Session 14: Scala - Session III

Assignment 1 Solution

# **Problem**

**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)**

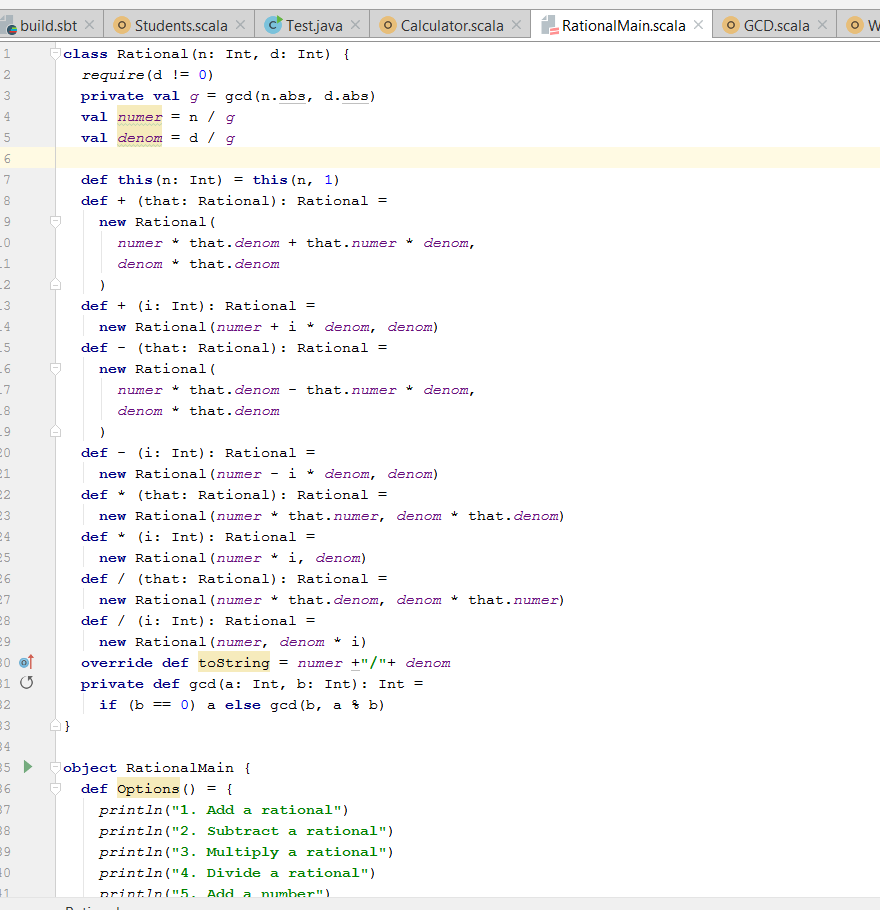
**○ Achieve the above using auxiliary constructors**

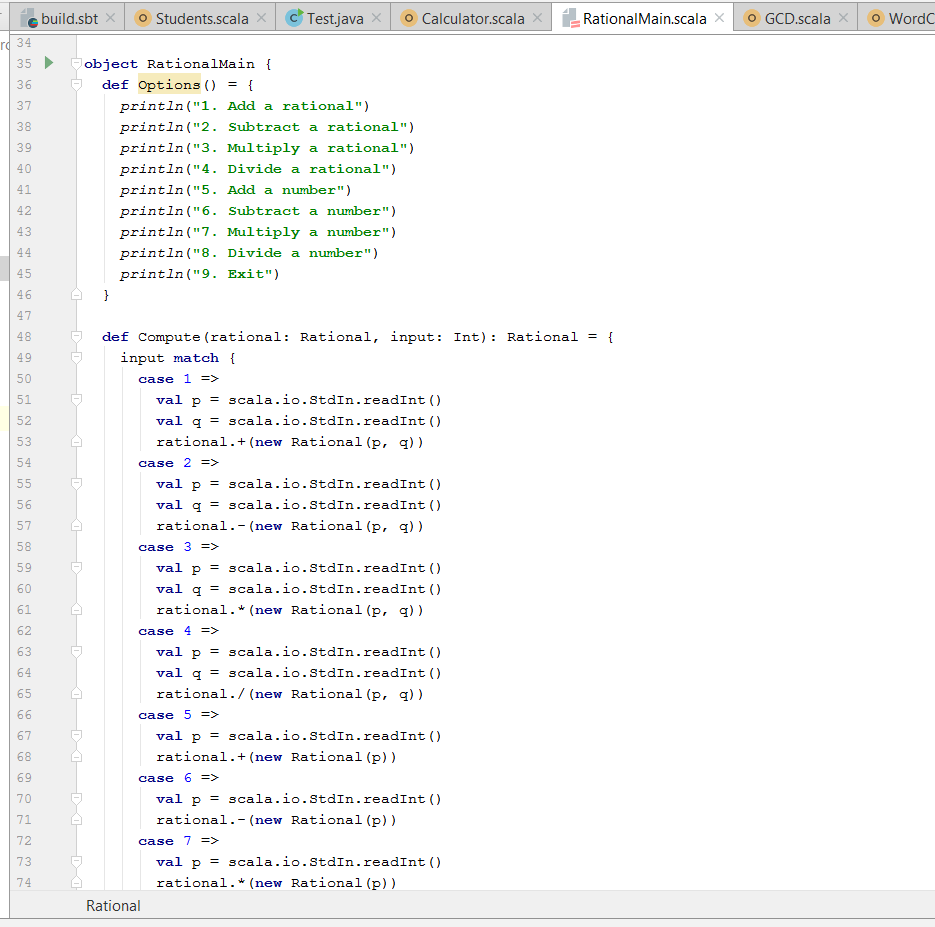
**○ Enable method overloading to enable each function to work with numbers and rational.**

