Session 15 - Scala Session - II

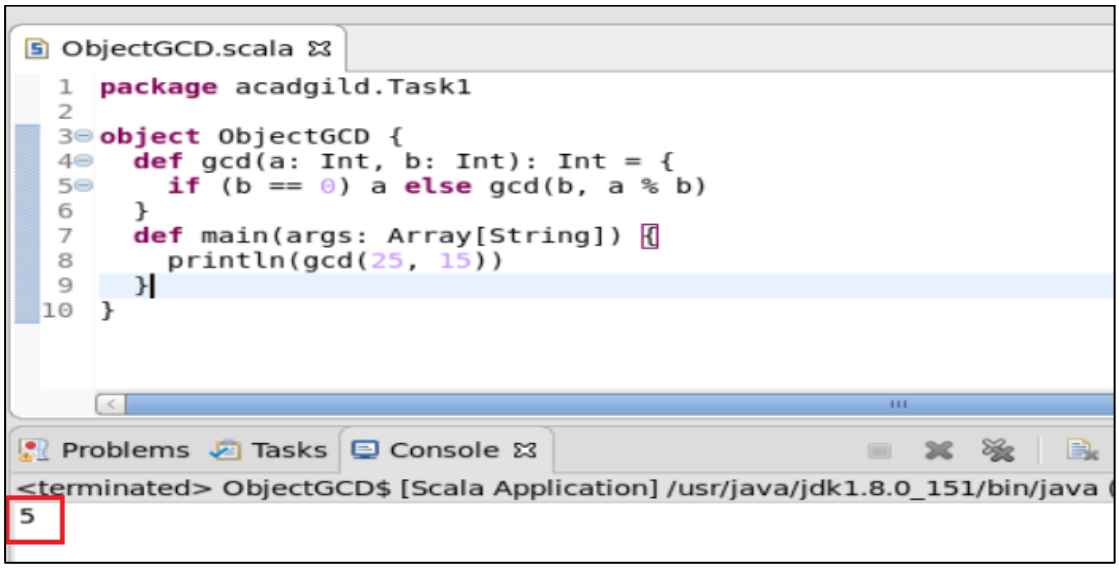
Assignment 1

I created a Scala application(project) in Eclipse and created 3 separate packages for each of the tasks.

Task 1:

Create a Scala application to find the GCD of two numbers.

```
object ObjectGCD {
  def gcd(a: Int, b: Int): Int = {
   if (b == 0) a else gcd(b, a % b)
  }
  def main(args: Array[String]) {
   println(gcd(25, 15))
  }
}
```



The output clearly shows the GCD of 25 and 5 that is 5.

Task 2:

Fibonacci series (starting from 1) written in order without any spaces in between, thus producing a sequence of digits.

Write a Scala application to find the Nth digit in the sequence.

➢Write the function using standard for loop

➢Write the function using recursion

Printing Fibonacci series in orde without spaces

```
object FibonacciSequence extends App {
  def fib(n: Int) = {
    var a = 1
    var b = 1
    var i = 2
    print(a + "" + b)
    while (i < n) {
      val c = a + b
      print(c)
      a = b
      b = c
      i = i + 1
    }
  }
  print(fib(10))
}
```

FibonacciSequence.scala

```scala
package acadgild.Task2

object FibonacciSequence extends App {

  def fib(n: Int) = {
    var a = 1
    var b = 1
    var i = 2
    print(a + "" + b)
    while (i < n) {
      val c = a + b
      print(c)
      a = b
      b = c
      i = i + 1
    }
  }
  print(fib(10))
}
```

Problems Tasks Console

```
<terminated> FibonacciSequence$ [Scala Application] /usr/java/
11235813213455
```

The function takes in an integer n, thus printing the first n numbers from the fibonacci sequence. Here, n =10 and the first 10 values I.e,

1,1,2,3,5,8,13,21,34,55

are printed without any spaces.

