ASSIGNMENT 44.1 – SCALA 2

Mar 2018 batch - Student: K. Anandaranga

---

1. Task 1

A 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

Solution:

First, using a while loop:

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help

scala> 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
     | }
fib: (n: Int)Int

scala> fib(10)
1
2
3
5
8
13
21
34
55
89
res22: Int = 89

scala>
```

Second, using recursion

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help

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

scala> fib_r(10)
res23: Int = 89
```

Task 2

Create a calculator to work with rational numbers.

Requirements:
○ It should provide capability to add, subtract, divide and multiply rational numbers
○ Create a method to compute GCD (this will come in handy during operations on rational)
Add option to work with whole numbers which are also rational numbers i.e. (n/1)
- achieve the above using auxiliary constructors
- enable method overloading to enable each function to work with numbers and rational.

```
scala> class Rational (n:Int, d:Int = 1) {
     | private def gcd(x: Int, y:Int): Int = {
     | if (x ==0) y
     | else if (x <0) gcd(-x,y)
     | else if (y <0) -gcd(x,-y)
     | else gcd(y%x, x)
     | }
     | private val g = gcd(n,d)
     |
     | val numer: Int = n/g
     | val denom: Int = d/g
     | def add_r (that:Rational) = new Rational (numer* that.denom + that.numer*denom, denom*that.denom)
     | def sub_r (that:Rational) = new Rational (numer*that.denom - that.numer*denom, denom*that.denom)
     | def mult_r(that:Rational) = new Rational (numer *that.numer, denom*that.denom)
     | def div_r(that:Rational) = new Rational (numer*that.denom, denom*that.numer)
     | override def toString = numer + "/" + denom
     | }
defined class Rational

scala> var x = new Rational (3,4)
x: Rational = 3/4

scala> var y = new Rational (1)
y: Rational = 1/1

scala> x add_r y
res44: Rational = 7/4

scala> x sub_r y
res45: Rational = -1/4

scala> x mult_r y
res46: Rational = 3/4

scala> x div_r y
res47: Rational = 3/4

scala>
```

2. Task 3

1.Write a simple program to show inheritance in scala.

- Inheritance is an object-oriented concept which is used to reusability of code.
- One can achieve inheritance by using extends keyword
- To achieve inheritance a class must extend to other class.
    - A class which is extended called super or parent class
    - A class which extends is called derived or base class

acadgild@localhost:~

File Edit View Search Terminal Help

```
scala> class Employee {
     | var salary: Float = 100000
     | }
defined class Employee

scala>

scala> class DataScientist extends Employee {
     | var bonus: Float = 25000
     | println ("Salary = " + salary)
     | println ("Bonus = " + bonus)
     | }
defined class DataScientist

scala> new DataScientist()
Salary = 100000.0
Bonus = 25000.0
res51: DataScientist = DataScientist@6d95b51d

scala>
```

2.Write a simple program to show multiple inheritance in scala.

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help

scala> class BaseSalary {
     | var base_salary: Float = 100000
     | }
defined class BaseSalary

scala>

scala> class Perks extends BaseSalary {
     | var perks = 20000
     | }
defined class Perks

scala>

scala> class CTC extends Perks {
     | def show(){
     | println("BaseSalary = " + base_salary)
     | println ("Perks     = " + perks)
     | }
     | }
defined class CTC

scala>

scala> new CTC().show
BaseSalary = 100000.0
Perks     = 20000

scala>
```

3.Write a partial function to add three numbers in which one number is constant and two numbers can be passed as inputs and define another method which can take the partial function as input and squares the result.

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help

scala> val add_p: PartialFunction[(Double, Double), Double] = {
     | case (x, y) => x + y + 100 // assume that constant = 100
     | }
add_p: PartialFunction[(Double, Double),Double] = <function1>

scala>

scala> def sq_p(a:Double, b: Double):Double = {
     | var s = math.pow(add_p(a,b),2)  // using the 2nd power for squaring
     | return s
     | }
sq_p: (a: Double, b: Double)Double

scala> sq_p(1,2)
res75: Double = 10609.0

scala> sq_p(1.1, 2.2)
res76: Double = 10670.89

scala> sq_p(-1,-2)
res77: Double = 9409.0

scala> sq_p(0,0)
res78: Double = 10000.0

scala>
```

4.Write a program to print the prices of 4 courses of Acadgild: Android-12999,Big Data Development-17999,Big Data Development-17999,Spark-19999 using match and add a default condition if the user enters any other course

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help

scala> def acdg = {
     | println("Welcome! Acadgild offers 4 courses - Android, BigData, DataScience, Spark")
     | val courseName = scala.io.StdIn.readLine("Which course? ")
     | val courseFee = courseName match {
     | case "Android" => "12999"
     | case "BigData" => "15999"
     | case "DataScience" => "17999"
     | case "Spark" => "19999"
     | case _ => "Please type a valid course"
     | }
     | println (s"Course fee for $courseName = $courseFee")
     | }
acdg: Unit

scala> acdg
Welcome! Acadgild offers 4 courses - Android, BigData, DataScience, Spark
Which course? Course fee for Android = 12999

scala> acdg
Welcome! Acadgild offers 4 courses - Android, BigData, DataScience, Spark
Which course? Course fee for BigDAta = Please type a valid course

scala> acdg
Welcome! Acadgild offers 4 courses - Android, BigData, DataScience, Spark
Which course? Course fee for BigData = 15999

scala> acdg
Welcome! Acadgild offers 4 courses - Android, BigData, DataScience, Spark
Which course? Course fee for  = Please type a valid course

scala>
```