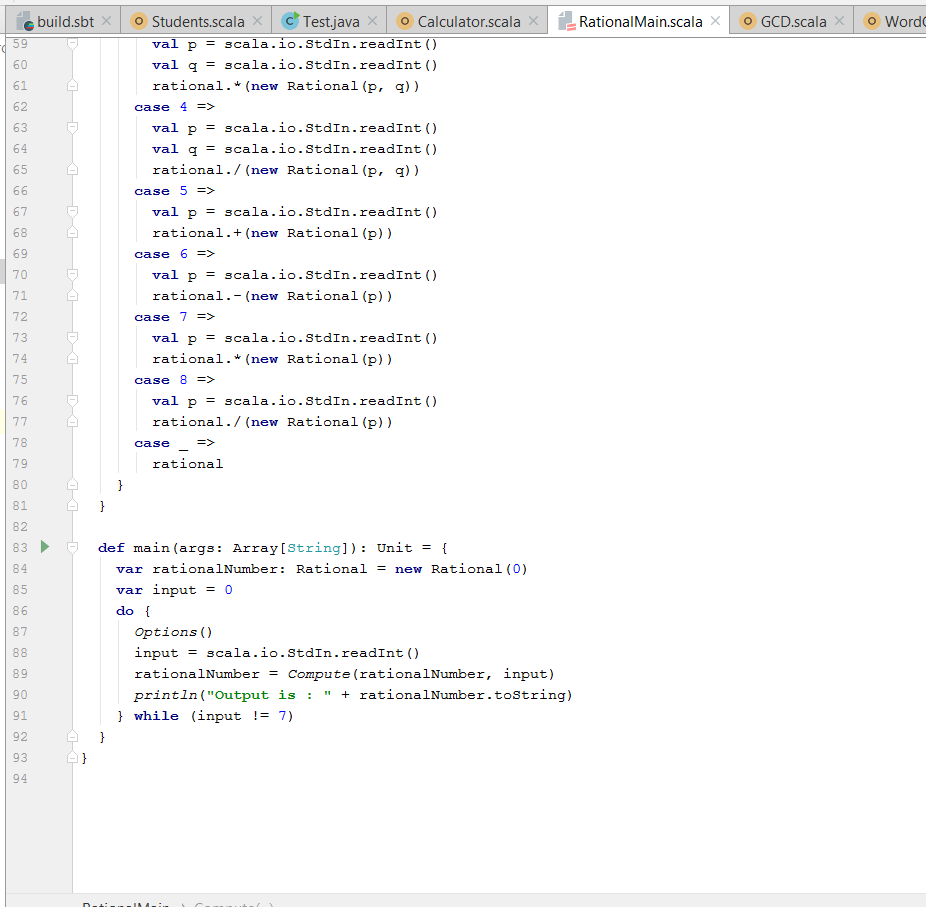
# **Solution**

**class** Rational(n: Int, d: Int) {  
 *require*(d != 0)  
 **private val** *g* = gcd(n.abs, d.abs)  
 **val** *numer* = n / *g* **val** *denom* = d / *g* **def this**(n: Int) = **this**(n, 1)  
 **def** + (that: Rational): Rational =  
 **new** Rational(  
 *numer* \* that.*denom* + that.*numer* \* *denom*,  
 *denom* \* that.*denom* )  
 **def** + (i: Int): Rational =  
 **new** Rational(*numer* + i \* *denom*, *denom*)  
 **def** - (that: Rational): Rational =  
 **new** Rational(  
 *numer* \* that.*denom* - that.*numer* \* *denom*,  
 *denom* \* that.*denom* )  
 **def** - (i: Int): Rational =  
 **new** Rational(*numer* - i \* *denom*, *denom*)  
 **def** \* (that: Rational): Rational =  
 **new** Rational(*numer* \* that.*numer*, *denom* \* that.*denom*)  
 **def** \* (i: Int): Rational =  
 **new** Rational(*numer* \* i, *denom*)  
 **def** / (that: Rational): Rational =  
 **new** Rational(*numer* \* that.*denom*, *denom* \* that.*numer*)  
 **def** / (i: Int): Rational =  
 **new** Rational(*numer*, *denom* \* i)  
 **override def** toString = *numer* +**"/"**+ *denom* **private def** gcd(a: Int, b: Int): Int =  
 **if** (b == 0) a **else** gcd(b, a % b)  
}  
  
**object** RationalMain {  
 **def** Options() = {  
 *println*(**"1. Add a rational"**)  
 *println*(**"2. Subtract a rational"**)  
 *println*(**"3. Multiply a rational"**)  
 *println*(**"4. Divide a rational"**)   
 *println*(**"5. Add a number"**)   
 *println*(**"6. Subtract a number"**)   
 *println*(**"7. Multiply a number"**)  
 *println*(**"8. Divide a number"**)   
 *println*(**"9. Exit"**)   
 }  
  
 **def** Compute(rational: Rational, input: Int): Rational = {  
 input **match** {  
 **case** 1 =>  
 **val** p = scala.io.StdIn.readInt()  
 **val** q = scala.io.StdIn.readInt()  
 rational.+(**new** Rational(p, q))  
 **case** 2 =>  
 **val** p = scala.io.StdIn.readInt()  
 **val** q = scala.io.StdIn.readInt()  
 rational.-(**new** Rational(p, q))  
 **case** 3 =>  
 **val** p = scala.io.StdIn.readInt()  
 **val** q = scala.io.StdIn.readInt()  
 rational.\*(**new** Rational(p, q))  
 **case** 4 =>  
 **val** p = scala.io.StdIn.readInt()  
 **val** q = scala.io.StdIn.readInt()  
 rational./(**new** Rational(p, q))  
 **case** 5 =>  
 **val** p = scala.io.StdIn.readInt()  
 rational.+(**new** Rational(p))  
 **case** 6 =>  
 **val** p = scala.io.StdIn.readInt()  
 rational.-(**new** Rational(p))  
 **case** 7 =>  
 **val** p = scala.io.StdIn.readInt()  
 rational.\*(**new** Rational(p))  
 **case** 8 =>  
 **val** p = scala.io.StdIn.readInt()  
 rational./(**new** Rational(p))  
 **case** \_ =>  
 rational  
 }  
 }  
  