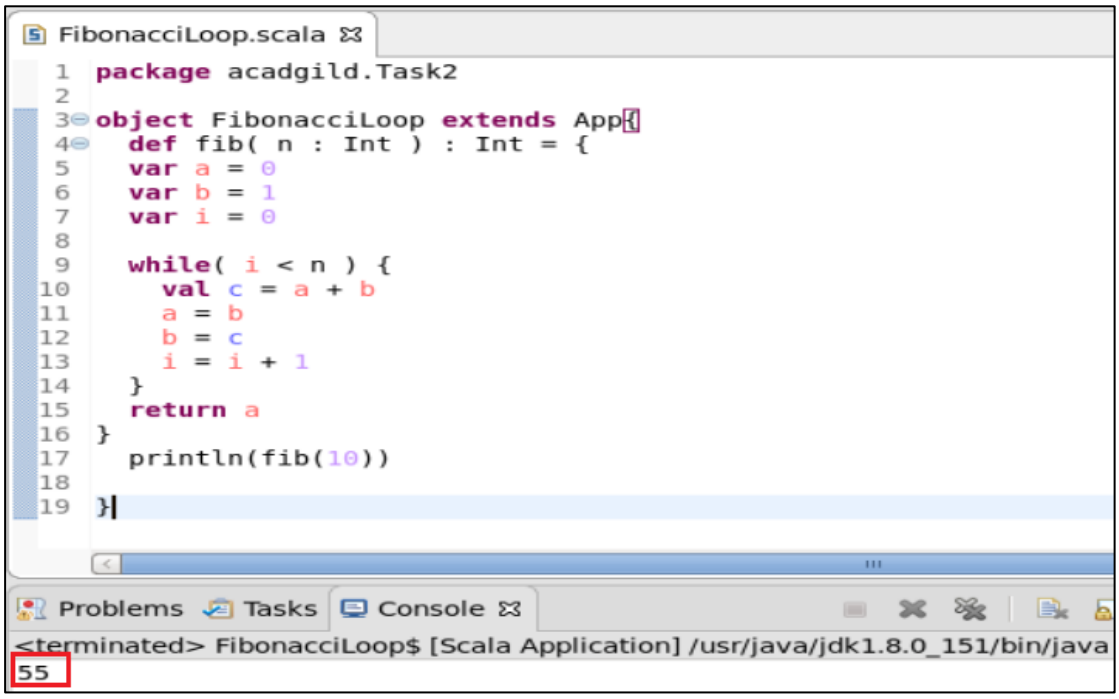
Write the function using standard for loop

```scala
object FibonacciLoop extends App{
  def fib( n : Int ) : Int = {
  var a = 0
  var b = 1
  var i = 0

  while( i < n ) {
   val c = a + b
   a = b
   b = c
   i = i + 1
```

```
    }
    return a
  }
    println(fib(10))

  }
```



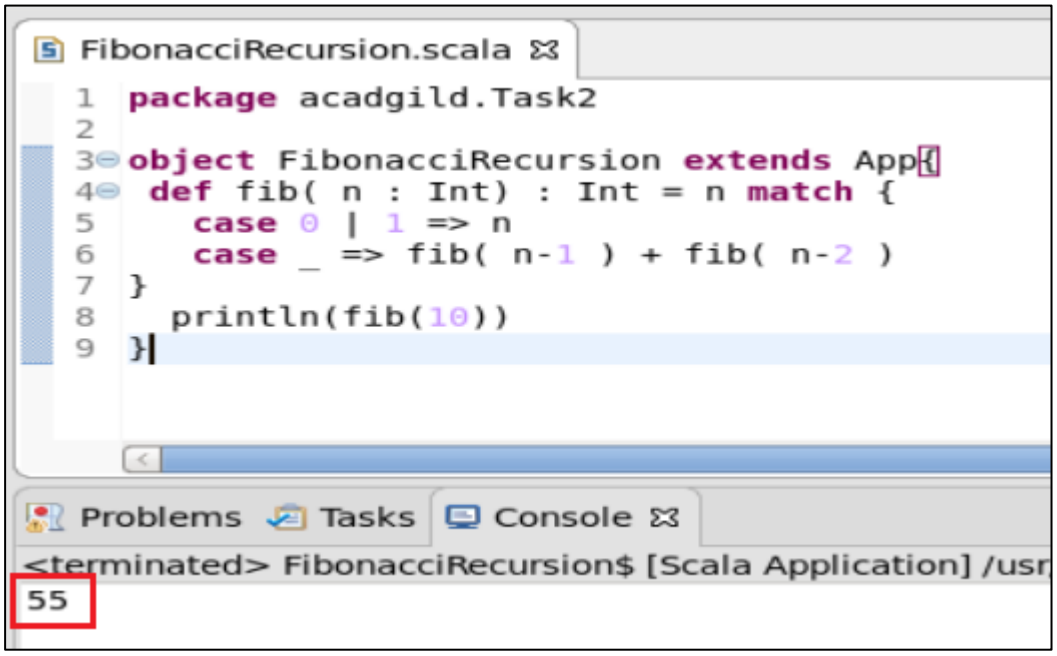
The function takes in an integer n, thus printing the nth number from the fibonacci sequence. Here, n =10 and the 10th value(starting from 1) I.e,

1,1,2,3,5,8,13,21,34,**55**

55 is printed.

