Assignment 16

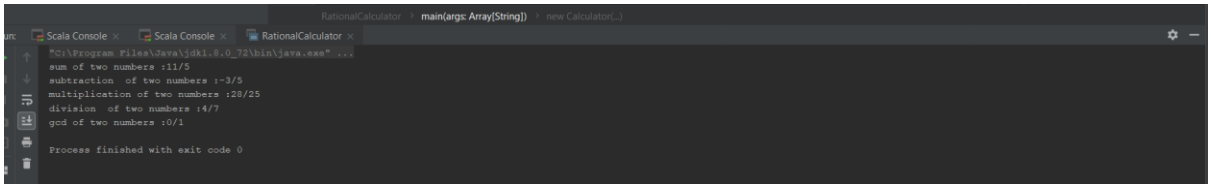
Create a calculator to work with rational numbers. Requirements: • It should provide capability to add, subtract, divide and multiply rational Numbers. • Create a method to compute GCD (this will come in handy during operations on Rational). Add option to work with whole numbers, which are also rational numbers i.e. (n/1) • Achieve the above using auxiliary constructors • Enable method overloading to enable each function to work with numbers and rational.

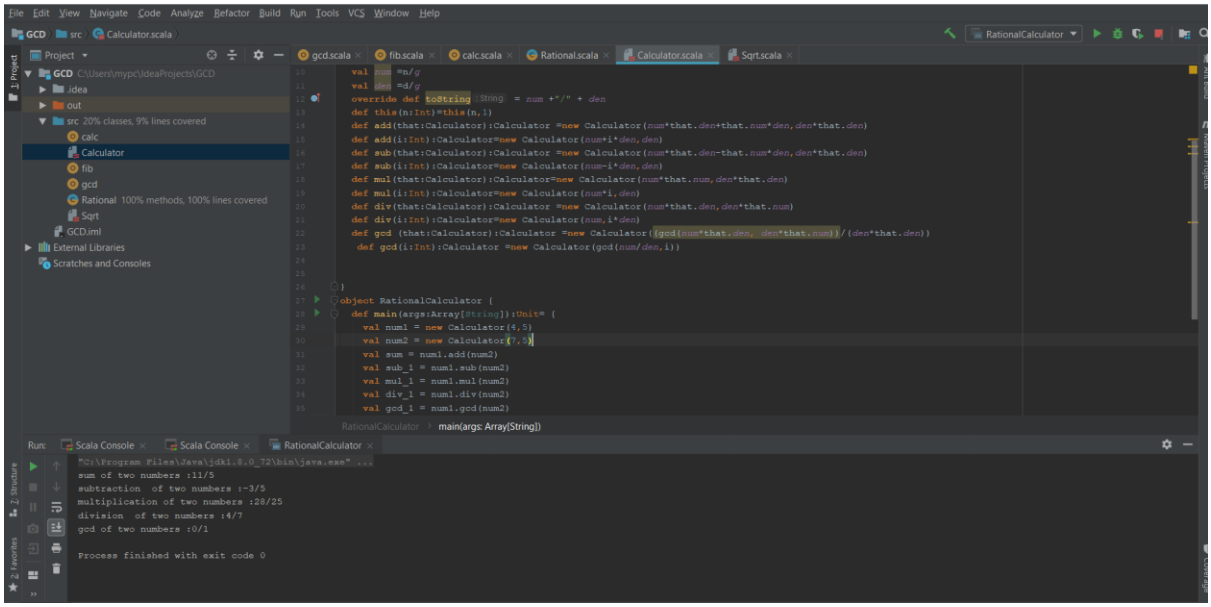
```
class Calculator (n:Int,d:Int){
  require(d!=0)    // for a rational  number denominator should be a natural number
// function to define gcd
  private def gcd (x:Int,y:Int):Int={
    if (y==0) x
    else if (x<0)gcd(x,y)
    else if (y<0) gcd(x,y)
    else gcd(y,x%y)
  }
//to calculate absolute gcd of numerator and denominator
  private val g =gcd(n.abs,d.abs)
  val num =n/g
  val den =d/g
//override the default implementation by adding a method toString  to class calculator .This will help us to print the values of numerator and denominator of rational number.
  override def toString = num +"/" + den
//auxillary constructor
  def this(n:Int)=this(n,1)
//function to define to calculate add operation
  def add(that:Calculator):Calculator =new Calculator(num*that.den+that.num*den,den*that.den)
//add method overloading
  def add(i:Int):Calculator=new Calculator(num+i*den,den)
```

```
//function to define to calculate subtract operation

  def sub(that:Calculator):Calculator =new Calculator(num*that.den-that.num*den,den*that.den)
  def sub(i:Int):Calculator=new Calculator(num-i*den,den)
//function to define to calculate multiply operation

  def mul(that:Calculator):Calculator=new Calculator(num*that.num,den*that.den)
  def mul(i:Int):Calculator=new Calculator(num*i,den)
//function to define to calculate division operation

  def div(that:Calculator):Calculator =new Calculator(num*that.den,den*that.num)
  def div(i:Int):Calculator=new Calculator(num,i*den)
//function to define to calculate gcd operation

  def gcd (that:Calculator):Calculator =new Calculator((gcd(num*that.den,
den*that.num))/(den*that.den))
   def gcd(i:Int):Calculator =new Calculator(gcd(num/den,i))



}
object RationalCalculator {
  def main(args:Array[String]):Unit= {
    val num1 = new Calculator(20)
    val num2 = new Calculator(32)
    val sum = num1.add(num2)
    val sub_1 = num1.sub(num2)
    val mul_1 = num1.mul(num2)
    val div_1 = num1.div(num2)
    val gcd_1 = num1.gcd(num2)
    println("sum of two numbers :" + sum)
    println("subtraction  of two numbers :" + sub_1)
```

