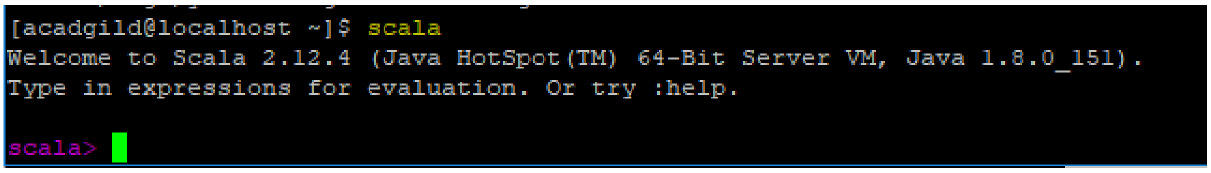
Big data and Hadoop – Assignment 5_Scala Basics

Task 1

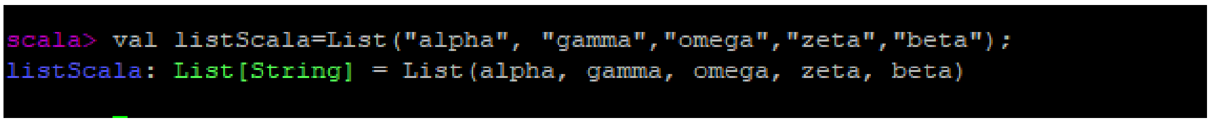
Given a list of strings - List[String] (“alpha”, “gamma”, “omega”, “zeta”, “beta”)

Step 1: open command prompt and simply type Scala. After that, you will see new Scala prompt waiting for your input as shown below



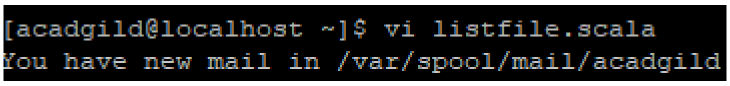
```
[acadgild@localhost ~]$ scala
Welcome to Scala 2.12.4 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_151).
Type in expressions for evaluation. Or try :help.

scala>
```

- List
    - Immutable sequence of objects of same type can be represented using lists
    - A List[Int] contains only integers.
    - Scala Lists are always immutable (whereas Java Lists can be mutable).
    - While creating a list we can define the data collection getting into that list o Val lstData : List[DataType] = List (Collection_of_data)



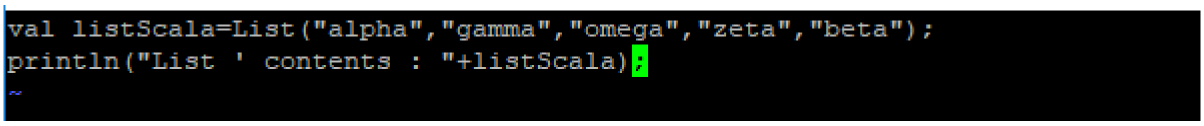
```
scala> val listScala=List("alpha", "gamma","omega","zeta","beta");
listScala: List[String] = List(alpha, gamma, omega, zeta, beta)
```

- Can save Scala code in the file with .scala extension and to run, providing file name with extension as parameter to Scala interpreter

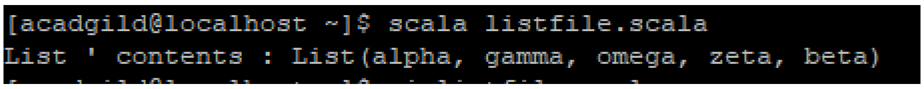


```
[acadgild@localhost ~]$ vi listfile.scala
You have new mail in /var/spool/mail/acadgild
```

OUTPUT:



```
val listScala=List("alpha","gamma","omega","zeta","beta");
println("List ' contents : "+listScala);
~
```