Write the function using recursion

```
object FibonacciRecursion extends App{
 def fib( n : Int) : Int = n match {
   case 0 | 1 => n
   case _ => fib( n-1 ) + fib( n-2 )
}
  println(fib(10))
}
```



The function takes in an integer n, thus printing the nth number from the fibonacci sequence. Here, n =10 and the 10th value(starting from 1) I.e,

1,1,2,3,5,8,13,21,34,**55**

55 is printed.

The function 'fib' is recursively called by itself multiple times untill the value of n is either 0 or 1.

Task 3:

Find square root of number using Babylonian method.

1. Start with an arbitrary positive start value x (the closer to the root, the better).

2.Initialize y = 1.

3. Do following until desired approximation is achieved.

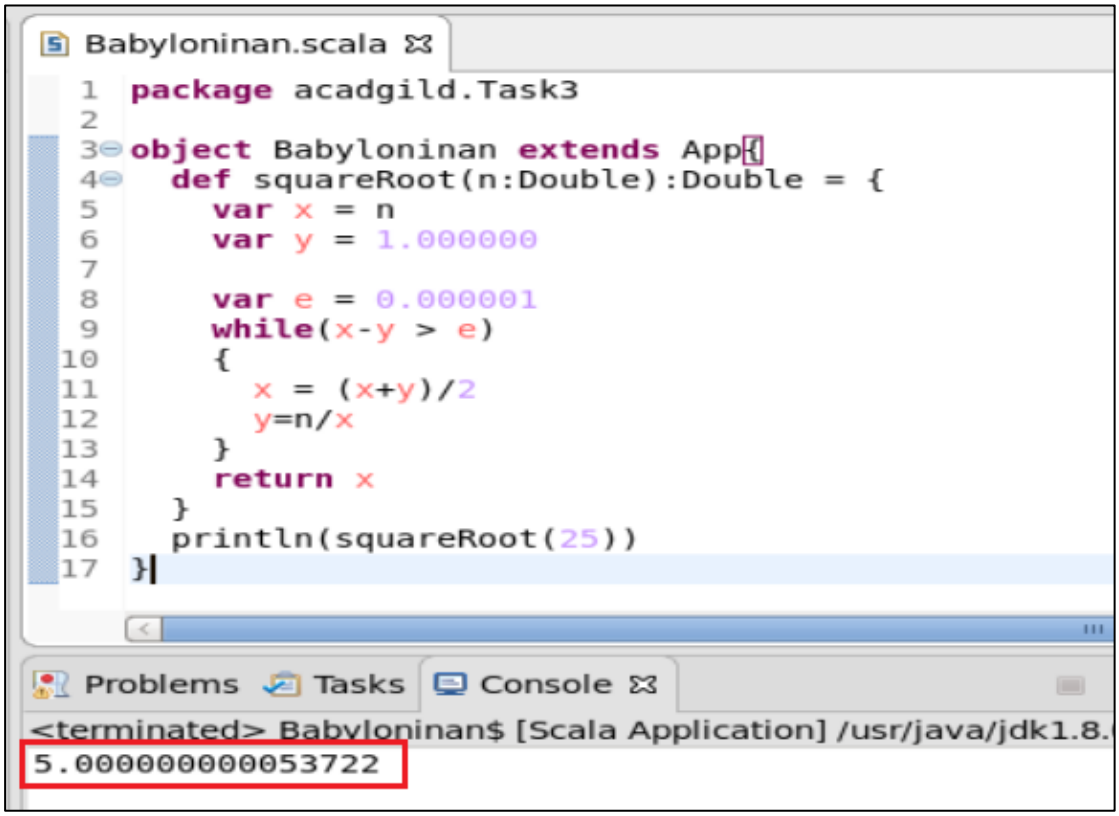
a) Get the next approximation for root using average of x and y

b) Set y = n/x

```
object Babyloninan extends App{
  def squareRoot(n:Double):Double = {
```

```
    var x = n
    var y = 1.000000

    var e = 0.000001
    while(x-y > e)
    {
      x = (x+y)/2
      y=n/x
    }
    return x
  }
  println(squareRoot(25))
}
```



The function takes in an Double value n whose square root has to be found out. Here n is 25 and the square root of 25 is found out to be 5.000000000053722 using Babylonian Method to find out square root.