Weitere Programmiersprache
- Serie 3 -

Aufgabe 3-1 (Funktionen)

Schreiben Sie folgende Funktionen in Haskell.

a) `myLength :: [a] -> Integer -- Länge einer Liste`
b) `mySum :: Num a => [a] -> a -- Summe einer Liste von Zahlen`
c) `stringConcat :: [String] -> String`
   `-- Fügt eine Liste von Strings zusammen`

Aufgabe 3-2 (Typen 2)

Geben Sie zu jedem folgenden Typ einen Ausdruck an, ohne den Typ explizit einzuschränken. Beispiel: Gesucht ist eine Funktion, die `example :: Num a => (Bool, a)` erfüllt. Dann ist eine mögliche Lösung `example = (True, 5)`.

Hinweis: Die einzelnen Buchstaben können variieren.

1. `fun3 :: Eq a => (a, b, a) -> (b, a, Bool)`
2. `apply :: (a -> b) -> a -> b`
3. `applyList :: (a -> b) -> [a] -> [b]`
4. `applyString :: (Char -> Char) -> [Char] -> [Char]`
5. `fun4 :: Num a => a -> a -> b -> (a, b)`
6. `fun5 :: Num a => p -> [a -> a]`

Aufgabe 3-3 (Funktionen)

Schreiben Sie folgende Funktionen in Haskell.

```
-- Gibt für zwei Funktionen f und g und einer Definitionsmenge die
-- Schnittpunkte der Funktionen aus.
-- Bspw: intersect (*2) (\x -> x*x) [-2,-1,0,1,2] = [(0, 0), (2, 4)]
intersect :: Eq b => (a -> b) -> (a -> b) -> [a] -> [(a, b)]

-- Ergibt für die gegebenen Funktionen f1 und f2 und die Eingabe x:
-- f1(x) wenn f1(x) > f2(x), sonst f2(x)
-- z.B. higherFunction (+ 6) (* 2) 3 ergibt 9
higherFunction :: Ord b => (a -> b) -> (a -> b) -> a -> b

-- Entfernt Leerzeichen und zählt, wie viele Zeichen entfernt wurden
-- z.B.: nospace "Programmieren in Haskell" = ("ProgrammiereninHaskell", 2)
nospace :: String -> (String, Integer)
```


Besprechung: 16.10./21.10.

Aufgabe 3-4 (List-Comprehension)

Erzeugen Sie die folgenden Listen oder Ergebnisse mithilfe von List-Comprehension und der Liste `input = [1,-5,3,0,11,3,25,100,7].`

a) Liste, die alle Elemente der Liste `input` enthält, welche im Intervall [1,7] liegen.
b) Liste aller Zahlen von 1 bis 100, die durch 6 teilbar sind und mit 2 subtrahiert durch 4 teilbar sind.
c) Liste, die die quadrierten Elemente von `input` enthält, die größer als 100 und ungerade sind.
d) Liste, die alle Elemente aus `input`, die auch in a) sind, mit True und alle, die nicht in a) sind, mit False ersetzt.
e) Anzahl der Zahlen von b), die durch 5 teilbar sind, wenn man 1 von ihnen subtrahiert.

Aufgabe 3-5 (List-Comprehension)

Schreiben Sie folgende Funktionen mithilfe von List-Comprehension:

a) `containsSet :: [Int] -> [Int] -> Bool`, die prüft, ob die 2. Menge vollständig in der 1. enthalten ist.
b) `filterDigits :: [Int] -> Int -> [Int]`, die alle Zahlen aus der Liste wirft, die eine angegebene Ziffer nicht enthalten.
c) `deleteVowels:: [Char] -> [Char]`, die alle Vokale aus der Liste löscht
d) `myElem :: Eq a => a -> [a] -> Bool`, die prüft, ob ein Element in einer Liste vorkommt
e) `countElems :: Eq a => [a] -> [a] -> Int`, die ausgibt, wie oft Elemente aus der ersten Liste in der zweiten enthalten sind.

Besprechung: 16.10./21.10.