 **def** main(args: Array[String]): Unit = {  
 **var** rationalNumber: Rational = **new** Rational(0)  
 **var** input = 0  
 **do** {  
 *Options*()  
 input = scala.io.StdIn.readInt()  
 rationalNumber = *Compute*(rationalNumber, input)  
 *println*(**"Output is : "** + rationalNumber.toString)  
 } **while** (input != 7)  
 }  
}

Source code:





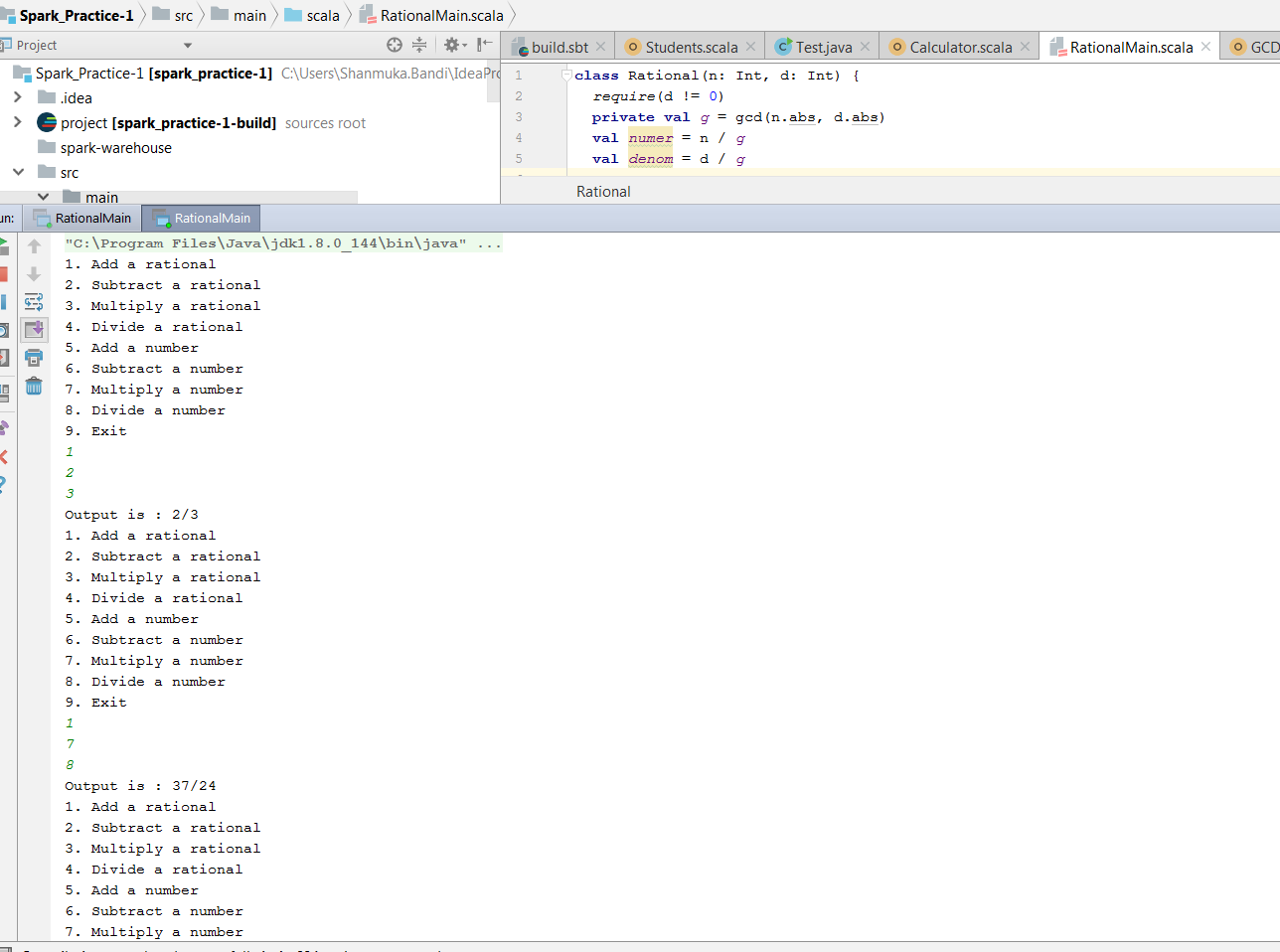


For operators(+, -, \*, /) we are using numbers as choice. Like rational addition 1, rational subraction 2, division 8, etc.

