SoSe 2014

Prof. Dr. Margarita Esponda

# ProInformatik II: Funktionale Programmierung

## **2. Übungsblatt** (3. Tag)

**1. Aufgabe**

a) Schreiben Sie eine rekursive Funktion, die alle geraden Zahlen zwischen 1 und n aufsummiert.

b) Gibt es neben der rekursiven Berechnung noch weitere Berechnungsmöglichkeiten? Wenn ja, programmieren Sie auch diese Funktion.

**2. Aufgabe**

Definieren Sie eine rekursive Funktion **arrow**, die bei Eingabe zweier natürliche Zahlen k und n, die Zahl **k** **n**-mal potenziert. D.h. Die Anzahl der **k**-Elemente in dem Potenz-Turm ist gleich **n**.

Vergessen sie dabei nicht, dass die Potenz-Funktion rechtsassoziativ ist.

$k \uparrow n = k^{k^{\cdot^{\cdot^{k}}}}$ (der Turm auf der rechten Seite der Gleichung hat **n** mal das Argument **k)**

**3. Aufgabe**

Schreiben Sie eine Funktion, die bei Eingabe einer positiven Zahl die Einsen der Binärstellung der Zahl addiert (Quersumme der Binärdarstellung der Zahl).

**4. Aufgabe**

Definieren Sie eine Haskell-Funktion, die bei Eingabe einer Zahl in Oktal-Darstellung die Hexadezimal-Darstellung der Zahl berechnet.

Die Zahl soll als Zeichenkette eingegeben werden.

Anwendungsbeispiel: **okt2hex** "0770" => "1F8"

**5. Aufgabe**

Definieren Sie eine Funktion **sumDigits**, die die Quersumme einer Zahl so lange berechnet, bis das Ergebnis nur aus einer Ziffer besteht (Zahl zwischen 0-9)

Anwendungsbeispiel: **sumDigits** 452317 => 4

**6. Aufgabe**

Programmieren Sie eine Funktion, die die Elemente aus zwei Listen multipliziert. Verwenden Sie die **error**-Funktion für den Fall, dass die Listen nicht gleich lang sind.

Anwendungsbeispiel:

**multLists** [2, 4, 0, 1] [0, 1, 0, 2] => [2, 4, 0, 2]

**7. Aufgabe**

Verändern Sie die balance-Funktion aus den Vorlesungsfolien, sodass in einer beliebigen Zeichenkette kontrolliert wird, ob die Klammersetzung korrekt ist.

Anwendungsbeispiele: balanced "(a+b)*[x-y]/{(x+1)*5}" => True

balanced "(a+b)*x-y)/{(x+1)*5}" => False

**8. Aufgabe**

Definieren Sie eine Funktion **flatten**, die aus einer Liste von Listen eine einfache Liste macht

Beispiel:

flatten [[8,2],[3],[],[4,5,0,1]] = [8,2,3,4,5,0,1]

**9. Aufgabe**

Definieren Sie eine Funktion **makeSet** , die wiederholte Elemente aus einer Liste entfernt, so dass am Ende jedes unterschiedliche Element nur einmal vorkommt.

Beispiel:

makeSet [1,2,1,2,2,1,3] = [1,2,3]
makeSet ['h','e','l','l','o'] = ['h','e','l','o']

**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.