Übung 5 – Aufgabe

Aufgabe 1: Implementieren Sie die folgenden Aufgabenstellungen:

a) Gegeben sei eine Liste von beliebigen Zahlen. Schreiben Sie eine Funktion, die mittels eines Aggregationsoperators den Durchschnitt aller geraden Zahlen und den Durchschnitt aller ungeraden Zahlen. Dabei soll nur einmal durch die Liste gegangen werden.

b) Schreiben Sie eine Funktion, die in einer Liste von Zahlen alle Werte dupliziert (nicht verdoppelt). Verwenden Sie dafür nur Higher Order Functions.

c) Gegeben seien die beiden folgenden Listen:
val l1=List(1,2,3,4)
val l2=List("a","b","c")
Schreiben Sie eine Funktion, die aus den beiden Listen ein kartesisches Produkt bildet. Ergebnis soll eine Liste von Tupeln sein, deren erstes Element aus l1 kommt und deren zweites aus l2. Verwenden Sie dafür nur Higher Order Functions.

Aufgabe 2: Implementieren Sie die folgenden Aufgabenstellungen:

a) Schreiben Sie eine Funktion moduloMap(l:List[Int], mod_value:Int):Map[Int,List[Int]], die aus einer Liste von Zahlen, eine Map erzeugt, deren Schlüssel ein Int-Wert ist, der sich aus der Modulo-Rechnung des Listenwertes mit mod_value ergibt. Zu den Schlüsselwerten werden dann alle Ints der Ausgangsliste innerhalb einer Liste gespeichert: z.B.:

val l= List(1,4,5,7,8,9)
moduloMap(l,3) ergibt dann:

Map(1 -> List(7, 4, 1), 2 -> List(8, 5), 0 -> List(9))

Benutzen Sie dafür nur einen Aggregationsoperator!

b) Gegeben sei eine Liste von Wörtern. Schreiben Sie eine Funktion countLetters(l:List[String]):Map[Int,Int], die aus der Liste von Wörtern eine Map generiert, in der gespeichert wird, wie viele Wörter es mit einer entsprechenden Buchstabenzahl (Schlüssel) gibt: z.B.:

val w=List("Hallo","das","sind","ein","paar", "Wörter")
countLetters(w)

ergibt: Map(5 -> 1, 3 -> 2, 4 -> 2, 6 -> 1)

Benutzen Sie dafür nur eine Aggregationsfunktion.

c) Wandeln Sie die Funktion so um, dass nicht die Anzahl der Wörter gespeichert wird, sondern die Wörter selbst. Benutzen Sie nur eine Aggregationsfunktion.

d) Schreiben Sie eine Funktion avgNumbers(l:List[Int]):Map[Boolean, Double]. Die Funktion soll aus der Liste die Durchschnittswerte der geraden und der ungeraden Zahlen bilden. Der Schlüsselwert soll dabei ein Boolean sein, der bei true alle geraden Werte zusammenfasst und false bei allen ungeraden: z.B.:

val l2= List(1,4,5,7,8,9)

avgNumbers(l) ergibt:

Map(false -> 5.5, true -> 6.0)