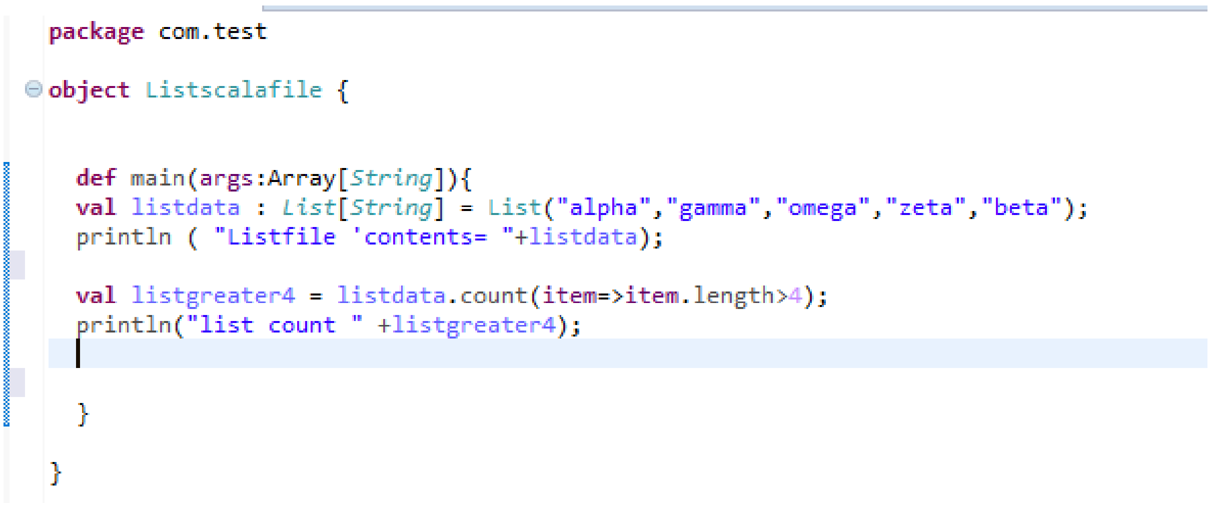


```
[acadgild@localhost ~]$ scala listfile.scala
List ' contents : List(alpha, gamma, omega, zeta, beta)
```

Big data and Hadoop – Assignment 5_Scala Basics

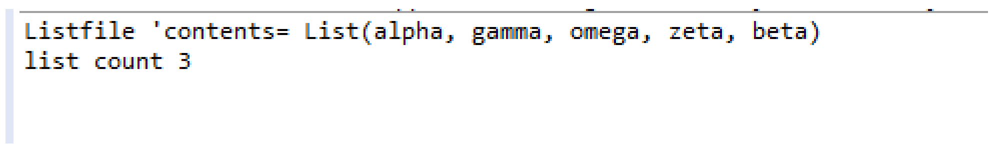
- **Can create a Scala project in IDE**

1.1 Find count of all strings with length 4.



```
package com.test

object Listscalafile {


  def main(args:Array[String]){
  val listdata : List[String] = List("alpha","gamma","omega","zeta","beta");
  println ( "Listfile 'contents= "+listdata);

  val listgreater4 = listdata.count(item=>item.length>4);
  println("list count " +listgreater4);


  }

}
```

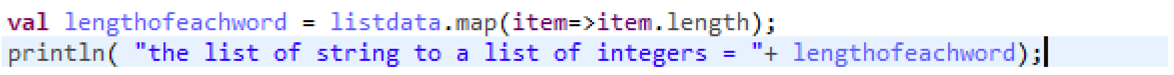
OUTPUT:



```
Listfile 'contents= List(alpha, gamma, omega, zeta, beta)
list count 3
```

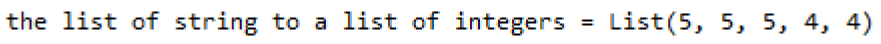
1.2 Convert the list of string to a list of integers, where each string is mapped to its corresponding length.

Map - Returns a list resulting from adding a "y" to each string element in the list



```
val lengthofeachword = listdata.map(item=>item.length);
println( "the list of string to a list of integers = "+ lengthofeachword);
```

OUTPUT:



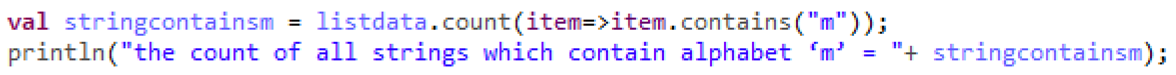
```
the list of string to a list of integers = List(5, 5, 5, 4, 4)
```

1.3 Find count of all strings which contain alphabet ‘m’.

Here to iterate through every record we will be using lambda expression

To get count we will be using count method of List

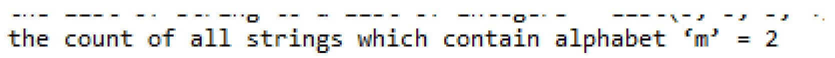
And to check if any item from List contains ‘m’, we will be using string operator ‘contains’



```
val stringcontainsm = listdata.count(item=>item.contains("m"));
println("the count of all strings which contain alphabet ‘m’ = "+ stringcontainsm);
```

Big data and Hadoop – Assignment 5_Scala Basics

OUTPUT:



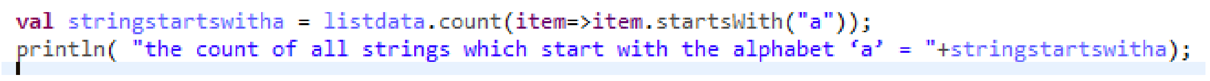
```
the count of all strings which contain alphabet 'm' = 2
```

1.4 Find the count of all strings which start with the alphabet ‘a’.

Here to iterate through every record we will be using lambda expression

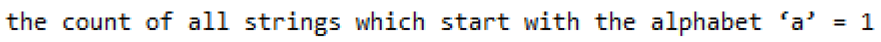
To get count we will be using count method of List

And to check if any item from List starts with ‘a’ we will be using startsWith string operator.



