Bigdata Assignment 5.4

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)

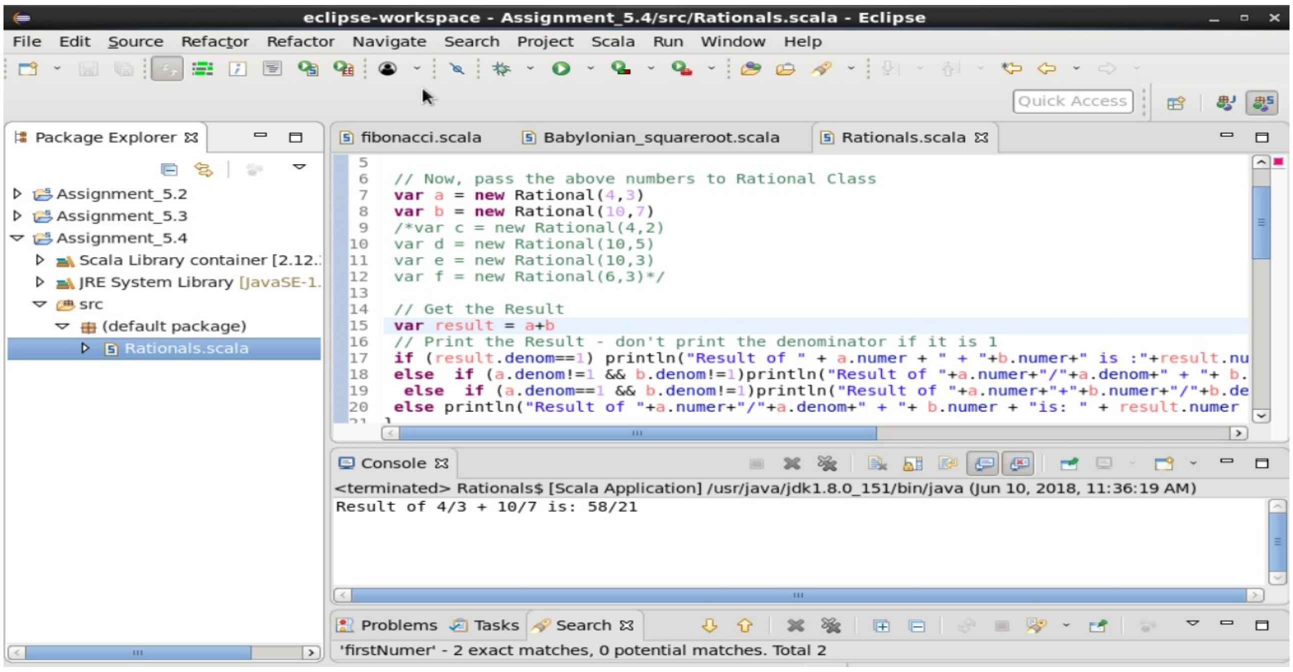
Add option to work with whole numbers which are also rational numbers i.e. (n/1)
- achieve the above using auxiliary constructors
- enable method overloading to enable each function to work with numbers and rational.

Solution -

```
object Rationals {
  def main(args: Array[String]) {

 // We are passing numbers to Rational Class for three cases when two nos
are integers , two nos are rationals and one is rational and the other is
integer.
 /*var a = new Rational(4,3)
 var b = new Rational(10,7)
 var a = new Rational(4,2)
 var b = new Rational(10,5)*/
 var a = new Rational(10,3)
 var b = new Rational(6,3)

 // Get the Result
 var result = a+b
 // Print the Result - don't print the denominator if it is 1
 if (result.denom==1) println("Result of " + a.numer + " + "+b.numer+"
is :"+result.numer)
 else  if (a.denom!=1 && b.denom!=1)println("Result of
"+a.numer+"/"+a.denom+" + "+ b.numer +"/"+b.denom+" is: " +
result.numer + "/" + result.denom)
  else  if (a.denom==1 && b.denom!=1)println("Result of
"+a.numer+"+"+b.numer+"/"+b.denom+" is: " + result.numer + "/" +
result.denom)
 else println("Result of "+a.numer+"/"+a.denom+" + "+ b.numer + "is: "
+ result.numer + "/" + result.denom)
}

}

class Rational(n: Int, d: Int) {
//GCD method
  private def gcd(x: Int, y: Int): Int = {
  if (x == 0) y
  else if (x < 0) gcd(-x, y)
  else if (y < 0) -gcd(x, -y)
  else gcd(y % x, x)
  }
  //Auxillary Constructor
  def this(x:Int){
   this(x,1)
  }
  private val g = gcd(n, d)
  val numer: Int = n/g
  val denom: Int = d/g


  //Method Overloading for each type of operation
  //Addition
  def +(that: Rational) = new Rational(numer*that.denom +
that.numer*denom,denom*that.denom)

  //Subtraction
  def -(that: Rational) = new Rational(numer*that.denom -
that.numer*denom,denom*that.denom)

  //Multiplication
  def *(that: Rational) = new Rational(numer*that.numer,
denom*that.denom)

  //Division
  def /(that: Rational) = new Rational(numer*that.denom,
denom*that.numer)
}
```

Output -

Here input was 2 rational no(4/3 and 10/7) and we performed addition. The result was 58/21.

