Assignment15

Problem Statement

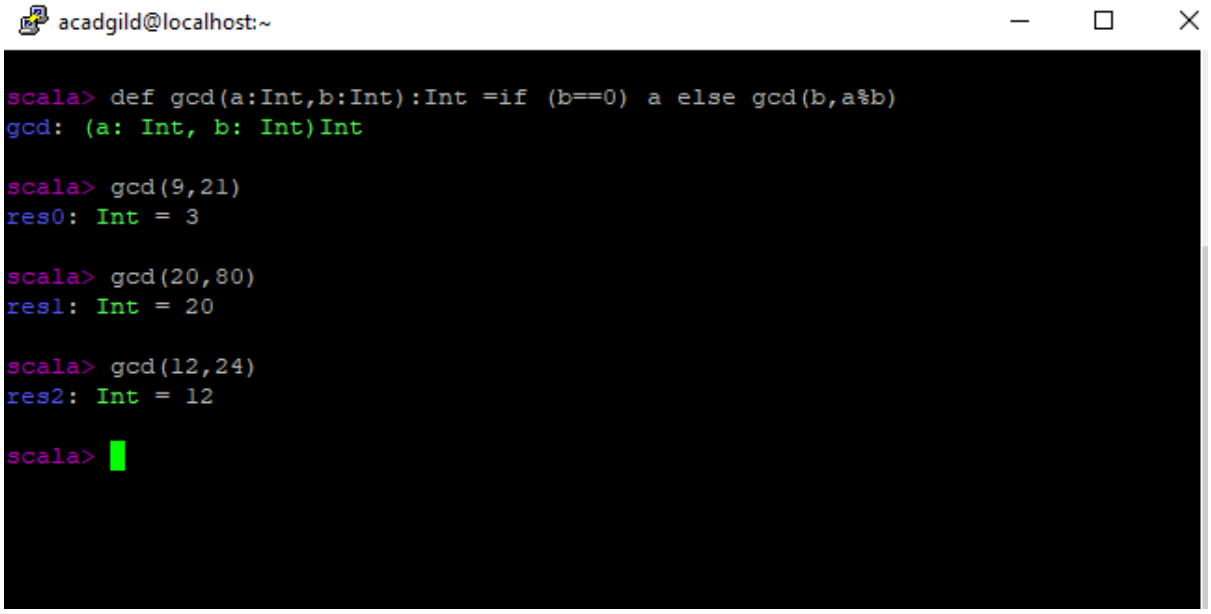
Task 1

Create a Scala application to find the GCD of two numbers

➔ Command : def gcd(a:Int,b:Int):Int =if (b==0) a else gcd(b,a%b)

To get the output

➔ Command : gcd(9,21)



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.

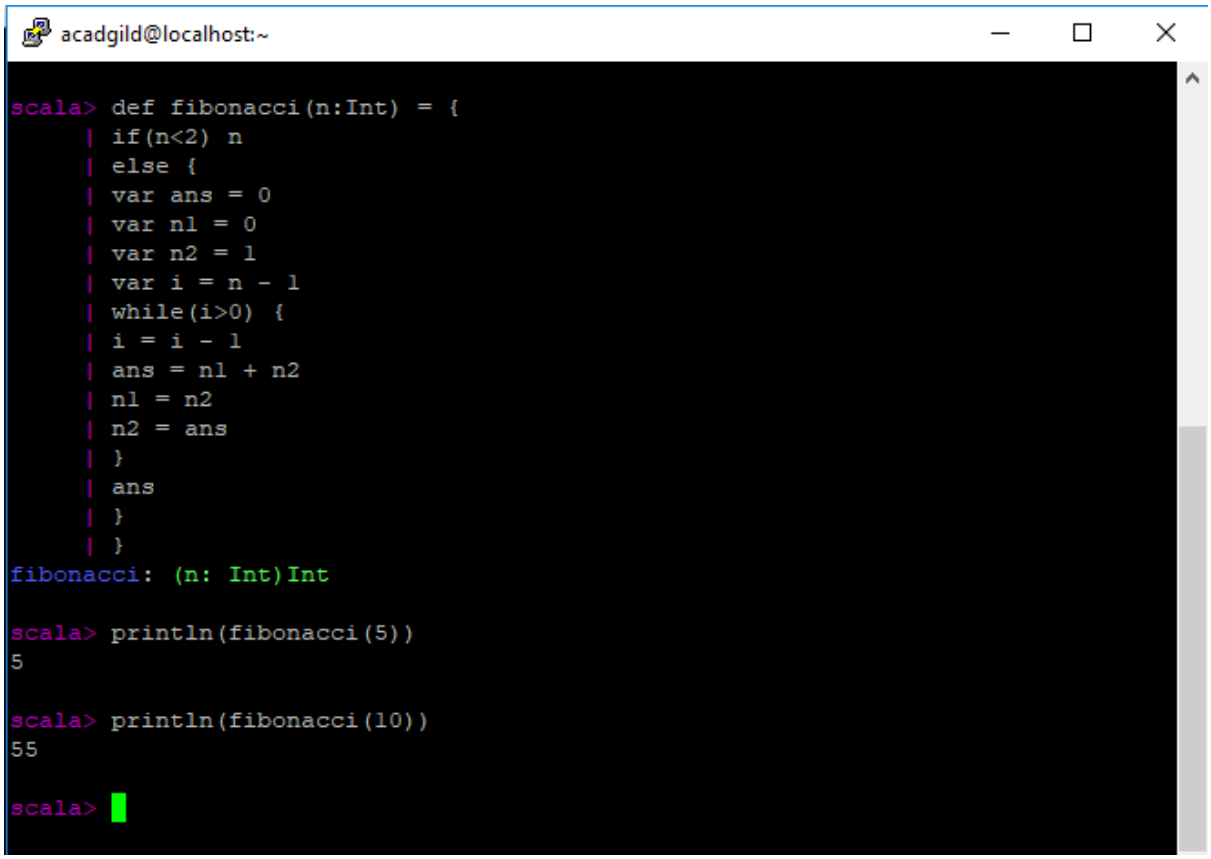
➢ Write the function using standard for loop

