

Task 1

Create a Scala application to find the GCD of two numbers

a Stanley\IdeaProjects\ScalaTutorial] - ...src\main\scala\Assignment15.sc [scalatutorial] - IntelliJ IDEA

```
Analyze Refactor Build Run Tools VCS Window Help
in > scala > Assignment15.sc >
Assignment15.sc x
1 def gcd(a:Int, b:Int):Int = {
2   if(a==b) a
3   if(b==0) a else gcd(b, a%b)
4 }
5
6 gcd( 50, 10)
7 gcd( 10,10)
8 gcd( 0, 5 )
9
```

```
1 gcd: gcd[](val a: Int, val b: Int) => Int
2
3
4
5
6 res0: Int = 10
7 res1: Int = 10
8 res2: Int = 5
9
```

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

```
26
27 def fibonacciil( n:Int ): Int={
28
29   var n1 = 0
30   var n2 = 1
31   var i = 0
32
33   for( i <- 0 to n if i < n ){
34     val c = n1 + n2
35     n1 = n2
36     n2 = c
37   }
38   return n1
39 }
40
41
42
43 fibonacciil( 10 )
44 fibonacciil( 20 )
45
```

```
26 fibonacciil: fibonacciil[](val n: Int) => Int
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43 res3: Int = 55
44 res4: Int = 6765
45
```

➤ Write the function using recursion

```
41
42 def fibonacci2 ( n: Int ): Int = {
43   def recursion( n: Int, a: Int, b: Int): Int = n match{
44     case 0 => a
45     case _ => recursion( n-1, b, a+b)
46   }
47   return( recursion(n, 0, 1))
48 }
49 fibonacci2( 10 )
50 fibonacci2( 20 )
51
```

```
41 fibonacci2: fibonacci2[](val n: Int) => Int
42
43
44
45
46
47
48
49 res3: Int = 55
50 res4: Int = 6765
51
```

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 $y = n/x$

[illegible]

Output:

```
SR: SR[](val n: BigDecimal) => Stream[BigDecimal]
```

```
res3: Stream[BigDecimal] = Stream(1, ?)
```

```
iterations: Int = 10
```

```
res4: BigDecimal = 9.00000000000000000000000000000000
```

```
res5: List[BigDecimal] = List(1, 41.0, 21.487804878048780487804878049,  
12.62869245037512804185930622075801, 9.521329066772004602215646432912058,  
9.014272376994607516886591168570482, 9.000011298790216061592675931729374,  
9.0000000000007092361115353800790776, 9.000000000000000000000002794532566,  
9.000000000000000000000000000000)
```