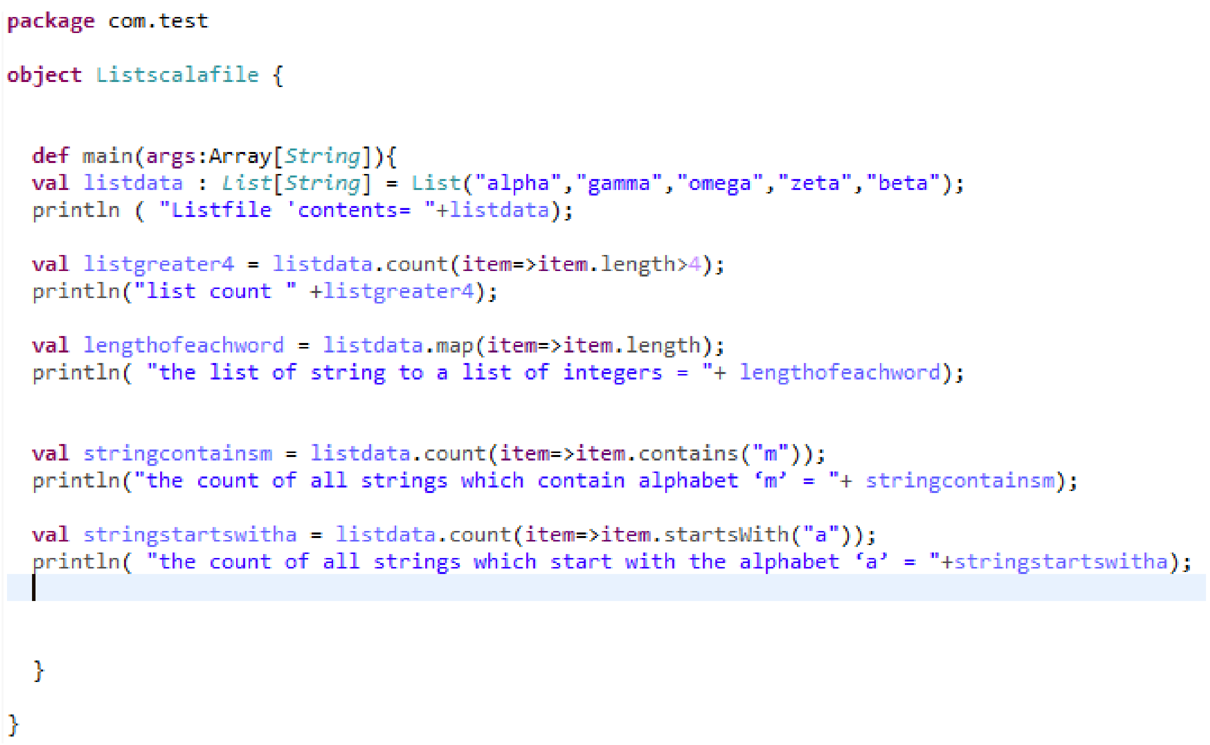
```
val stringstartswitha = listdata.count(item=>item.startsWith("a"));
println( "the count of all strings which start with the alphabet 'a' = "+stringstartswitha);
```

OUTPUT:



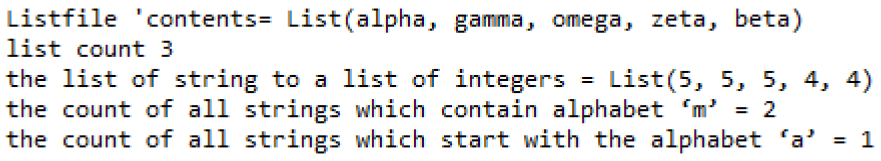
```
the count of all strings which start with the alphabet 'a' = 1
```