# **Output**

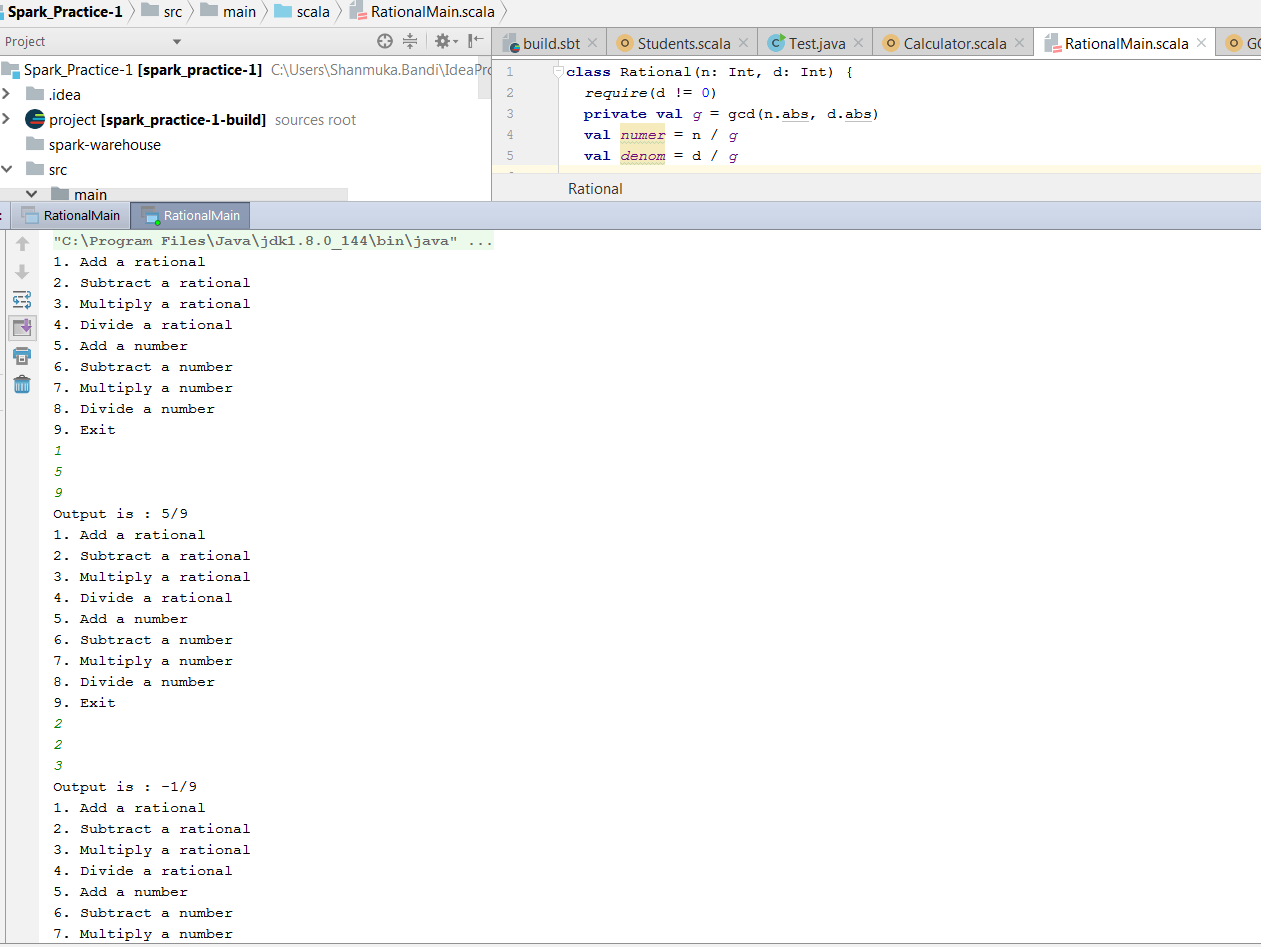
Rational Number addition:

2/3 + 7/8 = 37/24



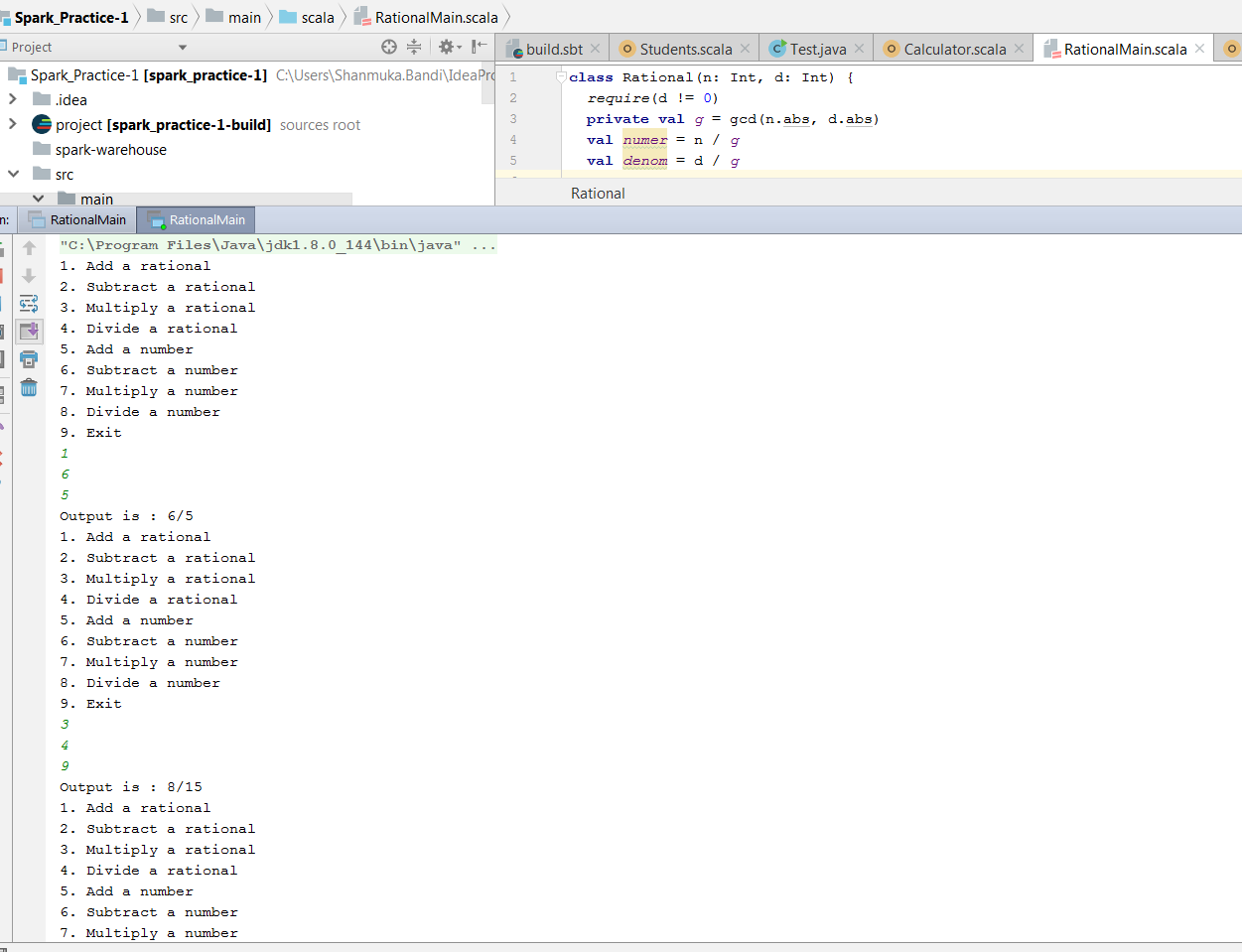
Rational Subraction:

5/9 – 2/3 = -1/9



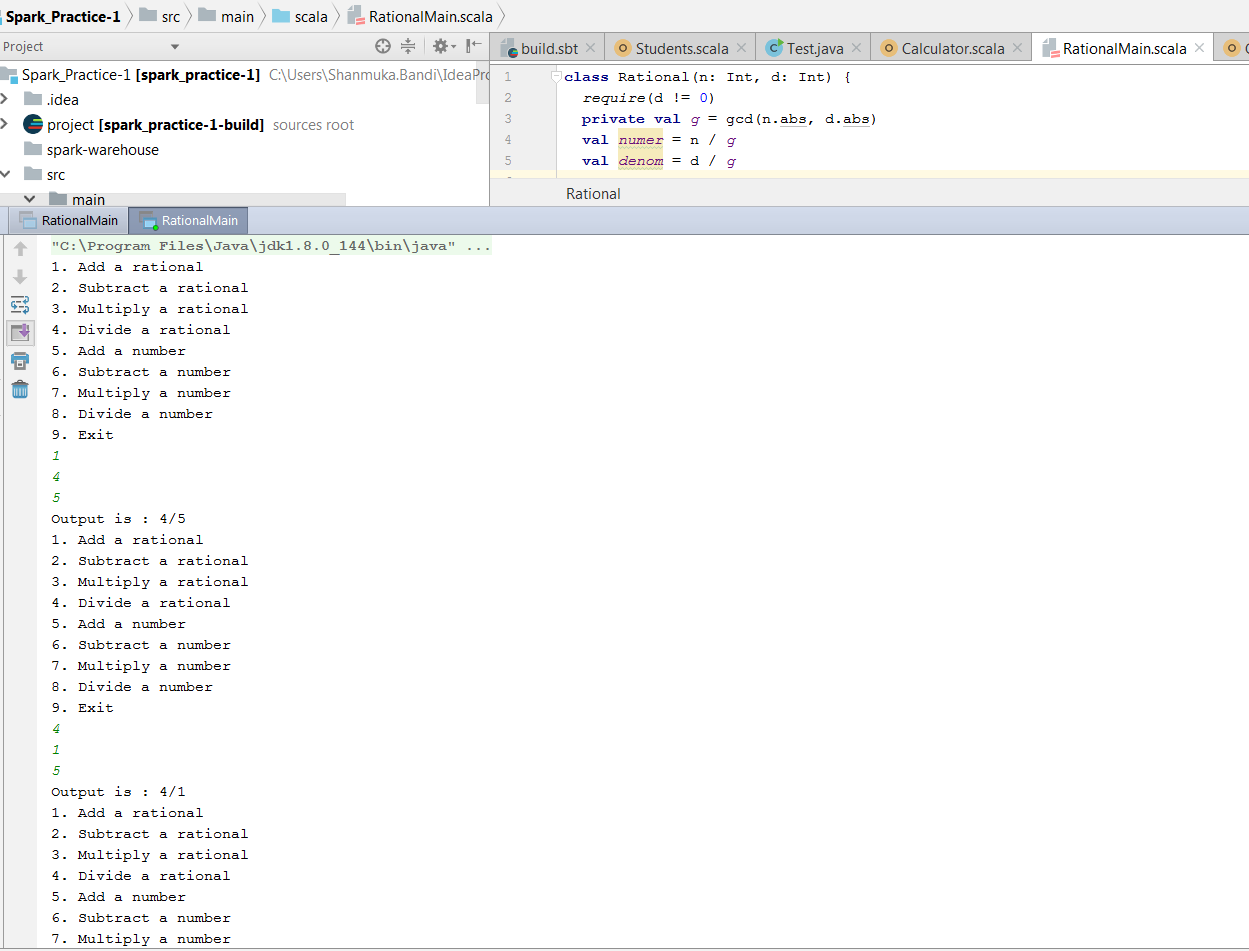
Rational Multiplication:

6/5 \* 4/9 = 24/45 = 8/15.