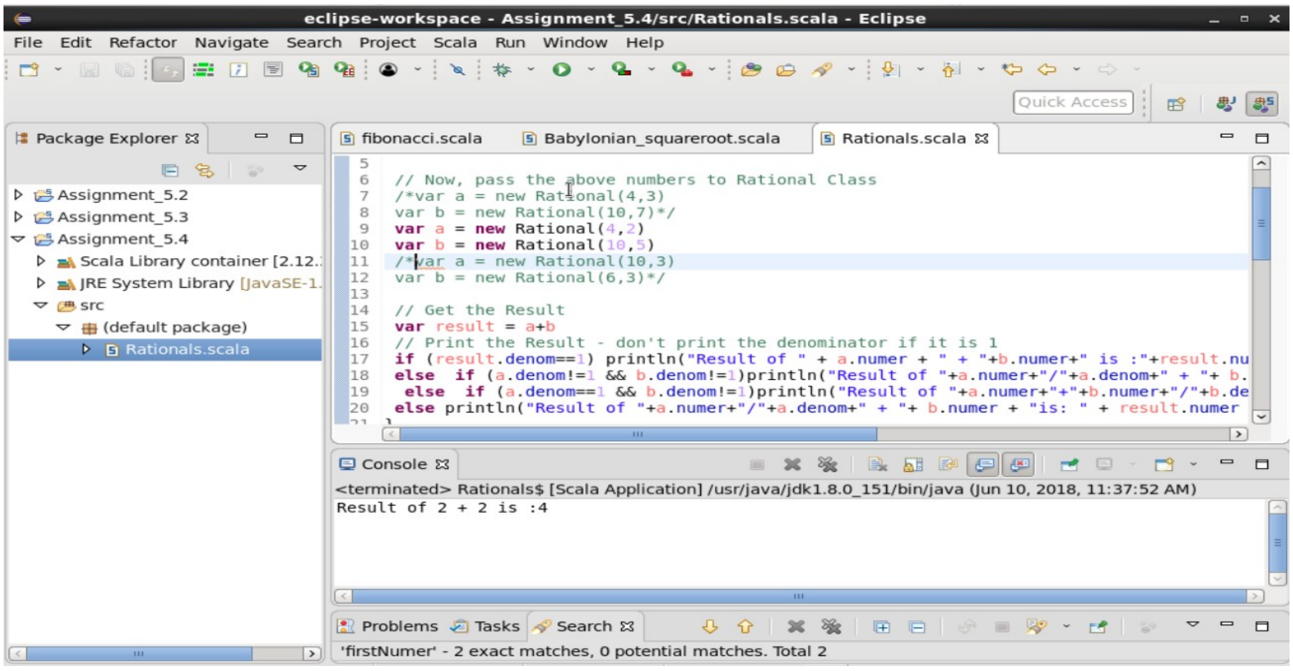


```scala
// Now, pass the above numbers to Rational Class
var a = new Rational(4,3)
var b = new Rational(10,7)
/*var c = new Rational(4,2)
var d = new Rational(10,5)
var e = new Rational(10,3)
var f = new Rational(6,3)*/

// Get the Result
var result = a+b
// Print the Result - don't print the denominator if it is 1
if (result.denom==1) println("Result of " + a.numer + " + "+b.numer+" is :"+result.nu
else  if (a.denom!=1 && b.denom!=1)println("Result of "+a.numer+"/"+a.denom+" + "+ b.
 else  if (a.denom==1 && b.denom!=1)println("Result of "+a.numer+"+"+b.numer+"/"+b.de
else println("Result of "+a.numer+"/"+a.denom+" + "+ b.numer + "is: " + result.numer
```

```
<terminated> Rationals$ [Scala Application] /usr/java/jdk1.8.0_151/bin/java (Jun 10, 2018, 11:36:19 AM)
Result of 4/3 + 10/7 is: 58/21
```

```
'firstNumer' - 2 exact matches, 0 potential matches. Total 2
```

Here input was 2 integers(2 aand 2) and performed addition , we got the output as 4.