Whole code and output of Task 1



```
package com.test

object Listscalafile {


  def main(args:Array[String]){
  val listdata : List[String] = List("alpha","gamma","omega","zeta","beta");
  println ( "Listfile 'contents= "+listdata);

  val listgreater4 = listdata.count(item=>item.length>4);
  println("list count " +listgreater4);

  val lengthofeachword = listdata.map(item=>item.length);
  println( "the list of string to a list of integers = "+ lengthofeachword);


  val stringcontainsm = listdata.count(item=>item.contains("m"));
  println("the count of all strings which contain alphabet 'm' = "+ stringcontainsm);

  val stringstartswitha = listdata.count(item=>item.startsWith("a"));
  println( "the count of all strings which start with the alphabet 'a' = "+stringstartswitha);



  }

}
```

OUTPUT :



```
Listfile 'contents= List(alpha, gamma, omega, zeta, beta)
list count 3
the list of string to a list of integers = List(5, 5, 5, 4, 4)
the count of all strings which contain alphabet 'm' = 2
the count of all strings which start with the alphabet 'a' = 1
```

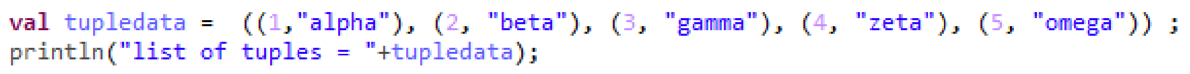
Big data and Hadoop – Assignment 5_Scala Basics

Task 2

Create a list of tuples, where the 1st element of the tuple is an int and the second element is a string. Example - ((1, 'alpha'), (2, 'beta'), (3, 'gamma'), (4, 'zeta'), (5, 'omega'))

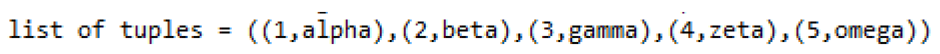
For the above list, print the numbers where the corresponding string length is 4. - find the average of all numbers, where the corresponding string contains alphabet 'm' or alphabet 'z'.

Using Scala IDE



```
val tupledata =  ((1,"alpha"), (2, "beta"), (3, "gamma"), (4, "zeta"), (5, "omega")) ;
println("list of tuples = "+tupledata);
```

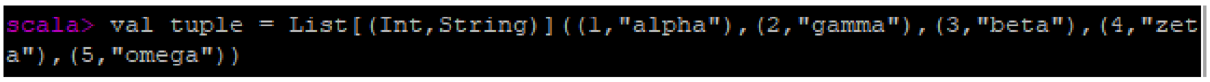
OUTPUT:



```
list of tuples = ((1,alpha),(2,beta),(3,gamma),(4,zeta),(5,omega))
```

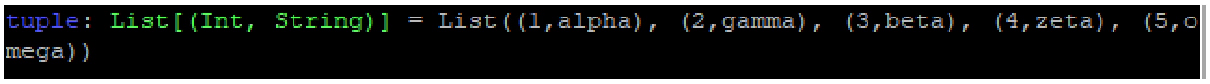
Using command prompt - scala

Step 1: By using the following command, the list of tuples is created, where the 1st element of the tuple is an int and the second element is a string.



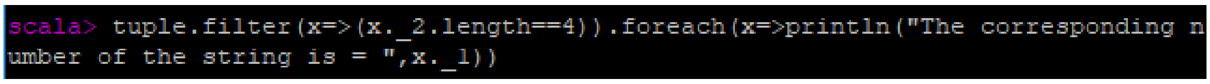
```
scala> val tuple = List[(Int,String)]((1,"alpha"),(2,"gamma"),(3,"beta"),(4,"zet
a"),(5,"omega"))
```

OUTPUT:



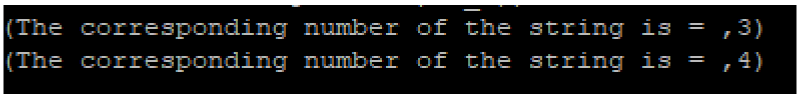
```
tuple: List[(Int, String)] = List((1,alpha), (2,gamma), (3,beta), (4,zeta), (5,o
mega))
```

Step 2: For the above list, print the numbers where the corresponding string length is 4.



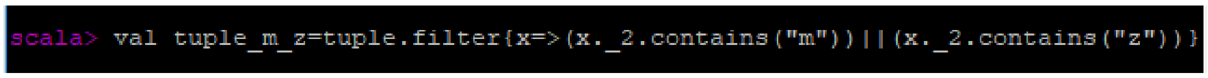
```
scala> tuple.filter(x=>(x._2.length==4)).foreach(x=>println("The corresponding n
umber of the string is = ",x._1))
```

OUTPUT:



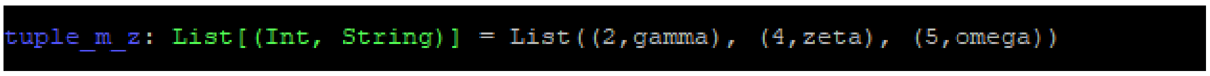
```
(The corresponding number of the string is = ,3)
(The corresponding number of the string is = ,4)
```

Step 3 : Find the average of all numbers, where the corresponding string contains alphabet 'm' or alphabet 'z'.



