Create a calculator to work with rational numbers.

Requirements:

○ It should provide capability to add, subtract, divide and multiply rational

numbers

***package*** *firstpackage*

***object*** *Rationals*

*{*

***def*** *main(args:Array[String]){*

***val*** *x=* ***new*** *Rational(1, 2) // 1/2*

***val*** *y =* ***new*** *Rational(2, 3) // 2/3*

*println("sum of rational numbers="+x.add(y))*

*println("multiplication of rational numbers="+x.mul(y))*

*println("subtraction of rational numbers="+x.sub(y))*

*println("division of rational numbers="+x.div(y))*

*}*

*}*

***class*** *Rational (x : Int, y : Int)*

*{*

***def*** *numer= x*

***def*** *denom= y*

***def*** *add (that : Rational) =* ***new*** *Rational (*

*numer \* that.denom + that.numer \* denom,*

*denom \* that.denom)*

***def*** *mul (that : Rational) =* ***new*** *Rational (*

*numer \* that.numer,*

*denom \* that.denom)*

***def*** *sub (that : Rational) =* ***new*** *Rational (*

*numer \* that.denom - that.numer \* denom,*

*denom \* that.denom)*

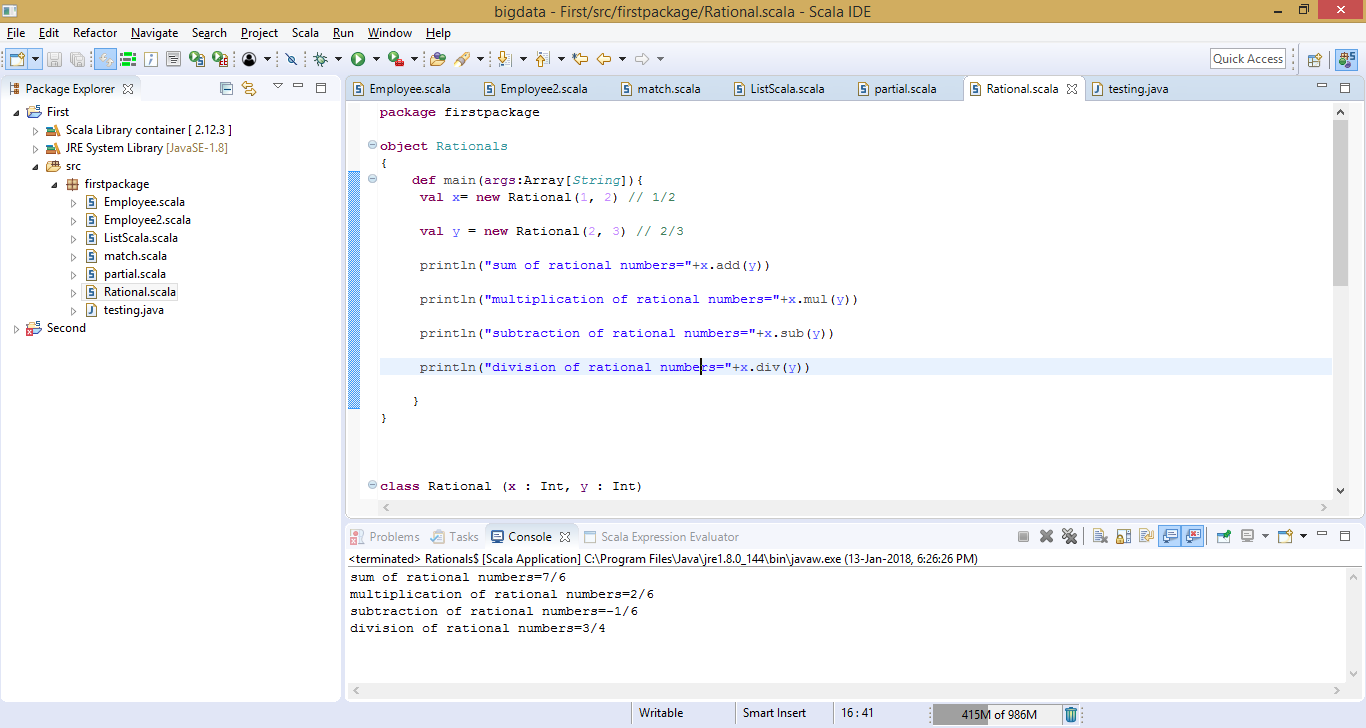
***def*** *div (that : Rational) =* ***new*** *Rational (*

*numer \* that.denom ,*

*denom \* that.numer)*

***override******def*** *toString = numer + "/" + denom*

*}*



○ Create a method to compute GCD (this will come in handy during operations on

rational)