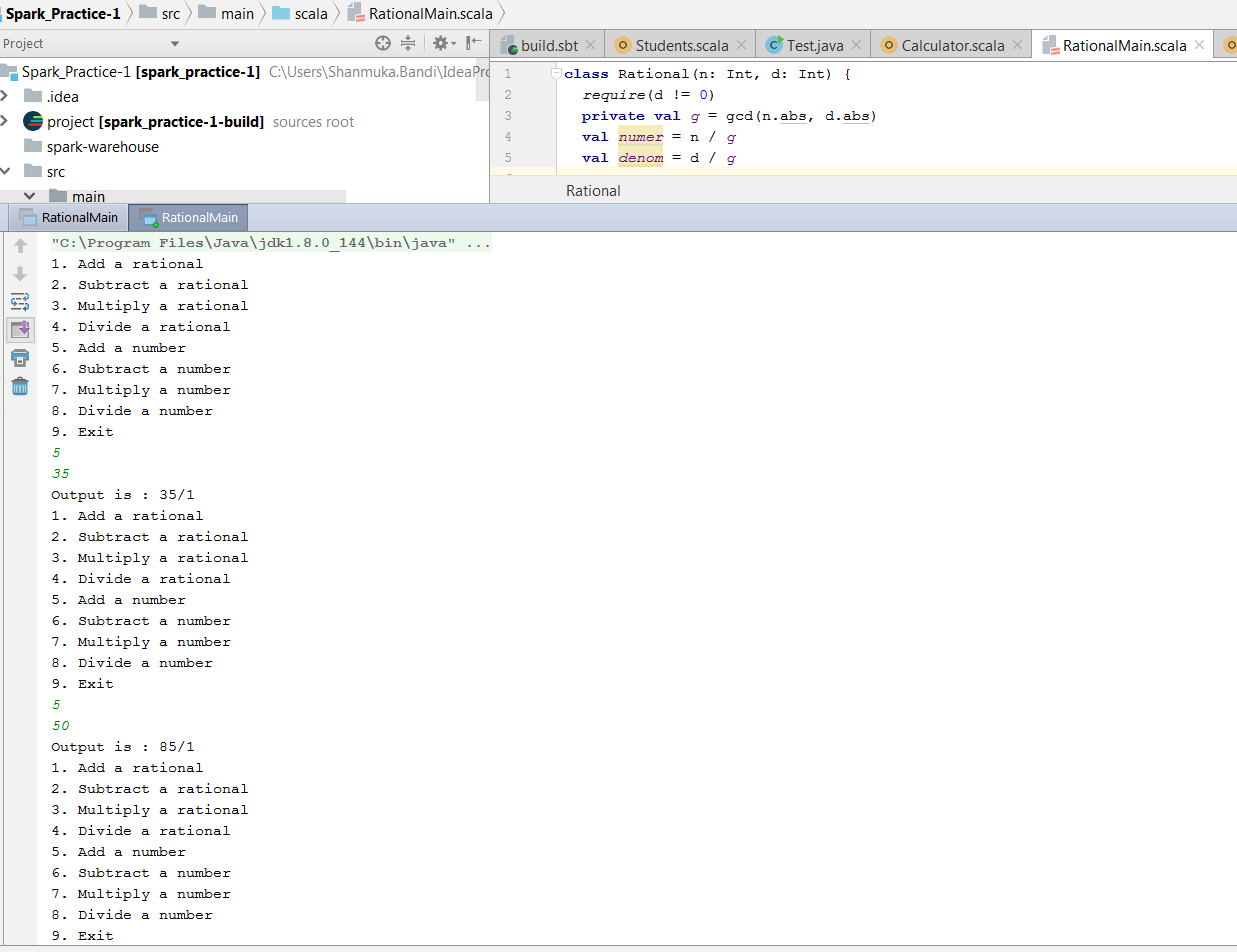
Rational Division:

(4/5) / (1/5) = 4/1



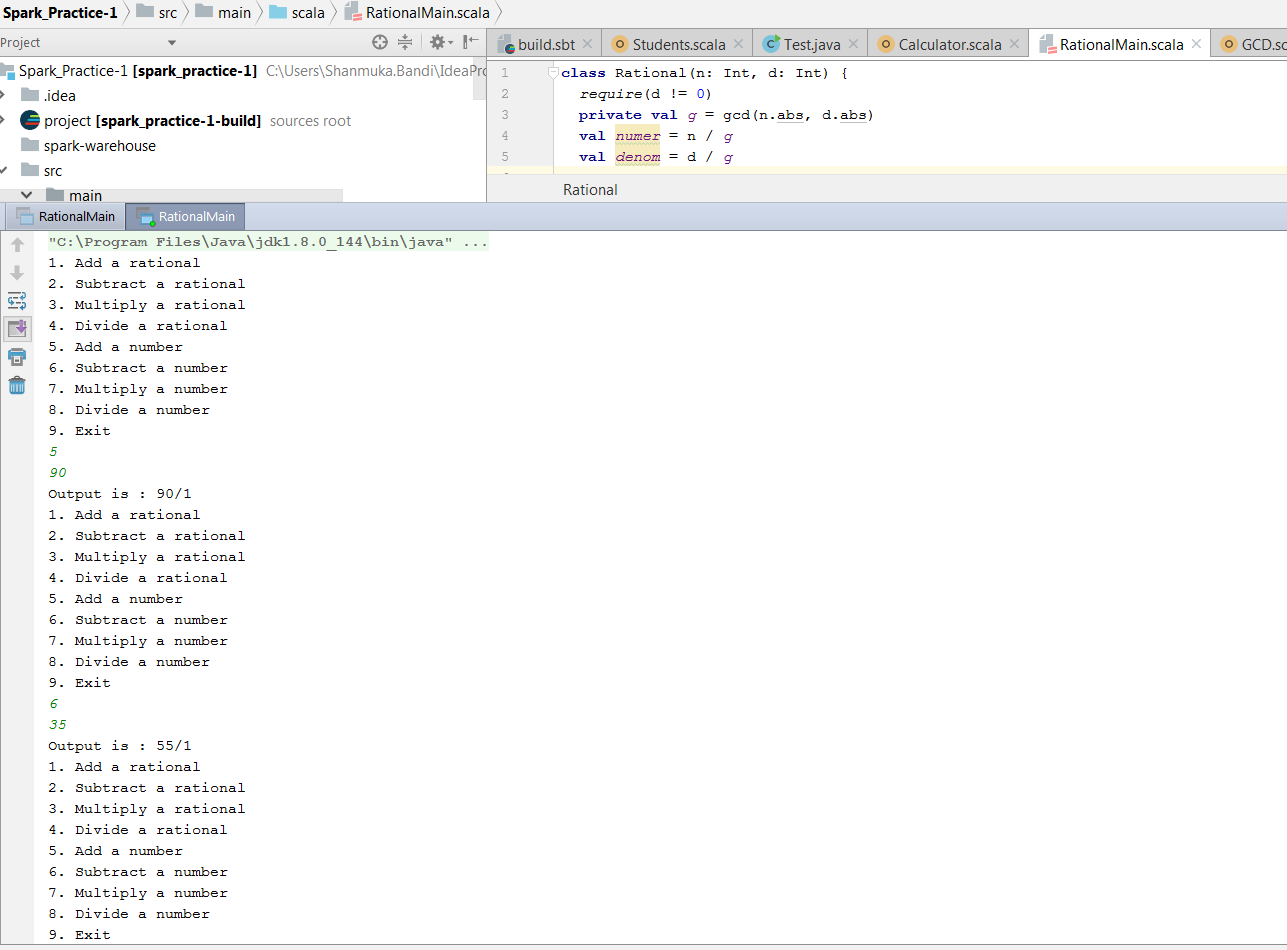
Addition:

35 + 50 = 85.



