

Scala 3

Methods(+,-,/,*) for rational and whole numbers using parameterised and no argument auxiliary constructors are written **class Rational**

Inputs are taken in **Object Assignment16**

Source code screen shot

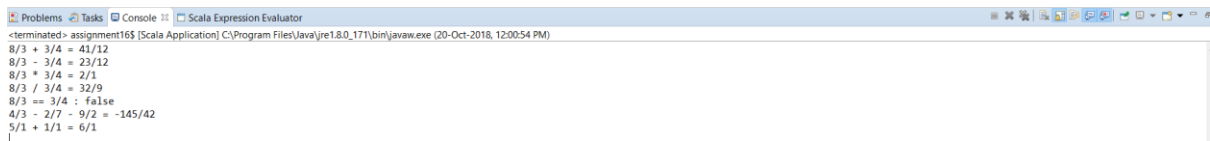
Class

```
class Rational(n: Int, d: Int) {
  require(d != 0)
  //Defining constructor
  def this(n: Int) = this(n, 1) // auxiliary constructor to support single arg
  def this() = this(1, 1) // auxiliary constructor to support no arg
  private val g = gcd(n.abs, d.abs)
  val number = n / g
  val denom = d / g
  //Add for Rational numbers
  def + (that: Rational): Rational =
    new Rational(
      number * that.denom + that.number * denom,
      denom * that.denom
    )
  //Overloaded Add for whole numbers
  def + (i: Int): Rational =
    new Rational(number + i * denom, denom)
  //Subtract for Rational numbers
  def - (that: Rational): Rational =
    new Rational(
      number * that.denom - that.number * denom,
      denom * that.denom
    )
  //Overloaded Subtract for whole numbers
  def - (i: Int): Rational =
    new Rational(number - i * denom, denom)
  //Multiply for Rational numbers
  def * (that: Rational): Rational =
    new Rational(number * that.number, denom * that.denom)
  //Overloaded Multiply for whole numbers
  def * (i: Int): Rational =
    new Rational(number * i, denom)
  //Div for Rational numbers
  def / (that: Rational): Rational =
    new Rational(number * that.denom, denom * that.number)
  //Overloaded Div for whole numbers
  def / (i: Int): Rational =
    new Rational(number, denom * i)
  //GCD
  private def gcd(a: Int, b: Int): Int =
    if (b == 0) a else gcd(b, a % b)
  //overridden To String method, to print meaningful output
  override def toString = number + "/" + denom
}
```

Object

```
object assignment16 extends App {  
  //rational number input  
  val r1 = new Rational(8, 3)  
  val r2 = new Rational(3, 4)  
  val r3 = new Rational(5, 7)  
  println(r1 + " + " + r2 + " = " + (r1 + r2));  
  println(r1 + " - " + r2 + " = " + (r1 - r2));  
  println(r1 + " * " + r2 + " = " + (r1 * r2));  
  println(r1 + " / " + r2 + " = " + (r1 / r2));  
  println(r1 + " == " + r2 + " : " + (r1 == r2))  
  val x = new Rational(4, 3)  
  val y = new Rational(2, 7)  
  val z = new Rational(9, 2)  
  println(x + " - " + y + " - " + z + " = " + (x - y - z))  
  //passing whole number as rational number  
  val a = new Rational(5)  
  val b = new Rational  
  println(a + " + " + b + " = " + (a + b));  
}
```

Out Put



The screenshot shows a Scala REPL window titled "Scala Expression Evaluator". The output of the code is as follows:

```
<terminated> assignment16$ [Scala Application] C:\Program Files\Java\jre1.8.0_171\bin\javaw.exe (20-Oct-2018, 12:00:54 PM)  
8/3 + 3/4 = 41/12  
8/3 - 3/4 = 23/12  
8/3 * 3/4 = 2/1  
8/3 / 3/4 = 32/9  
8/3 == 3/4 : false  
4/3 - 2/7 - 9/2 = -145/42  
5/1 + 1/1 = 6/1
```