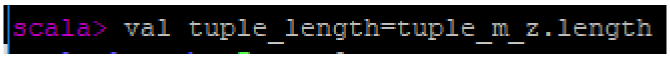
```
scala> val tuple_m_z=tuple.filter{x=>(x._2.contains("m"))||(x._2.contains("z"))}
```

OUTPUT:



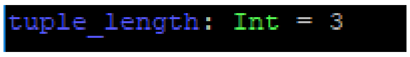
```
tuple_m_z: List[(Int, String)] = List((2,gamma), (4,zeta), (5,omega))
```

Step 4: the length of strings containing" m" or "z" alphabets are found.



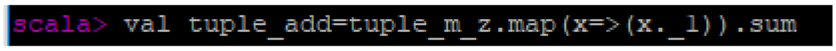
```
scala> val tuple_length=tuple_m_z.length
```

OUTPUT :



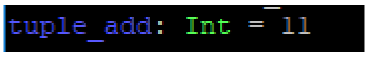
```
tuple_length: Int = 3
```

Step 5 : the integers which are mapped to strings containing" m" or "z" alphabets are summed up together.



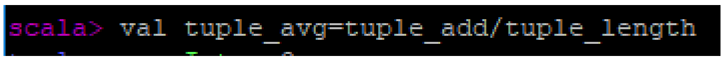
```
scala> val tuple_add=tuple_m_z.map(x=>(x._1)).sum
```

OUTPUT:



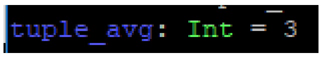
```
tuple_add: Int = 11
```

Step 6: the average of all numbers, where the corresponding string contains alphabet 'm' or alphabet 'z' are determined.



```
scala> val tuple_avg=tuple_add/tuple_length
```

OUTPUT :

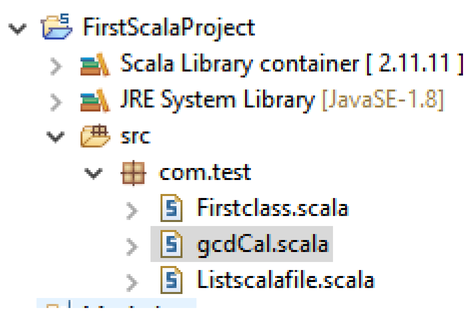


```
tuple_avg: Int = 3
```

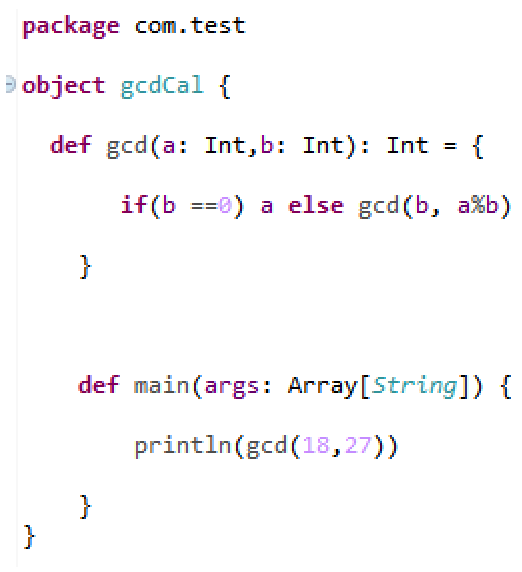
Big data and Hadoop – Assignment 5_Scala Basics

Task 3

Create a Scala application to find the GCD of two numbers

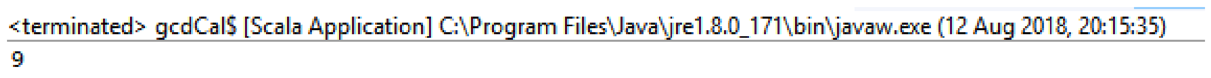


```
FirstScalaProject
  Scala Library container [ 2.11.11 ]
  JRE System Library [JavaSE-1.8]
  src
    com.test
      Firstclass.scala
      gcdCal.scala
      Listscalafile.scala
```



```scala
package com.test

object gcdCal {

  def gcd(a: Int,b: Int): Int = {

      if(b ==0) a else gcd(b, a%b)

    }


    def main(args: Array[String]) {

      println(gcd(18,27))

    }
}
```

OUTPUT:



```
<terminated> gcdCal$ [Scala Application] C:\Program Files\Java\jre1.8.0_171\bin\javaw.exe (12 Aug 2018, 20:15:35)
9
```