***package*** *firstpackage*

***object*** *Rationals*

*{*

***def*** *main(args:Array[String]){*

***val*** *x=* ***new*** *Rational(2) // 1/2*

***val*** *y =* ***new*** *Rational(3) // 2/3*

*println("GCD of 9 and 15="+x.gcd(9,15))*

*println("sum of numbers="+x.add(y))*

*println("multiplication of numbers="+x.mul(y))*

*println("subtraction of numbers="+x.sub(y))*

*println("division of numbers="+x.div(y))*

*}*

*}*

***class*** *Rational (x : Int, y : Int)*

*{*

***def******this****( x : Int){*

***this****(x, 1)*

*println("From One-Argument Auxiliary Constructor")*

*}*

***def*** *numer= x*

***def*** *denom= y*

***def*** *gcd(a: Int, b: Int): Int =* ***if*** *(b == 0) a* ***else*** *gcd(b, a % b)*

***def*** *add (that : Rational) =* ***new*** *Rational (*

*numer \* that.denom + that.numer \* denom,*

*denom \* that.denom)*

***def*** *mul (that : Rational) =* ***new*** *Rational (*

*numer \* that.numer,*

*denom \* that.denom)*

***def*** *sub (that : Rational) =* ***new*** *Rational (*

*numer \* that.denom - that.numer \* denom,*

*denom \* that.denom)*

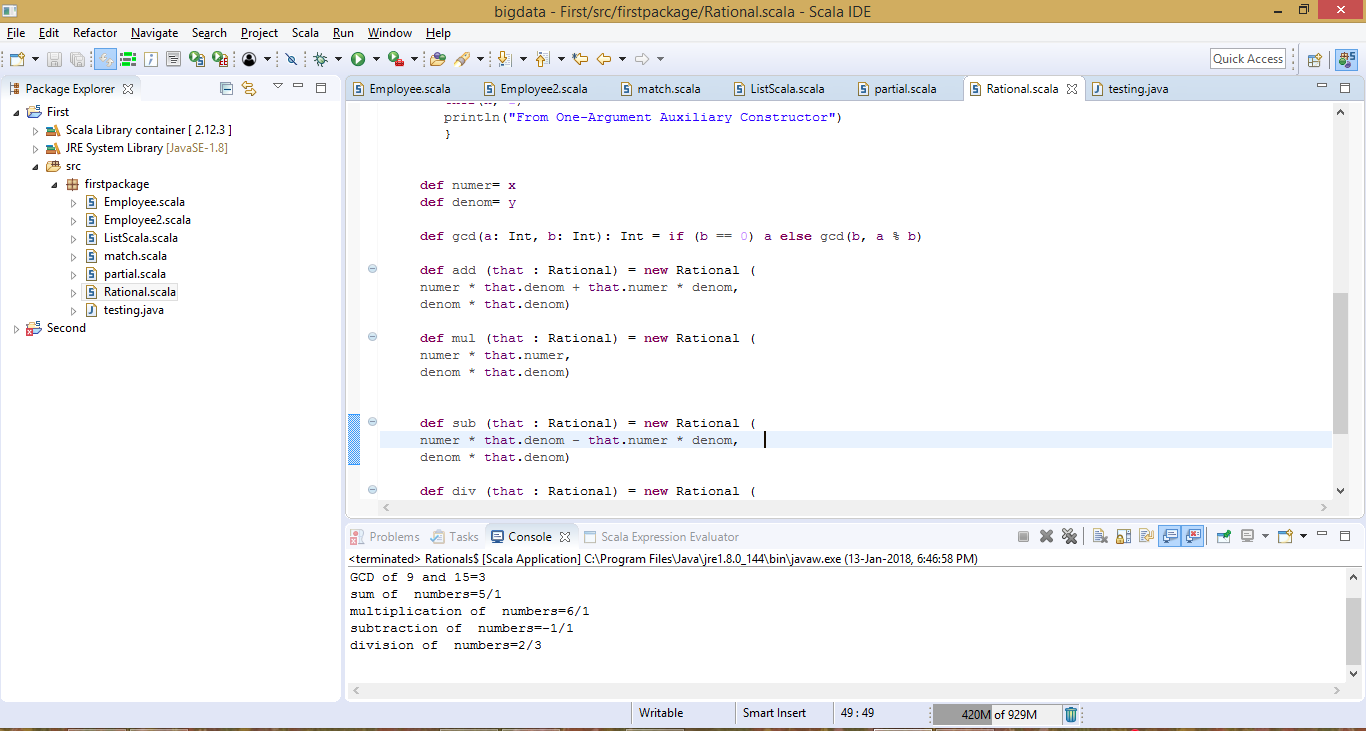
***def*** *div (that : Rational) =* ***new*** *Rational (*

*numer \* that.denom ,*

*denom \* that.numer)*

***override******def*** *toString = numer + "/" + denom*

*}*



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.

***package*** *firstpackage*

***object*** *Rationals*

*{*

***def*** *main(args:Array[String]){*

***val*** *x=* ***new*** *Rational(2) // 1/2*

***val*** *y =* ***new*** *Rational(3) // 2/3*

*println("sum of numbers="+x.add(y))*

*println("multiplication of numbers="+x.mul(y))*

*println("subtraction of numbers="+x.sub(y))*

*println("division of numbers="+x.div(y))*

*}*

*}*

***class*** *Rational (x : Int, y : Int)*

*{*

***def******this****( x : Int){*

***this****(x, 1)*

*println("From One-Argument Auxiliary Constructor")*

*}*

***def*** *numer= x*

***def*** *denom= y*

***def*** *add (that : Rational) =* ***new*** *Rational (*

*numer \* that.denom + that.numer \* denom,*

*denom \* that.denom)*

***def*** *mul (that : Rational) =* ***new*** *Rational (*

*numer \* that.numer,*

*denom \* that.denom)*

***def*** *sub (that : Rational) =* ***new*** *Rational (*

*numer \* that.denom - that.numer \* denom,*

*denom \* that.denom)*

***def*** *div (that : Rational) =* ***new*** *Rational (*

*numer \* that.denom ,*

*denom \* that.numer)*

***override******def*** *toString = numer + "/" + denom*

*}*

