

## Assignment16

### Problem Statement

#### Task 1

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

#### Scala Code:

➔ Create a Scala Class “Calc”

```
class Calc (n:Int, d:Int)
{
  required(d!=0)
  private val g = gcd(n.abs,d.abs)
  val num = n/g
  val den = d/g

  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)}
```

```
def this(n: Int) = this(n,1) // auxiliary constructor
```

```
def add (r:Calc): Calc = new Calc(num*r.den + r.num*den , den*r.den)
```

```
def add (i:Int): Calc = new Calc(num - i * den, den) //method overloading for add
```

```
def subtract (r:Calc): Calc = new Calc(num *r.den - r.num*den,den*r.den)
```

```
def subtract (i:Int): Calc = new Calc(num - i * den, den) //method overloading for  
subtract
```

```
def multiply (r:Calc): Calc = new Calc(num*r.num,den*r.den)
```

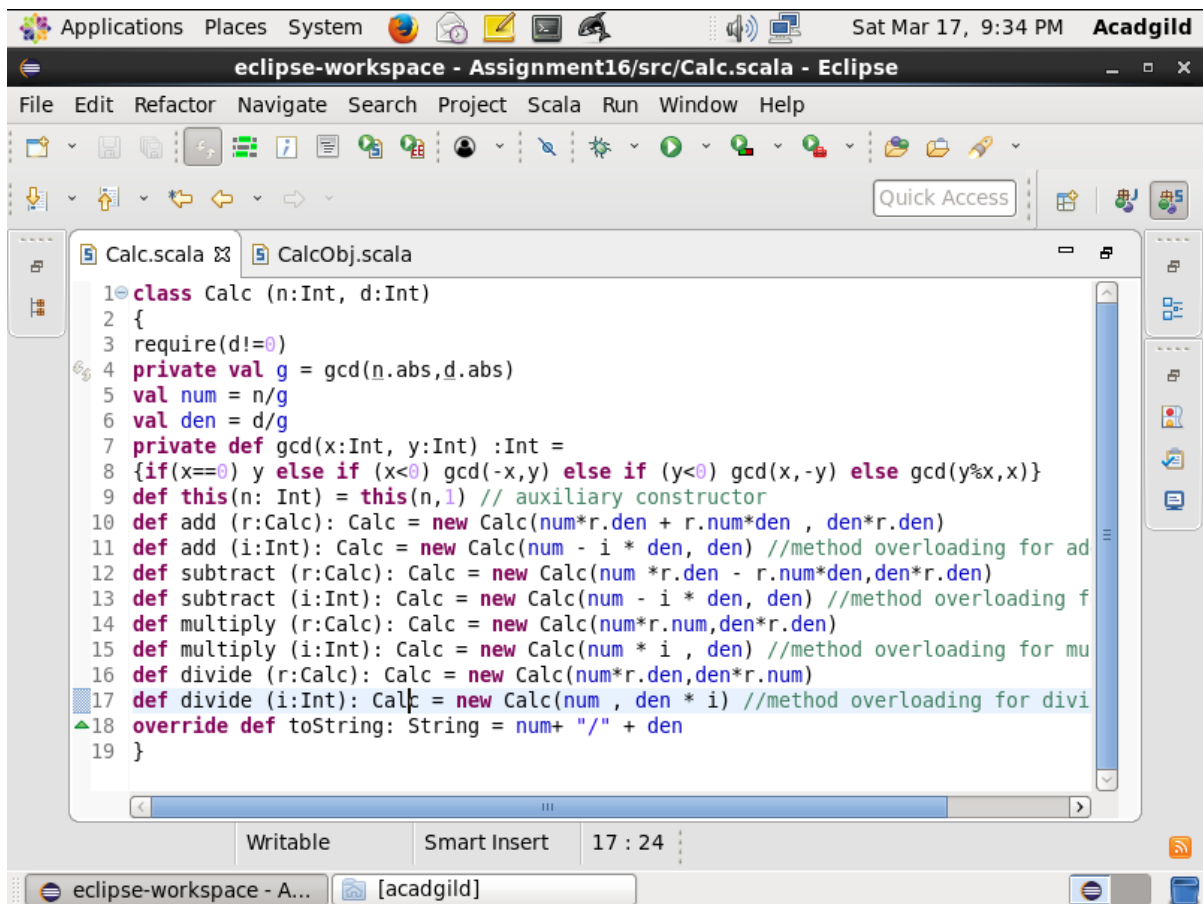
```
def multiply (i:Int): Calc = new Calc(num * i , den) //method overloading for  
multiplication
```

```
def divide (r:Calc): Clac = new Calc(num*r.den,den*r.num)
```

```
def divide (i:Int): Calc = new Calc(num , den * i) //method overloading for division
```

```
override def toString: String = num+ "/" + den
```

```
}
```



