

Funktionale Programmierung

2. Übungsblatt (für das Tutorium)

Prof. Dr. Margarita Esponda

Ziel: Auseinandersetzung mit Tuples, Listen und erste rekursive Funktionen.

1. Aufgabe (Ziel: Variablenbindung bzw. Gültigkeitsbereich oder Sichtbarkeit von Variablen)

Betrachten Sie folgende haskell-Funktionsdefinitionen:

```
u = let a = 0
      b = a + ( let c = 3 in c + 2*a )
      in b * b

v n = let x = n
      y = x + ( let x = 3
                  y = z
                  in x+y )
      z = x^2
      in y*y
```

Ohne die Funktionen in Haskell einzugeben, berechnen Sie den Wert folgender Ausdrücke.

u v 2

2. Aufgabe (Ziel: Einfache rekursive Funktionen mit Listen zu definieren)

Schreiben Sie eine Haskell-Funktion, die alle Leerzeichen aus einem beliebigen Text entfernt.

- Verwenden Sie in Ihrer Funktionsdefinition explizite Rekursion.
- Definieren Sie die gleiche Funktion mit Hilfe von Listengeneratoren.

Anwendungsbeispiel:

skipSpaces "abe cd a b h" => "abcdabh"

3. Aufgabe (Ziel: weiter mit rekursive Funktionen mit Listen)

Definieren Sie eine Haskell-Funktion, die aus einer beliebigen Binärzahl **n** (Zweierkomplement-Darstellung) die entsprechende negative Zahl (**-n**) berechnet. Es wird selbstverständlich angenommen, dass die Eingabe- und Ergebniszahl der Funktion immer die gleiche Bitlänge haben.

Anwendungsbeispiel:

twoComplement [0,0,0,1,1,0,1,0] => [1,1,1,0,0,1,1,0]

Wichtige Hinweise:

- 1) Verwenden Sie geeignete Namen für Ihre Variablen und Funktionsnamen, die den semantischen Inhalt der Variablen oder die Semantik der Funktionen wiedergeben.
- 2) Verwenden Sie vorgegebene Funktionsnamen, falls diese angegeben werden.
- 3) Kommentieren Sie Ihre Programme.
- 4) Verwenden Sie geeignete lokale Funktionen und Hilfsfunktionen in Ihren Funktionsdefinitionen.
- 5) Schreiben Sie in alle Funktionen die entsprechende Signatur.