University of Colombo School of Computing

SCS 2204
Functional Programming

Tutorial 4

Dinuka Amarasinghe
20000103

Question 01

```
def interest(x:Double):Double = x match{
     case x if x<= 20000 => x*0.02
     case x if x<= 200000 => x*0.04
     case x if x<= 2000000 => x*0.035
     case x if x> 2000000 => x*0.065
}
print("Enter the number of amount : ")
var amount = scala.io.StdIn.readDouble()
print("Interest : "+interest(amount))
```

Question 02

```
case class cartitem(name:String, price:Float, quantity:Int)
var cart = new Array[cartitem](3)
cart(0) = cartitem("Vanilla ice cream",3.99,13);
cart(1) = cartitem("Chocolate Biscuits",4,6);
cart(2) = cartitem("Cupcakes",7.77,7);
def printcart(x:Array[cartitem]):Any = {
     if(!x.isEmpty){
         val quantity = x.head.quantity;
         val name = x.head.name;
         val price = x.head.price;
         val str = s"$quantity $name at Rs $price each"
         println(str)
         printcart(x.tail)
     }
}

println("Printing the cart items : ")
printcart(cart)
println("")
def printonly(x:Array[cartitem]):Any = {
     if(!x.isEmpty){
         val name = x.head.name;
         if(name contains "Vanilla ice cream"){
             val quantity = x.head.quantity;
             val price = x.head.price;
             val str = s"$quantity $name at Rs $price each"
             println(str)
         }else{
```

```
            println("Found another item")
        }
        printonly(x.tail)
    }
}

println("Printing only vanilla ice cream items: ")
printonly(cart)
println("")
```

Question 03

```
case class vehicle(name:String, price:Double)
object Carxt{
   implicit class CarUUID(x: vehicle){
         def uuid: String = s"car uuid = ${x.name} - ${x.name.hashCode}"
   }
}
import Carxt._
val vehicles = List(vehicle("bmw 3 series",20000),
                    vehicle("bmw 5 series",50000),
                    vehicle("vw passat",10000),
                    vehicle("vw golf",12000),
                    vehicle("mazda 3",15000))
vehicles.foreach(v => println(v.uuid))
```

Question 04

```
print("Enter a number : ")
var input = scala.io.StdIn.readInt()
def PatternMatching(x:Int):Any = x match{
     case x if x<=0 => println("The number is Negative/Zero")
     case x if x%2==0 => println("The number is even")
     case x => println("The number is odd")
}
PatternMatching(input)
```

Question 05

```
def toUpper(x: String): String = {
   x.toUpperCase()
}
def toLower(x: String): String = {
   x.toLowerCase()
}

def formatNames(name: String, func: (String) => String): String = {
   func(name)
}

println(formatNames("Benny",toUpper(_)))
print(formatNames("N",toUpper(_))+formatNames("iroshan",toLower(_)))
println("")
println(formatNames("Saman",toLower(_)))
println(formatNames("K",toUpper(_))+formatNames("umar",toLower(_))+formatNames("a
",toUpper(_)))
```