Assignment-15

Task 1:

Create a Scala application to find the GCD of two numbers

The below screenshot shows the code for GCD of two numbers.

```
build.sbt   GCD.scala
package corescala

object GCD {

  def gcd(x: Int,y: Int): Int = {
    if(y ==0) x else gcd(y, x%y)
  }

  def main(args: Array[String]) : Unit {
    println(gcd(4,5))
  }
}
```

The output of the above code is as shown in the below screenshot.

```
GCD
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
1

Process finished with exit code 0
```

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**

The scala code for the Fibonacci series using loop is shown in the below screenshot.

```
build.sbt   Fib_loop.scala

package corescala

object Fib_loop {
  def fib( n : Int ) : Int = {
    var a = 1
    var b = 2
    var i = 1
    println(a)

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

    return a
  }

  def main(args: Array[String]) : Unit {
    var nth_element = 0
    println("Fibonacci series with total " + args(0) + " elements is (starting from 1) : ")
    nth_element = fib(args(0).toInt)
    println("element " + args(0) + " in fibonacci series is : " + nth_element)
  }
}
```

The output for the Fibonacci series using loop is shown in the below screenshot.

```
Fib_loop
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
Fibonacci series with total 10 elements is (starting from 1) :
1
2
3
5
8
13
21
34
55
89
element 10 in fibonacci series is : 89

Process finished with exit code 0
```

- **Write the function using recursion**

The scala code for Fibonacci series using recursion is as shown in the below screenshot.

```
build.sbt   Fib_recurse.scala
package corescala

object Fib_recurse {
  def fib( n : Int) : Int = n match {
    case 1 | 2 => n
    case _ => fib(n-1) + fib(n-2)
  }

  def main(args: Array[String]) :Unit {
    println("Element " + args(0) + " in fibonacci series is (starting from 1): " + fib(args(0).toInt))
  }
}
```

The output for the Fibonacci series using recursion is as shown in the below screenshot.

```
Fib_recurse
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
Element 10 in fibonacci series is (starting from 1): 89

Process finished with exit code 0
```

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

The code for Babylonian method to find square root of a number is as shown in the below screenshot.

build.sbt × Babylonian_sqrt.scala ×

```scala
package corescala

object Babylonian_sqrt {

  def sqrt( number : Double) : Double = {

    var y : Double = 1
    var x : Double = number
    val error : Double = 0.000001

    while ((x - y)> error) {

      x = (x + y)/2
      y = number/x
    }

    return x
  }

  def main(args: Array[String]) :Unit = {

    var n : Double = 0
    println("Given Number :" +args(0))
    n = sqrt(args(0).toDouble)
    println("The square root of the number is :" +n)
  }

}
```

The output for the Babylonian method is as shown in the below figure.

```
Babylonian_sqrt
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
Given Number :65
The square root of the number is :8.062257748298771

Process finished with exit code 0
```