Weitere Programmiersprache
- Serie 5 -

Aufgabe 5-1 (Funktionen)

Schreiben Sie folgende Funktion in Haskell.

a) `deleteVowels :: [Char] -> [Char]`, die alle Vokale aus der Liste löscht
b) `countElems :: Eq a => [a] -> [a] -> Int`, die ausgibt, wie oft Elemente aus der ersten Liste in der zweiten enthalten sind.

Aufgabe 5-2 (Pattern-Matching)

Lösen Sie die folgenden Aufgaben mit Pattern-Matching und ohne Zuhilfenahme von Standardfunktionen, die die geforderte Funktionalität implementieren.

a) Schreiben Sie eine Funktion `multipleElems :: [Integer] -> [Integer]`, die alle Elemente aus einer Liste rausfiltert, die nur einmal vorkommen
b) Schreiben Sie eine Funktion `descending :: [Integer] -> Bool`, die prüft, ob eine Liste absteigend sortiert – also jedes Element höchstens genau so groß wie sein Vorgänger – ist.
c) Schreiben Sie eine Funktion `myToLowerCase :: [Char] -> [Char]`, die die Großbuchstaben einer Zeichenfolge in Kleinbuchstaben umwandelt.
d) Schreiben Sie eine Funktion `smallestSum:: [Integer] -> Integer`, die aus einer Liste natürlicher Zahlen die kleinste Summer von 3 aufeinander folgenden Zahlen der Liste berechnet.

Aufgabe 5-3 (Abgabe!)

Die ersten 5 Zeilen des Pascal'schen Dreiecks lauten wie folgt:

```
1.        1
2.       1 1
3.      1 2 1
4.     1 3 3 1
5.    1 4 6 4 1
```

Die erste und letzte Zahl jeder Reihe ist 1. Jeder weitere Eintrag ergibt sich aus der Summe der beiden darüberstehenden Werte.

Schreiben Sie eine Funktion `pascal :: Int -> [Int]`, die Ihnen die n-te Reihe des Pascal'schen Dreiecks liefert, z.B. `pascal 3 = [1,2,1]` oder `pascal 5 = [1,4,6,4,1]`.

Aufg. 5.3 bitte bis 29.10. 23:59 Uhr in Moodle hochladen! Max. 2 Punkte!

Dateiname: Serie5.hs, als erste Zeile fügen Sie bitte ein: `module Serie5 where`