Steps of output:

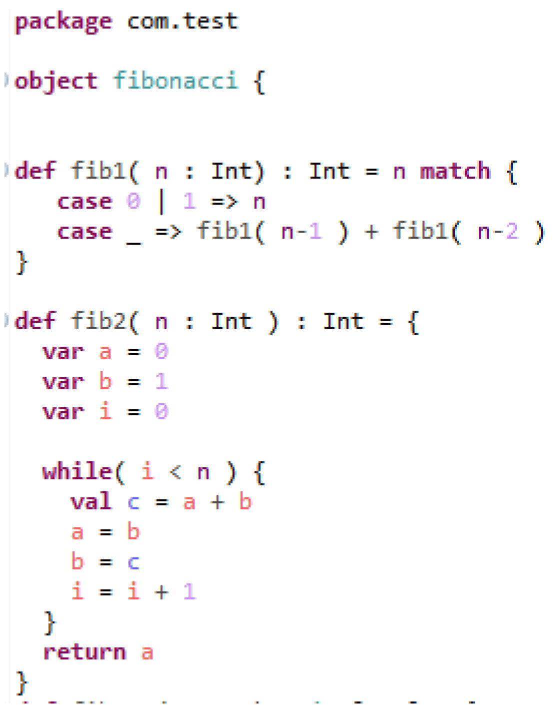
1. (18%27) is 18 which is !=0
2. (27%18) is 9 which is !=0
3. (18%9) is 0 which ==0
4. So output is "9"

Big data and Hadoop – Assignment 5_Scala Basics

Task 4:

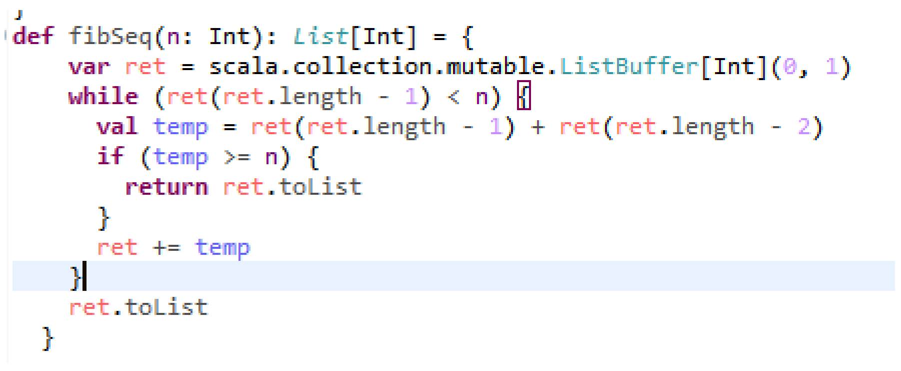
Fibonacci series (starting from 1) written in order without any spaces in between, thus producing a sequence of digits. Write a Scala application to find the Nth digit in the sequence. ➢ Write the function using standard for loop ➢ Write the function using recursion

1. **to find the Nth digit in the sequence**

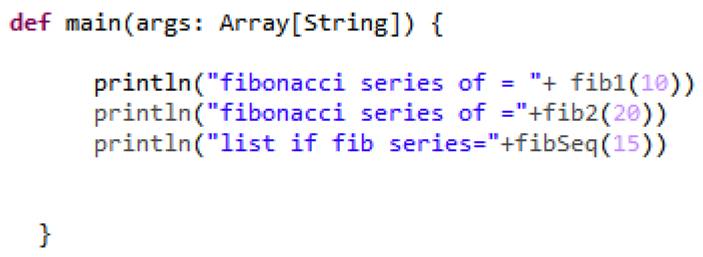


```
package com.test

object fibonacci {


def fib1( n : Int) : Int = n match {
   case 0 | 1 => n
   case _ => fib1( n-1 ) + fib1( n-2 )
}

def fib2( n : Int ) : Int = {
  var a = 0
  var b = 1
  var i = 0

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

sequence of digits



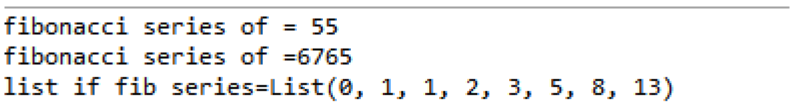
```
def fibSeq(n: Int): List[Int] = {
    var ret = scala.collection.mutable.ListBuffer[Int](0, 1)
    while (ret(ret.length - 1) < n) {
      val temp = ret(ret.length - 1) + ret(ret.length - 2)
      if (temp >= n) {
        return ret.toList
      }
      ret += temp
    }
    ret.toList
  }
```



