WiSe 2015/2016

Funktionale Programmierung

6. Übungsblatt (Abgabe: Mi., den 02. Dez. um 10:10 Uhr)

Prof. Dr. Margarita Esponda

Ziel: weitere Auseinandersetzung mit der Komplexität-Analyse von Funktionen, Funktionen höherer Ordnung und erster Einstieg in algebraische Datentypen.

1. Aufgabe (3 Punkte)

Definieren Sie eine Haskell-Funktion **abflachen** :: [[a]] -> [a], welche eine Liste von Listen zu einer Liste kombiniert.

Anwendungsbeispiel:

abflachen ["abc", "bcd", "abc"] => "abcbcdabc"

a) Schreiben Sie zuerst Ihre Definition mit Hilfe der **foldr**-Funktion.

b) Schreiben Sie eine zweite Definition mit der **foldl**-Funktion.

c) Was sind die Vorteile und/oder Nachteile beider Definitionen aus a) und b)?

2. Aufgabe (3 Punkte)

Definieren Sie unter sinnvoller Verwendung der **foldl** Funktion die Funktion **toDecFrom**, die eine Zahl als Liste der einzelnen Ziffern in Basis **b (**mit **1<b<10**) bekommt und die Dezimaldarstellung der Zahl berechnet.

Anwendungsbeispiel: **toDecFrom** [1,3,2] 4 => 30

3. Aufgabe (2 Punkte)

Definieren Sie einen algebraischen Datentyp **Color** indem Sie Farben in verschiedenen Formaten darstellen, und definieren Sie damit die **rgb2cmyk** Funktion aus dem 1. Übungsblatt.

4. Aufgabe (7 Punkte)

Nehmen wir an, wir müssen eine Reihe Berechnungen realisieren mit Zahlen, die folgende Form haben:

$a + b\sqrt{2}$, wobei ***a*** und ***b*** ganze Zahlen sind.

Würden wir zuerst die Wurzeln ausrechnen und dann die Summen machen, hätten wir Rundungsfehler, die im Laufe der Berechnungen größer werden können. Die Rundungsfehler können minimiert werden, wenn wir zuerst nur die ganzzahligen Operationen realisieren und das Ausrechnen der Wurzeln ans Ende verschieben. Beispiel:

$$\left(3 + 2\sqrt{2}\right) \cdot \left(2 + \sqrt{2}\right) = 10 + 7\sqrt{2}$$

a) Definieren Sie einen algebraischen Datentyp **RootNum**, der unsere Zahlen mit Hilfe der Koeffizienten **a** und **b** darstellt, und definieren Sie die Additions-, Subtraktions- und Multiplikationsoperation für diesen Datentyp.

b) Zum Testen definieren Sie eine Funktion **getValue,** die eine **RootNum** Variable in einer Gleitkommazahl ausrechnet.

5. Aufgabe (9 Punkte)

Ein Element einer Liste von ***n*** Objekten stellt die absolute Mehrheit der Liste dar,

wenn das Element mindestens $\left(\frac{n}{2}+1\right)$ mal in der Liste vorkommt.

a) Definieren Sie eine **majority** Funktion, die das Majority-Element der Liste findet, wenn eines existiert oder sonst **Nothing** zurückgibt.

Die Signatur der Funktion soll wie folgt aussehen:

majority :: (Eq a) => [a] -> **Maybe** a

b) Analysieren Sie die Komplexität Ihrer Funktionsdefinition.

6. Aufgabe (10 Punkte)

Definieren Sie folgende Funktionen für den algebraischen Datentyp **Nat** aus der Vorlesung.

natUngerade :: Nat -> Bool

cutSub :: Nat -> Nat -> Nat -- a - b => 0, wenn b>=a

natMax :: Nat -> Nat -> Nat

natFib :: Nat -> Nat

Wichtige Hinweise:

1) **Zur jeder Funktion sollen Testfunktionen geschrieben werden.**

2) Verwenden Sie geeignete Namen für Ihre Variablen und Funktionsnamen, die den semantischen Inhalt der Variablen oder die Semantik der Funktionen wiedergeben.

3) Verwenden Sie vorgegebene Funktionsnamen, falls diese angegeben werden.

4) Kommentieren Sie Ihre Programme.

5) Verwenden Sie geeignete lokale Funktionen und Hilfsfunktionen in Ihren Funktionsdefinitionen.

5) Schreiben Sie in allen Funktionen die entsprechende Signatur.