Prof. Martin Hofmann, PhD
Dr. Steffen Jost

Ludwig-Maximilians-Universität München
Institut für Informatik
26. April 2013

1. Übung zur Vorlesung

Einführung in die Funktionale Programmierung

Hinweise: Die Aufgaben dieses Blattes sind dazu gedacht, "zu Fuß" gelöst zu werden, d.h. ohne die Verwendung von Funktion der Standardbibliothek. Die Verwendung sehr grundlegender Funktionen oder Standard-Operatoren für die Typen `Bool` und `Int`, wie z.B. Gleichheit, Größenvergleich, Addition und logische Verknüpfungen sind gestattet, nicht jedoch Funktionen auf Listen oder Funktionen höherer Ordnung.

Falls Sie in Aufgaben verwendete Funktion aus der Standardbibliothek nicht kennen, dürfen Sie GHCI nach deren Typ befragen oder in der Dokumentation nachschlagen!

A1-1 ***Erste Schritte*** Berechnen Sie die die 27. Fibonacci-Zahl. Kopieren Sie dazu den Code aus der letzten Vorlesung in eine Datei, laden Sie diese Datei in GHCI und führen Sie die Berechnung durch!

A1-2 ***Kartesisches Produkt***

a) Was sind die Elemente des Produkt $\{1,2\} \times \{3,4\}$?

b) Sei A eine Menge mit n Elementen. Wie viele Elemente gibt es in $A \times \{27,69\}$?

c) Was sind die Elemente des Produkt $(\{1,2\} \times \{3,4\}) \times \{5,6\}$?

d) Was sind die Elemente des Produkt $\{1,2\} \times (\{3,4\} \times \{5,6\})$?

A1-3 ***Auswertung*** Berechnen Sie möglichst mit Papier und Bleistift den Wert, zu dem der gegebene Ausdruck jeweils auswertet. Überprüfen Sie Ihr Ergebnis mit GHCI.

a) `2 + 3 * 5== 34 ‘div‘ 2`

b) `(4 == 5.0):[]`

c) `’c’:’o’:’o’:’l’:"!"`

d) `let mond="käse" in if mond=="käse" && 1==2 then False else 1+1==2`

e) `[c | c <- "Schön mies dat!", c/=’i’, c/=’e’, c/=’s’, c/=’m’]`

f) `[(a,b) | a<-[1..10], b<-[10..1], a/=b]`

g) `[(y,z) | x<-[’m’..’n’], y<-[1,200..500], let z = x:""]`

h) `(\x->"nope!") [c | c <- "yes!"]`

A1-4 ***Typen*** Welchen Typ haben folgenden Ausdrücke?

Hinweis: Auch wenn wir Typinferenz in der Vorlesung noch nicht behandelt haben, sollten Sie in der Lage sein, die meisten dieser einfachen Aufgaben bereits alleine mit Papier und Bleistift lösen zu können. Dies Aufgabe ist auch eigentlich Einfacher als Aufgabe A1-3.

a) `['a','b','c']`

b) `('a','b','c')`

c) `[(False,[1]),(True,[(2.0)])]`

d) `([True,True],('z','o','o'))`

e) `(\x -> ('a':x,False,([x])))`

f) `[z == 2.0 | x <- [1..10], let y = (2*x,x+1), let (z,_) = y]`

g) `[(\x y-> (x*2,y-1)) m n| m<-[1..5], even m, n<-[6..10]]`

h) `(\x -> (x,x))`

i) `(\(x,y) -> ([y],[],x:x:x:[]))`

j) `(\a b c -> a:b:c:[])`

k) `[tail,init,reverse]`

A1-5 ***Listenbearbeitung***

a) Schreiben Sie eine Funktion mit dem Namen `oddCollatzStep` und mit dem Typ `[Int] -> [Int]`, welche für jede Eingabeliste eine Ausgabeliste produziert, in dem man jedes Element der Eingabeliste mit 3 multipliziert und danach um eins erhöht.

b) War Ihre Lösung zu a) rekursiv?

   Falls ja, schreiben Sie noch einmal eine nicht-rekursive Funktion, welche das gleiche berechnet.

   Falls nein, dann schreiben Sie nun eine rekursive Funktion, welche das gleiche berechnet.

c) Schreiben Sie eine Funktion `seekPosMaxMin` mit dem Typ `[Double] -> (Double,Double)`, welche den kleinsten und größten nicht-negativen Wert einer Liste von Fließkommazahlen berechnet.

d) Schreiben Sie eine Funktion `umkehrer` mit dem Typ `String -> String`, welche einen die Reihenfolge der einzelnen Zeichen eines Strings genau umkehrt.

   Ist Ihre Lösung endrekursiv?

Abgabe: Lösungen zu den Hausaufgaben können bis Freitag, den 10.05.2013, 08:00 Uhr früh mit UniworX abgegeben werden.