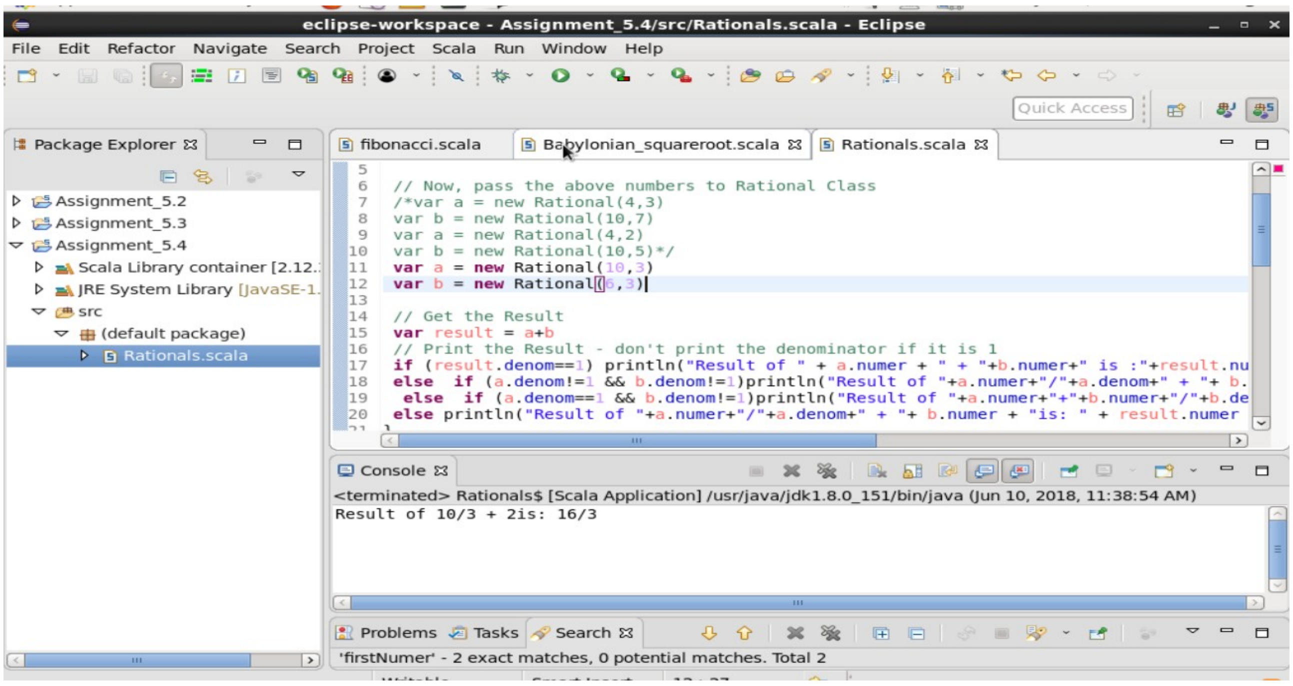


```scala
// Now, pass the above numbers to Rational Class
/*var a = new Rational(4,3)
var b = new Rational(10,7)*/
var a = new Rational(4,2)
var b = new Rational(10,5)
/*var a = new Rational(10,3)
var b = new Rational(6,3)*/

// Get the Result
var result = a+b
// Print the Result - don't print the denominator if it is 1
if (result.denom==1) println("Result of " + a.numer + " + "+b.numer+" is :"+result.nu
else  if (a.denom!=1 && b.denom!=1)println("Result of "+a.numer+"/"+a.denom+" + "+ b.
 else  if (a.denom==1 && b.denom!=1)println("Result of "+a.numer+"+"+b.numer+"/"+b.de
else println("Result of "+a.numer+"/"+a.denom+" + "+ b.numer + "is: " + result.numer
```

```
<terminated> Rationals$ [Scala Application] /usr/java/jdk1.8.0_151/bin/java (Jun 10, 2018, 11:37:52 AM)
Result of 2 + 2 is :4
```

```
'firstNumer' - 2 exact matches, 0 potential matches. Total 2
```

Here the input is one integer 2 and one rational 10/3 , we performed addition and we got the result as 16/3.



eclipse-workspace - Assignment_5.4/src/Rationals.scala - Eclipse

```scala

  // Now, pass the above numbers to Rational Class
  /*var a = new Rational(4,3)
  var b = new Rational(10,7)
  var a = new Rational(4,2)
  var b = new Rational(10,5)*/
  var a = new Rational(10,3)
  var b = new Rational(6,3)

  // Get the Result
  var result = a+b
  // Print the Result - don't print the denominator if it is 1
  if (result.denom==1) println("Result of " + a.numer + " + "+b.numer+" is :"+result.nu
  else  if (a.denom!=1 && b.denom!=1)println("Result of "+a.numer+"/"+a.denom+" + "+ b.
   else  if (a.denom==1 && b.denom!=1)println("Result of "+a.numer+"+"+b.numer+"/"+b.de
  else println("Result of "+a.numer+"/"+a.denom+" + "+ b.numer + "is: " + result.numer
```

```
<terminated> Rationals$ [Scala Application] /usr/java/jdk1.8.0_151/bin/java (Jun 10, 2018, 11:38:54 AM)
Result of 10/3 + 2is: 16/3
```

'firstNumer' - 2 exact matches, 0 potential matches. Total 2