Albert-Ludwigs-Universität Freiburg
Informatik 1
Wintersemester 2011/12

Prof. Dr. Wolfram Burgard
Dipl. Ing. Jörg Röwekämper
Dipl. Inf. Markus Kuderer

Übungsblatt 12

Abgabe bis Donnerstag, 02.02.12, 12:00 Uhr

Aufgabe 12.1

Implementieren Sie folgende Haskell Funktionen:

- `factorial :: Int->Int`
  ist für positive Integer definiert und liefert die Fakultät ($n!$) der Eingabe zurück.
- `f :: Int->Int->Int`
  soll folgende Funktion berechnen.

$$f(x,y) = \begin{cases} 1 & \text{falls } y = 0 \\ f\left(f\left(x, y/2\right), 2\right) & \text{falls } y > 2 \text{ und } y \bmod 2 = 0 \\ x \cdot f(x, y-1) & \text{sonst} \end{cases}$$

Aufgabe 12.2

Implementieren Sie folgende Haskell Funktionen:

- `factorialList` berechnet $n!$ für alle Elemente einer Liste:

```
factorialList [1,3,4]
[1,6,24]
```

- `sortInsert` fügt ein Element an der richtigen Stelle in eine bereits sortierte Liste ein.

```
sortInsert 3 [1,2,4,5]
[1,2,3,4,5]
```

- `removeElementAt` entfernt das Element an der angegebenen Position aus der Liste.

```
removeElementAt 0 [5,2,17,21,13]
[2,17,21,13]
```

- `findElement` liefert die Position eines Elements in der Liste zurück. Wenn das Element nicht in der Liste vorhanden ist, wird -1 zurückgegeben.

```
findElement [5,2,17,21,13] 4
-1
findElement [5,2,17,21,13] 17
2
```

Aufgabe 12.3

Betrachten Sie das folgende Haskell-Programm `f`, das für $\mathbb{N}_0$ definiert ist ($\mathbb{N}_0$ sind die natürliche Zahlen inklusive 0). Was berechnet dieses Programm?

```
f 0 = g [0]
f 1 = g [1]
f x = g [y | y <- [2..x-1], mod x y == 0]

g [] = True
g (x:xs) = False
```

Aufgabe 12.4

Betrachten Sie die folgenden Funktionen und berechnen Sie den Wert des Ausdrucks `take 1 (filter p [2..])`. Geben Sie alle Reduktionsschritte an.

```
take 0 xs     = []
take _ []     = []
take n (x:xs) = x:take (n-1) xs

filter p []     = []
filter p (x:xs) = if (p x) then (x:filter p xs)
                  else (filter p xs)

p x = if mod x 2 == 0 then True
      else False
```