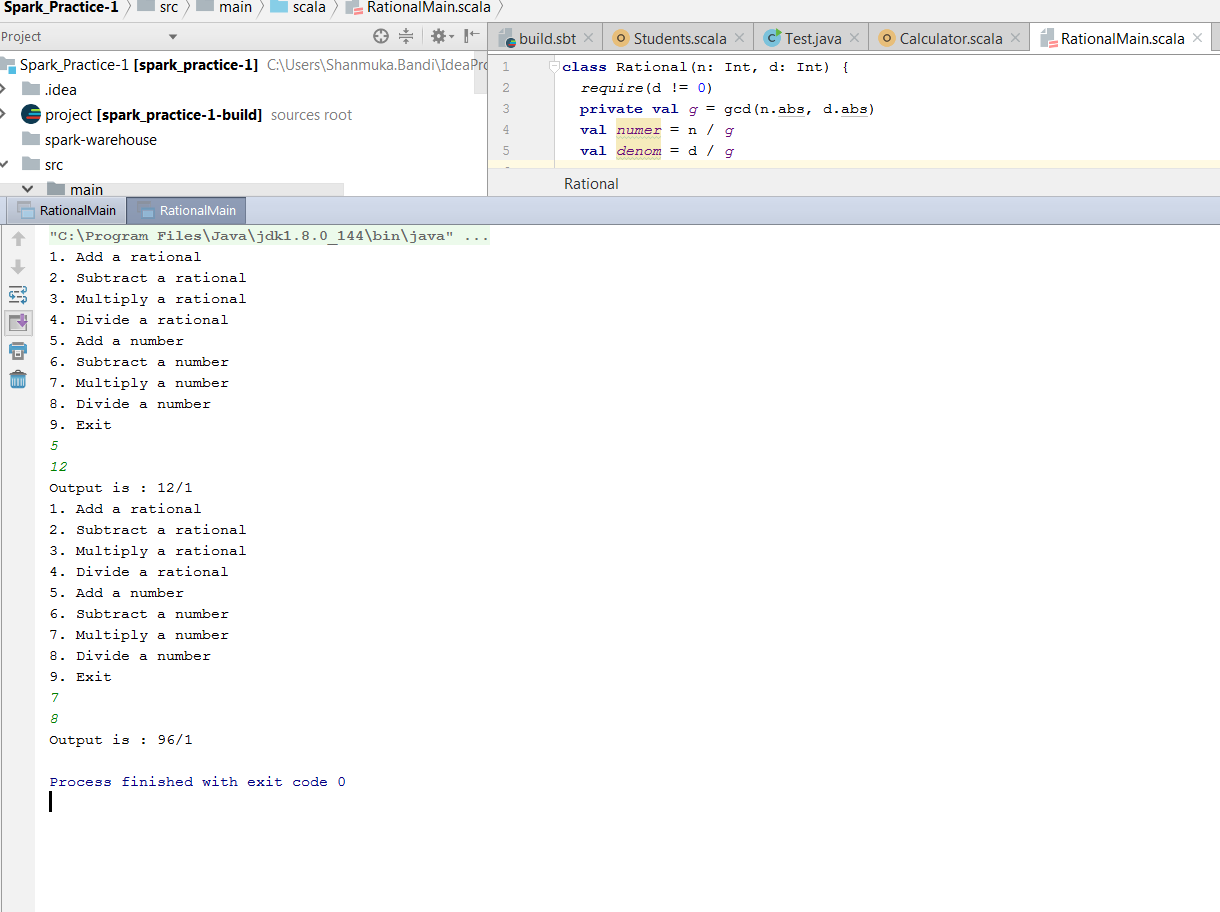
Subraction:

