**Task 1 – Create a Scala Class *“Calc”***

class Calc (n:Int, d:Int)

{

require(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): Calc = 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

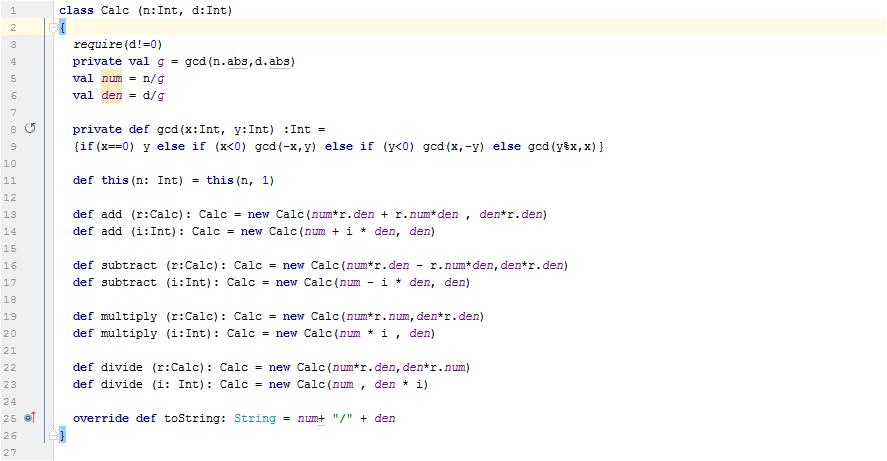
}



The statement, ***“def this(n: Int) = this(n, 1) “*** is an auxiliary constructor, we have created an Object **“CalcObj”** to perform the above functions.

We have Enabled method **overloading** to enable each function (add, sub, multiplication and division) to work with numbers and rational.

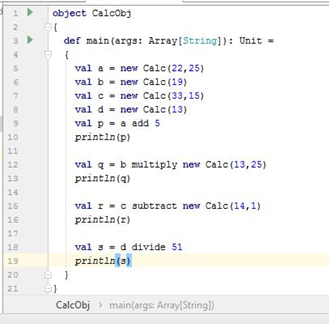
We have written the code in such a way that it works with whole numbers as well as with rational numbers (n/1).



**Task 2 – Create a Scala Object “CalObj**

object CalcObj {  
 def main(args: Array[String]): Unit =  
 {  
 val a = new Calc(22,25)  
 val b = new Calc(19)  
 val c = new Calc(33,15)  
 val d = new Calc(13)  
 val p = a add 5  
 println(p)  
 val q = b multiply new Calc(13,25)  
 println(q)  
 val r = c subtract new Calc(14,1)  
 println(r)  
 val s = d divide 51  
 println(s)  
 }  
}

Screenshot



Results