```
def main(args: Array[String]) {

      println("fibonacci series of = "+ fib1(10))
      println("fibonacci series of ="+fib2(20))
      println("list if fib series="+fibSeq(15))


  }
```

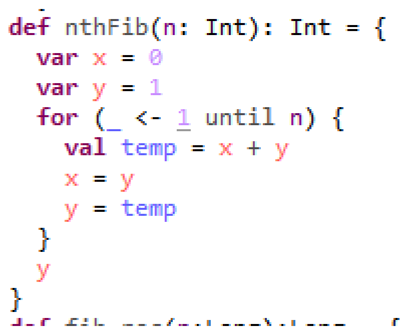
Big data and Hadoop – Assignment 5_Scala Basics

OUTPUT:

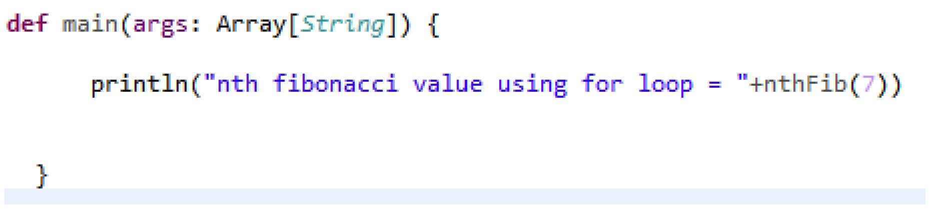


```
fibonacci series of = 55
fibonacci series of =6765
list if fib series=List(0, 1, 1, 2, 3, 5, 8, 13)
```

- **Write the function using standard for loop**



```
def nthFib(n: Int): Int = {
  var x = 0
  var y = 1
  for (_ <- 1 until n) {
    val temp = x + y
    x = y
    y = temp
  }
  y
}
```

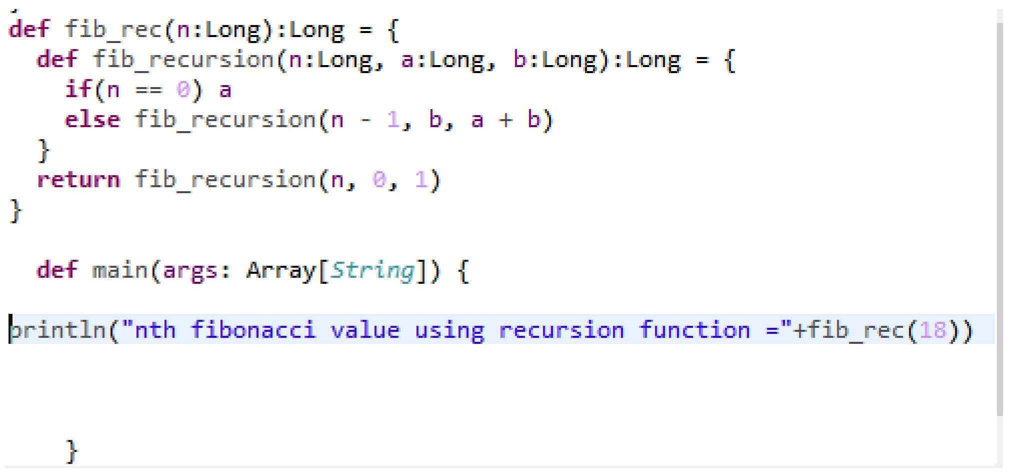


```
def main(args: Array[String]) {

      println("nth fibonacci value using for loop = "+nthFib(7))


  }
```

OUTPUT:

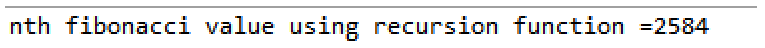
```
nth fibonacci value using for loop = 13
```

- **Write the function using recursion**



```
def fib_rec(n:Long):Long = {
  def fib_recursion(n:Long, a:Long, b:Long):Long = {
    if(n == 0) a
    else fib_recursion(n - 1, b, a + b)
  }
  return fib_recursion(n, 0, 1)
}

  def main(args: Array[String]) {

println("nth fibonacci value using recursion function ="+fib_rec(18))



    }
```

OUTPUT :