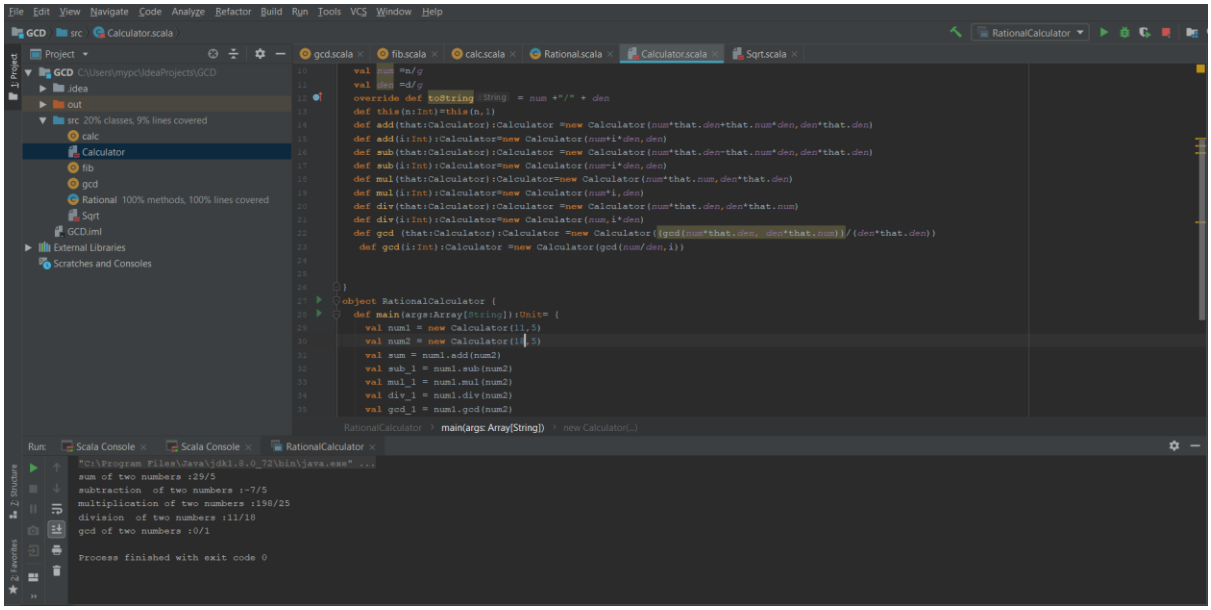
```
println("multiplication of two numbers :" + mul_1)
println("division  of two numbers :" + div_1)
println("gcd of two numbers :" + gcd_1)
}
}
```

O/P: Output with rational numbers, num1=4/5 and num2=7/5





O/P: Output with rational numbers, num1=4/5 and num2=7/5


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
GCD src Calculator.scala
RationalCalculator
Project
GCD C:\Users\mypc\IdeaProjects\GCD
.idea
out
src 20% classes, 9% lines covered
calc
Calculator
fib
gcd
Rational 100% methods, 100% lines covered
Sqrt
GCD.iml
External Libraries
Scratches and Consoles
gcd.scala
fib.scala
calc.scala
Rational.scala
Calculator.scala
Sqrt.scala
val num =n/g
val den =d/g
override def toString : String = num +"/" + den
def this(n:Int)=this(n,1)
def add(that:Calculator):Calculator =new Calculator(num*that.den+that.num*den,den*that.den)
def add(i:Int):Calculator=new Calculator(num+i*den,den)
def sub(that:Calculator):Calculator =new Calculator(num*that.den-that.num*den,den*that.den)
def sub(i:Int):Calculator=new Calculator(num-i*den,den)
def mul(that:Calculator):Calculator=new Calculator(num*that.num,den*that.den)
def mul(i:Int):Calculator=new Calculator(num*i,den)
def div(that:Calculator):Calculator =new Calculator(num*that.den,den*that.num)
def div(i:Int):Calculator=new Calculator(num,i*den)
def gcd (that:Calculator):Calculator =new Calculator((gcd(num*that.den, den*that.num))/(den*that.den))
def gcd(i:Int):Calculator =new Calculator(gcd(num/den,i))
}
object RationalCalculator {
def main(args:Array[String]):Unit= {
val num1 = new Calculator(11,5)
val num2 = new Calculator(18,5)
val sum = num1.add(num2)
val sub_1 = num1.sub(num2)
val mul_1 = num1.mul(num2)
val div_1 = num1.div(num2)
val gcd_1 = num1.gcd(num2)
RationalCalculator › main(args: Array[String]) › new Calculator(...)
Run: Scala Console Scala Console RationalCalculator
"C:\Program Files\Java\jdk1.8.0_72\bin\java.exe" ...
sum of two numbers :29/5
subtraction of two numbers :-7/5
multiplication of two numbers :198/25
division of two numbers :11/18
gcd of two numbers :0/1
Process finished with exit code 0
Z: Structure
2: Favorites