➔ Command: def Fibonacci (n:Int) = {

```
If(n<2) n
Else {
Var ans = 0
Var n1 = 0
Var n2 = 1
Var I = n – 1
While(i>0) {
i = i – 1
ans = n1 + n2
n1 = n2
n2 = ans
}
ans
}
}
```

To print the output

➔ Command : println(fibonacci(5))

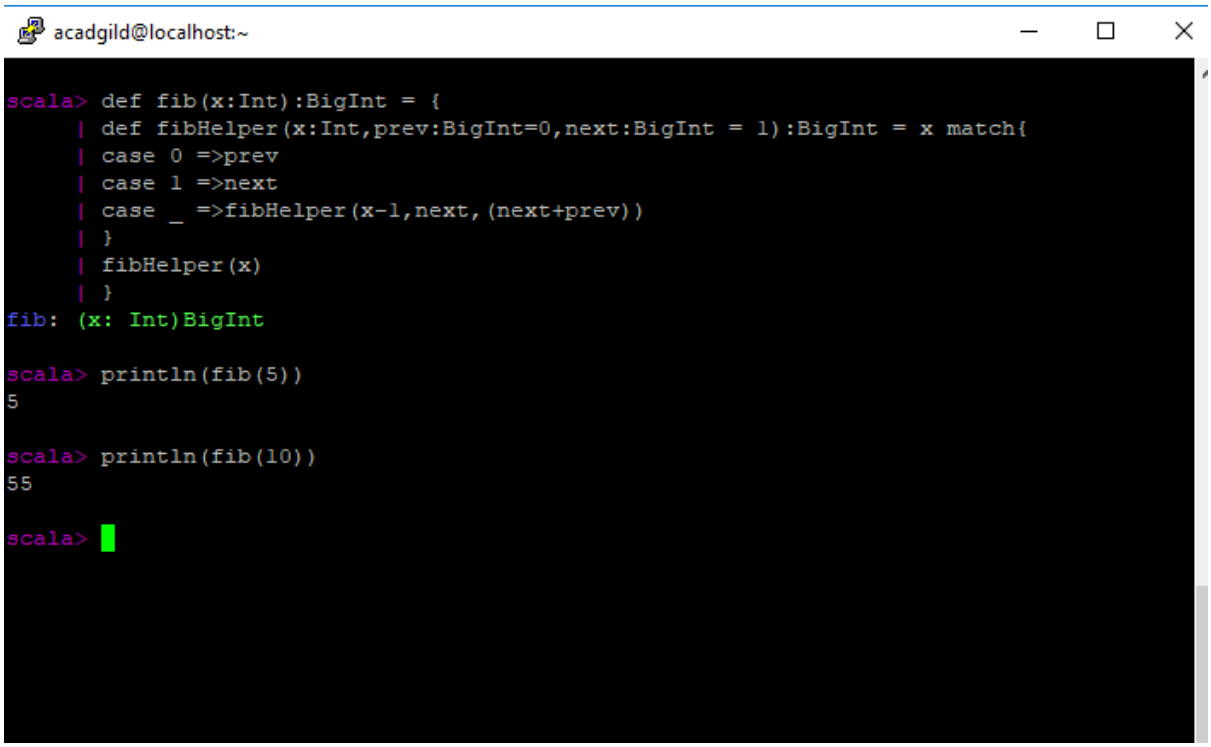


➢ Write the function using recursion

➔ Command : def fib(x:Int):BigInt = {
Def fibHelper(x:Int,prev:BigInt=0,next:B igInt = 1):BigInt = x match{
case 0 =>prev
case 1 =>next
case _ =>fibHelper(x-1,next,(next+prev))
}
fibHelper(x)
}

To print the output

➔ Command : println(fib(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

➔ Command : def sqrt(a:Double) = {
Val acc = 1e-10
Def findRoot(x:Double):Dou ble = {
Val nextx = (a/x + x)/2
If((x-nextx).abs<acc*x) nextx else findRoot(nextx)
}
findRoot(1)
}

To print the Output

➔ Command : println(sqrt(121))

acadgild@localhost:~

```
scala> def sqrt(a:Double) = {
     | val acc = 1e-10
     | def findRoot(x:Double):Double = {
     | val nextx = (a/x + x)/2
     | if((x-nextx).abs<acc*x) nextx else findRoot(nextx)
     | }
     | findRoot(1)
     | }
sqrt: (a: Double)Double

scala> println(sqrt(121))
11.0

scala> println(sqrt(81))
9.0

scala>
```