90 – 35 = 55



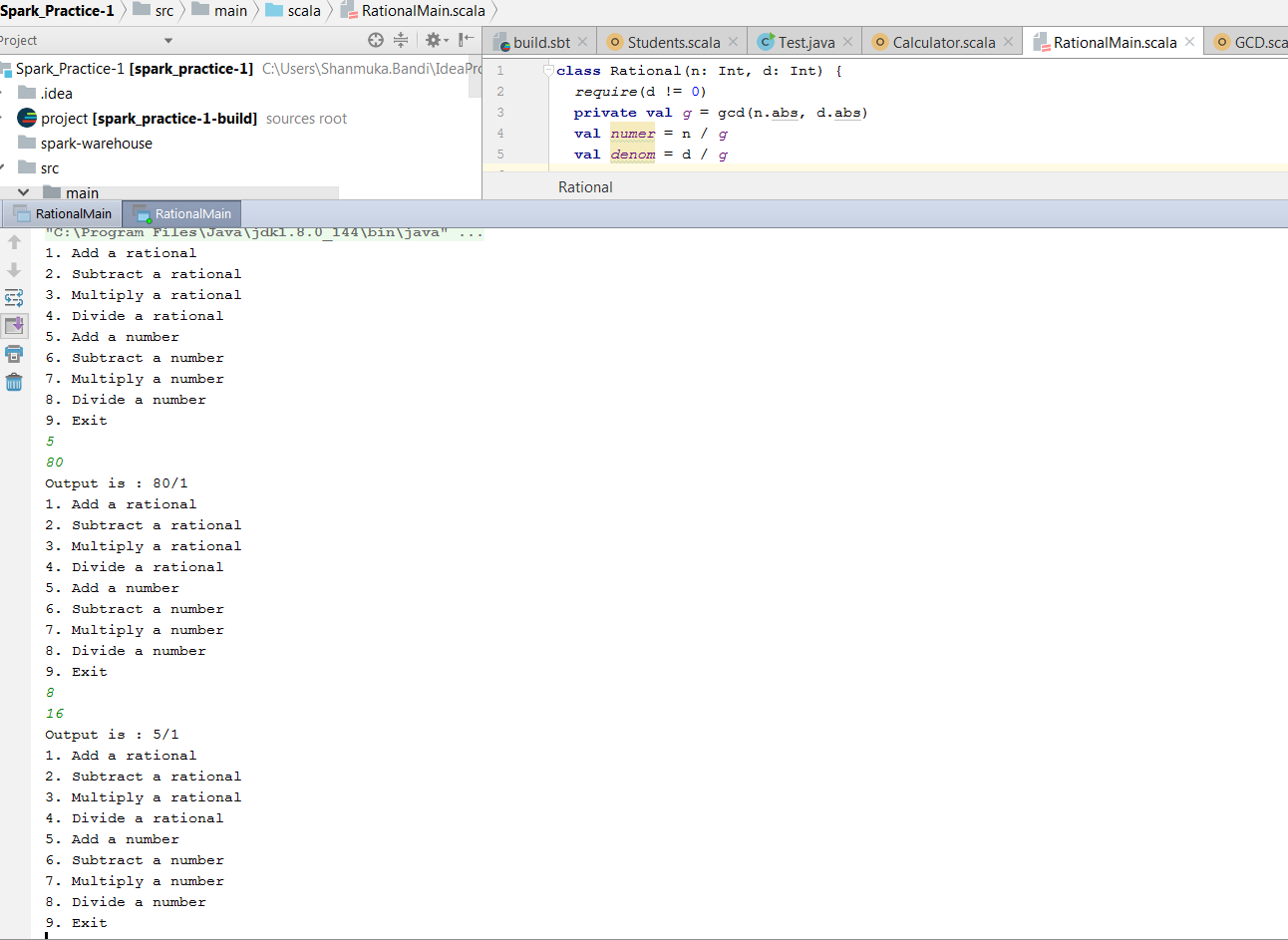
Multiplication:

12 \* 8 = 96



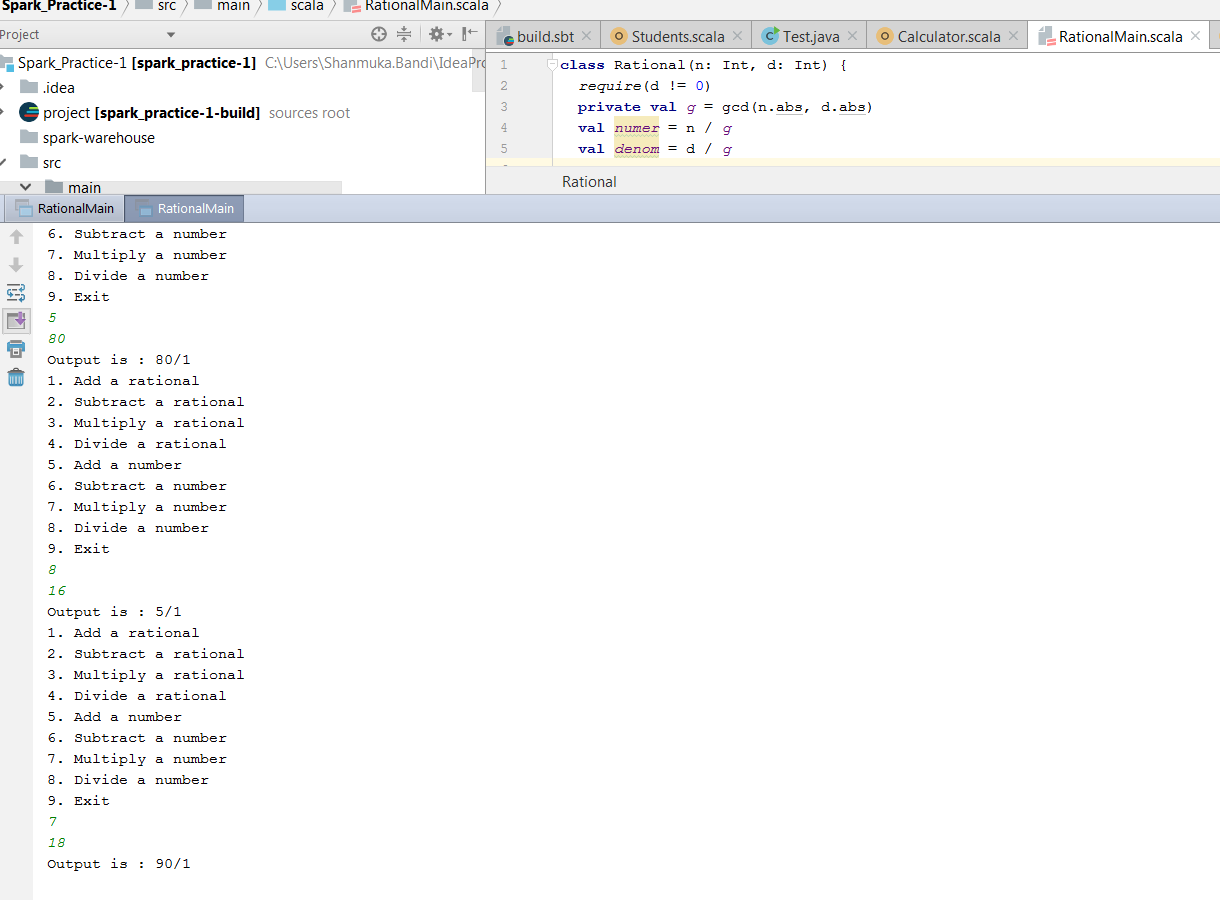
Division:

80 / 16 = 5/1



Complex calculation:

(80 / 16) \* 18 = 90



Complex calculation:

{ ( ( (7/8) – (3/5) ) \* (2/9) ) / (5/3) } = 11/300

