Session 15: Scala II Assignment 1

Task 1

Create a Scala application to find the GCD of two numbers

Using Tail Recursion as explained in ProgrammingInScala.pdf

```
[acadgild@localhost Session15-Scala2]$ scala
Welcome to Scala 2.12.4 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_151).
Type in expressions for evaluation. Or try :help.

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

scala> gcd(48,8)
res0: Int = 8

scala> gcd(200,100)
res1: Int = 100

scala> gcd(324,872)
res2: Int = 4

scala>
```

Task 2

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

```
object Fibon {
  def main(args: Array[String]): Unit = {
    var next: Int = 0
    var first: Int = 0
    var second: Int = 1

    for( i <- 0 to 20) {
        if (i <= 1)
          next = i
        else {
          next = first + second
          first = second
          second = next
        }
        print(next)
    }
    println("\n")
  }
}
~
~
~
~
"Fibon.scala" 19L, 321C written
```

```
ala2]$ scalac Fibon.scala
[acadgild@localhost Session15-Scala2]$ scala Fibon
0112358132134558914423337761098715972584418167 65

You have new mail in /var/spool/mail/acadgild
[acadgild@localhost Session15-Scala2]$
```

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

➢ Write the function using standard for loop

```
object FibonLoop {
  def main(args: Array[String]): Unit = {
    var next: Int = 0
    var first: Int = 0
    var second: Int = 1

    var flist: List[Int] = List()

    for( i <- 0 to 20) {
        if (i <= 1)
          next = i
        else {
          next = first + second
          first = second
          second = next
        }
        print(next)
        flist = flist:+next

    }
    println("\n")
    println(flist)

    println("The 12th element in the list:")
    println(flist(12))
  }
}
~
~
```

```
[acadgild@localhost Session15-Scala2]$ scalac FibonLoop.scala
[acadgild@localhost Session15-Scala2]$ scala FibonLoop
011235813213455891442333776109871597258441816765

List(0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 258
4, 4181, 6765)
The 12th element in the list:
144
[acadgild@localhost Session15-Scala2]$
```

➢ Write the function using recursion

```
object FibonRecur {
  def myRecurr(iterator: Int, value: Int, target: Int) {
     if (iterator == target)
        println(value)
  }

  def main(args: Array[String]): Unit = {
    var next: Int = 0
    var first: Int = 0
    var second: Int = 1

    var cycle: Int = 0

    var flist: List[Int] = List()

    for( i <- 0 to 20) {
        if (i <= 1)
          next = i
        else {
          next = first + second
          first = second
          second = next
        }
        print(next)
        flist = flist:+next

    }
    println("\n")
    println(flist)

    println("Showing 9th element using recursion")

    for( a <- flist) {
        myRecurr(cycle, a, 9)
        cycle= cycle + 1
    }

  }
}


~
~
~
~
"FibonRecur.scala" 40L, 676C written                                    35,
```

```
[acadgild@localhost Session15-Scala2]$ scala FibonRecur
01123581321345589144233377610987159725844181676 5

List(0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 258
4, 4181, 6765)
Showing 9th element using recursion
34
[acadgild@localhost Session15-Scala2]$
```

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

```
 1 object BabyRoot {
 2   def Root(root: Float) {
 3     var a: Float = root
 4     var b: Float = 1
 5     var c: Double = 0.0001
 6
 7     while (a-b > c) {
 8         a = (a + b) /2
 9         b = root/a
10     }
11     println("The squareroot of " + root + " is " + a)
12 }
13
14   def main(args: Array[String]): Unit = {
15   Root(49)
16   Root(63)
17   Root(100)
18   Root(873)
19   Root(160000)
20
21   }
22 }
23
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"BabyRoot.scala" 23L, 349C written
```

```
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost Session15-Scala2]$ scalac BabyRoot.scala
[acadgild@localhost Session15-Scala2]$ scala BabyRoot
The squareroot of 49.0 is 7.0
The squareroot of 63.0 is 7.9372554
The squareroot of 100.0 is 10.0
The squareroot of 873.0 is 29.546576
The squareroot of 160000.0 is 400.0
[acadgild@localhost Session15-Scala2]$
```