  
      case 1 =>  
        rationalNumber1 = makeRational(rationalNumber1)  
        println("CalculatorRational Number is : " + rationalNumber1.numer+"/"+rationalNumber1.denom)  
        rationalNumber2 = rationalNumber1  
        do {  
          Options()  
          println("Please enter any of the above options : ")  
          choice2 = readInt()  
          rationalNumber2 = Compute(rationalNumber2, choice2)  
          if (rationalNumber2.denom == 1)  
            println("Output is : " + rationalNumber2.numer)  
          else if (rationalNumber2.denom < 0)  
            println("Output is : " + "-" + rationalNumber2.numer + "/" + rationalNumber2.denom.abs)  
          else  
            println("Output is : " + rationalNumber2.numer + "/" + rationalNumber2.denom)  
          println("Press y to exit..... else press any character to continue...")  
          ch = readChar()  
        } while (ch != 'y')  
  
      case 2 =>  
        println("Enter a number : ")  
        num = readInt()  
        rationalNumber2 = new CalculatorRational(num)  
        do {  
          Options()  
          println("Please enter any of the above options : ")  
          choice2 = readInt()  
          rationalNumber2 = Compute(rationalNumber2, choice2)  
          if (rationalNumber2.denom == 1)  
            println("Output is : " + rationalNumber2.numer)  
          else if (rationalNumber2.denom < 0)  
            println("Output is : " + "-" + rationalNumber2.numer + "/" + rationalNumber2.denom.abs)  
          else  
            println("Output is : " + rationalNumber2.numer + "/" + rationalNumber2.denom)  
          println("Press y to exit..... else press any character to continue...")  
          ch = readChar()  
        } while (ch != 'y')  
  
    }  
  
  }  
  
}