➔ Create Scala object as CalcObj

object CalcObj

{

def main(args: Array[String])

{

val a = new Calc(20)

val b = new Calc(10)

val c = new Calc(5)

val d = new Calc(200)

val p = a add 5

println Addition:(p)

val q = b multiply new Calc(11,21)

```
println Multiplication(q)

val r = c subtract new Calc(13,1)

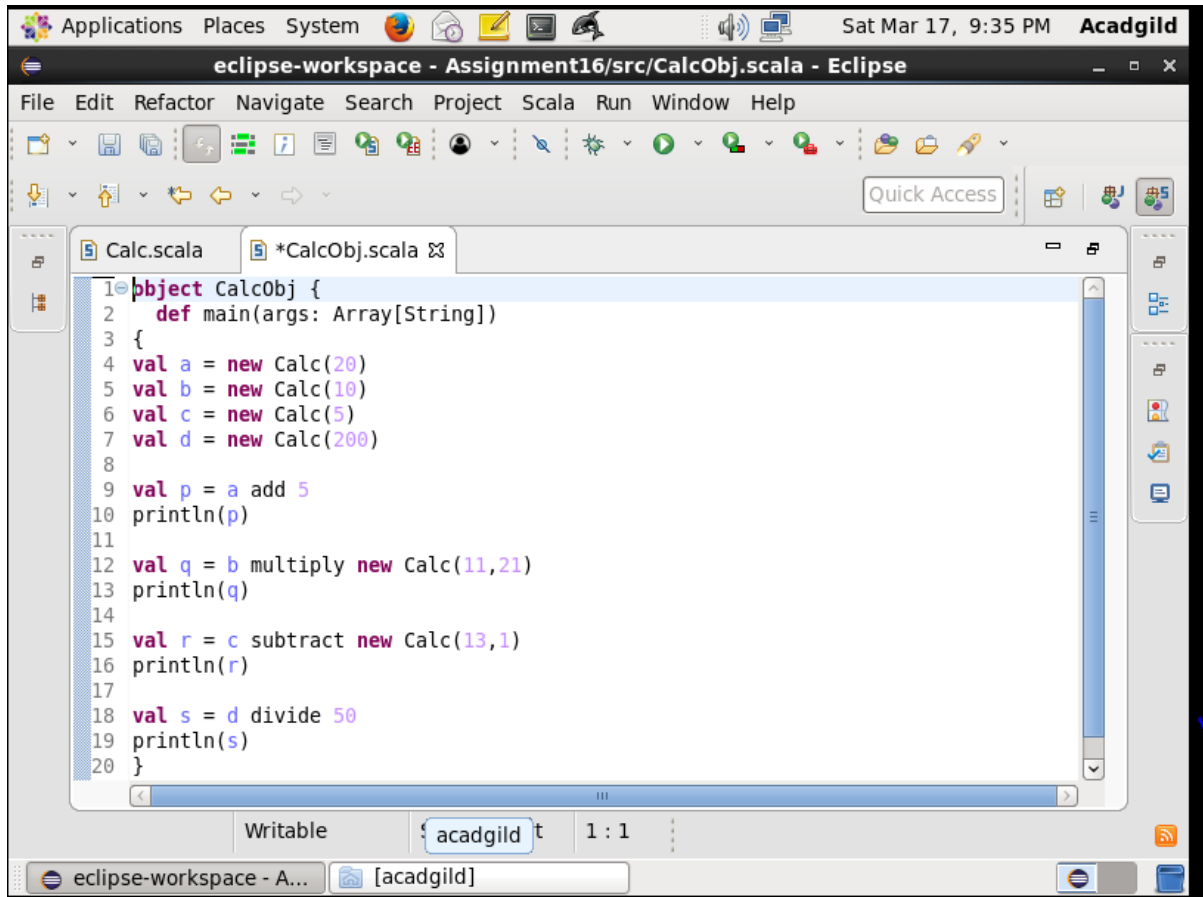
println Subraction(r)

val s = d divide 50

println division(s)

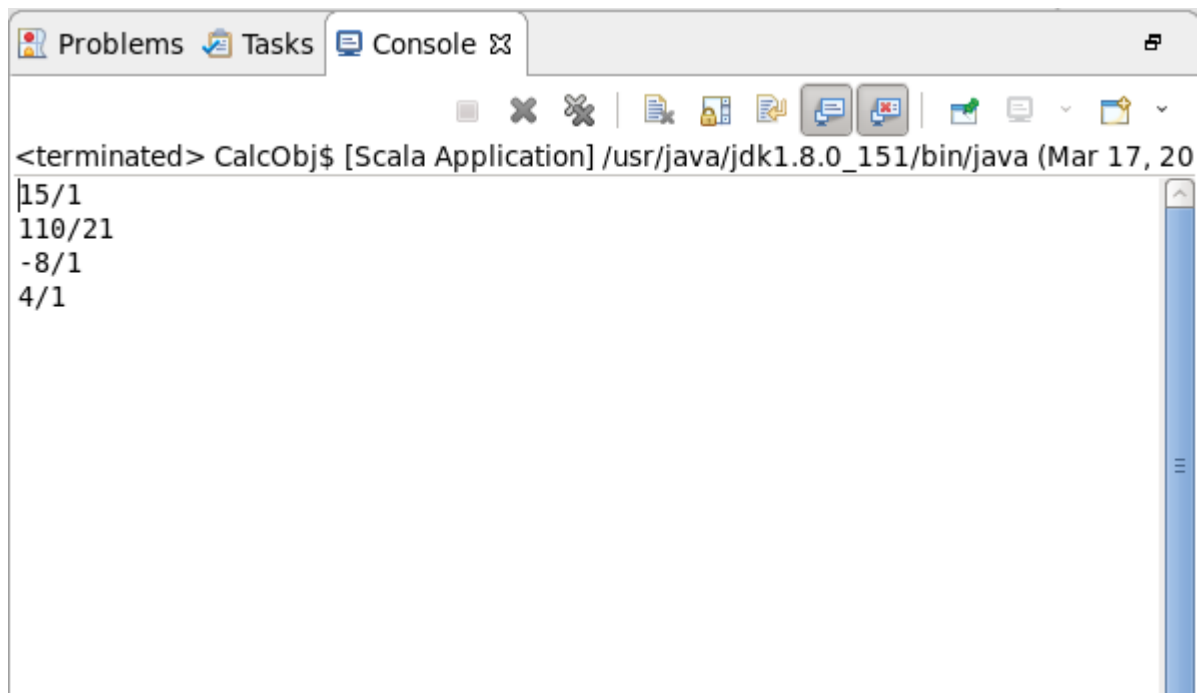
}

}
```



```
1 object CalcObj {
2   def main(args: Array[String])
3   {
4     val a = new Calc(20)
5     val b = new Calc(10)
6     val c = new Calc(5)
7     val d = new Calc(200)
8
9     val p = a add 5
10    println(p)
11
12    val q = b multiply new Calc(11,21)
13    println(q)
14
15    val r = c subtract new Calc(13,1)
16    println(r)
17
18    val s = d divide 50
19    println(s)
20  }
```

## Output:



The screenshot shows an IDE's console window. The title bar includes tabs for 'Problems', 'Tasks', and 'Console'. The console text shows a terminated Scala application running on Java 1.8.0\_151, with the following output:

```
<terminated> CalcObj$ [Scala Application] /usr/java/jdk1.8.0_151/bin/java (Mar 17, 20
15/1
110/21
-8/1
4/1
```