```
nth fibonacci value using recursion function =2584
```

Big data and Hadoop – Assignment 5_Scala Basics

Task 5

Find square root of number using Babylonian method.

1. Start with an arbitrary positive start value x (the closer to the root, the better).

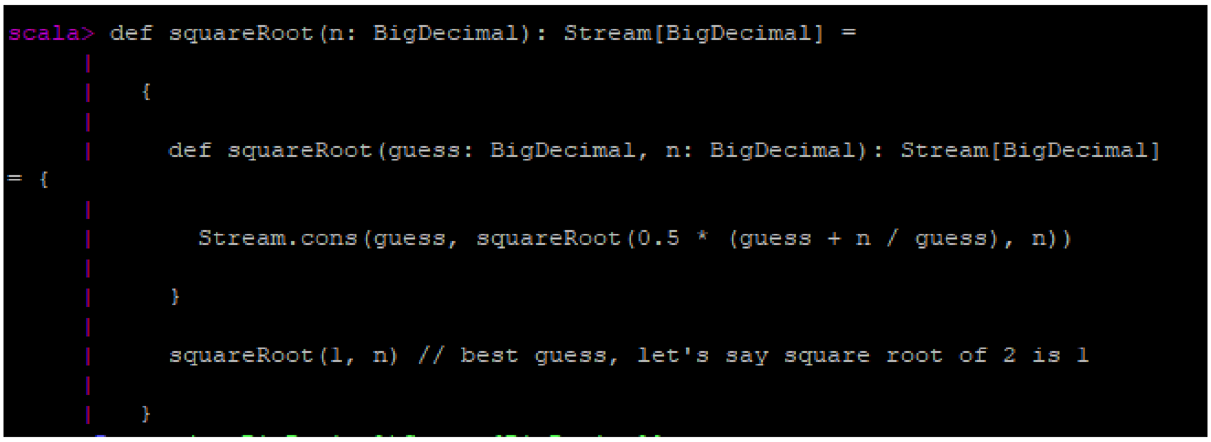
2.Initialize y = 1.

3. Do following until desired approximation is achieved.

a) Get the next approximation for root using average of x and y

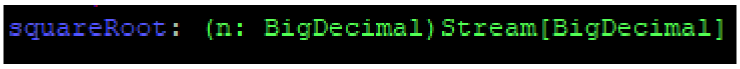
b) Set y = n/x

Step 1:



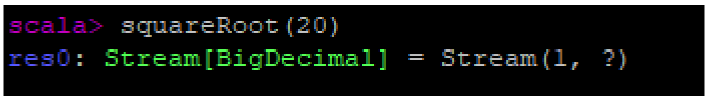
```
scala> def squareRoot(n: BigDecimal): Stream[BigDecimal] =
     |
     |    {
     |
     |      def squareRoot(guess: BigDecimal, n: BigDecimal): Stream[BigDecimal]
= {
     |
     |        Stream.cons(guess, squareRoot(0.5 * (guess + n / guess), n))
     |
     |      }
     |
     |      squareRoot(1, n) // best guess, let's say square root of 2 is 1
     |
     |    }
```

OUTPUT:



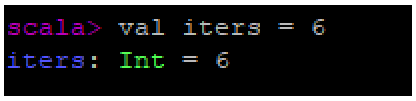
```
squareRoot: (n: BigDecimal)Stream[BigDecimal]
```

Step 2: Iterations need to be mentioned



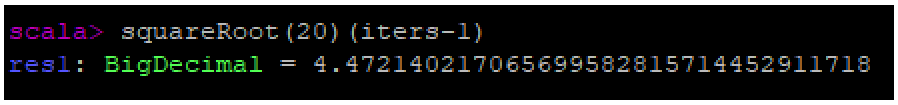
```
scala> squareRoot(20)
res0: Stream[BigDecimal] = Stream(1, ?)
```

Step 3: Number of iterations for calculating square root of a number



```
scala> val iters = 6
iters: Int = 6
```

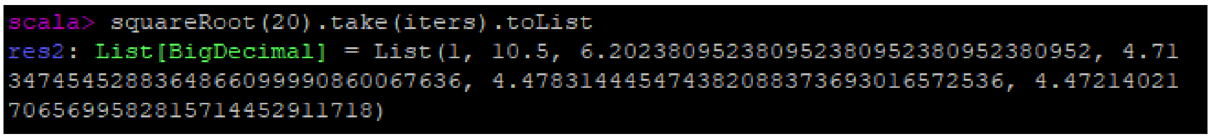
Step 4: Square root of "20" with result of 5th iteration



```
scala> squareRoot(20)(iters-1)
res1: BigDecimal = 4.472140217065699582815714452911718
```

Big data and Hadoop – Assignment 5_Scala Basics

Step 5: Presenting the result in the form of list with all 6 iterations of square root of “20”



```
scala> squareRoot(20).take(iters).toList
res2: List[BigDecimal] = List(1, 10.5, 6.202380952380952380952380952380952, 4.71
3474545288364866099990860067636, 4.478314445474382088373693016572536, 4.47214021
7065699582815714452911718)
```