Assignment 15: Scala Basics 2 Assignment Problems

Problem Statement

Task 1

Create a Scala application to find the GCD of two numbers

Terminal Execution :

```
[acadgild@localhost ~]$ scala
Welcome to Scala version 2.11.7 (Java HotSpot(TM) Client VM, Java 1.8.0_171).
Type in expressions to have them evaluated.
Type :help for more information.

scala> def gcm(a:Int,b:Int):Int = {
     | if(b == 0) a
     | else gcm(b,a%b)
     | }
gcm: (a: Int, b: Int)Int

scala> val gcm_no = gcm(15,80)
gcm_no: Int = 5

scala> println("The gcm of 2 nos are :"+ gcm(20,50))
The gcm of 2 nos are :10
```

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**

Terminal Execution :

```
scala> def fibonacci(num:Int):Int={
     | var a = 1
     | var b = 2
     | var c = 0
     | for(i <- 2 until num){
     | c = a + b
     | a=b
     | b=c
     | }
     | c
     | }
fibonacci: (num: Int)Int

scala> val fibon = fibonacci(4)
fibon: Int = 5
```

➢ **Write the function using recursion**

Terminal Execution :

```
scala>  def fibonaci(n:Int)=
     | {
     | def fibgo(n:Int,prev:Int=1,next:Int=2):Int= n-1 match
     | {
     | case 0 => prev
     | case 1 => next
     | case _ => fibgo(n-1,next,(next+prev))
     | }
     | fibgo(n)
     | }
fibonaci: (n: Int)Int

scala> val fibo_recur = fibonaci(4)
fibo_recur: Int = 5
```

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

Terminal Execution :

```
scala> def squareRoot(n: Int)= {
     | var x = n
     | var y = 1
     | def babyroot(x :BigDecimal,y :BigDecimal) : BigDecimal = {
     | if( x - y <= 0.00001)
     | x
     | else
     | babyroot(((x+y)/2), (n/x))
     | }
     | babyroot(x,y)
     | }
squareRoot: (n: Int)BigDecimal

scala> val root = squareRoot(2)
root: BigDecimal = 1.25
```