Weitere Programmiersprache
- Serie 7 -

Aufgabe 7-1 (MultiTrees)

Gegeben ist die folgende Datenstruktur, die einen Baum mit beliebig vielen Kindern beschreibt:

```
data MultiTree a = Node a [MultiTree a] deriving Show
```

Sie können folgenden Baum zum Testen verwenden:

```
testTree :: MultiTree Int
testTree = Node 1 [(Node 2 [(Node 4 [])]), (Node 3 []), (Node 8 [(Node 5 [(Node 6 []), (Node 7 [])])])]
```

1. Schreiben Sie eine Funktion, die alle Pfade von der Wurzel zu Blättern als Listen zurückgibt. Rückgabe für den Beispielbaum: `[[1,2,4], [1,3], [1,8,5,6], [1,8,5,7]]`

2. Gegeben ist eine Liste von Kanten in Form von Tupeln. Die Kantenliste für den Beispielbaum ist `[(1,2), (2,4), (1,3), (1,8), (8, 5), (5,6), (5,7)]`

   2.1 Schreiben Sie eine Funktion, die für eine gegebene Kantenliste die Wurzel des Baums zurückgibt (Hinweis: die Wurzel hat nur Kinder)

   2.2 Schreiben Sie eine Funktion, die für eine Kantenliste einen äquivalenten `MultiTree` zurückgibt. Der Beispielbaum aus 1. muss gleich dem durch diese Funktion generierten Baum sein.

Aufgabe 7-2 (Fehlerbehandlung)

Schreiben Sie eine Funktion, die in einer Liste beliebigen Typs `[a]` nach dem ersten Wert sucht, der ein gegebenen Prädikat `(a -> Bool)` erfüllt. Falls kein solcher Wert in der Liste vorhanden ist, soll die Fehlermeldung "`Kein passender Wert`" ausgegeben werden, sonst der Wert vom Typ `a`. Das Programm darf im „Fehlerfall" nicht abbrechen.

Aufgabe 7-3 (Funktionen auf Listen)

Schreiben Sie eine Funktion, die eine Liste als Parameter bekommt und alle gerade Zahlen der Liste durch die n-te ungerade Zahl ersetzt. Die ungeraden Zahlen sollen unverändert bleiben.

Beispiel: f [3,**2,2,2,2**,3,**4**] = [3,**1,3,5,7**,3,**9**] oder f [**2,2,2,2,2,2**] = [**1,3,5,7,9,11**]

Aufgabe 7-4 (Funktionen höherer Ordnung)

Schreiben Sie folgende Funktionen höherer Ordnung. Sie dürfen dabei auf die Funktionen `map` und `filter` zurückgreifen.

a) `series` bildet die Reihe zu einer gegebenen Funktion. Dabei soll die Anzahl der ausgegebenen Elemente und der Startwert der Reihe angegeben werden können.
b) `sumIf` summiert alle Elemente einer Liste von Zahlen, die ein bestimmtes Prädikat erfüllen.
c) `correctAnswers` überprüft bei einer Liste von Input,Output-Paaren (e,a) und einer gegebenen Funktion, ob die Funktion bei jeder Eingaben e die angegebene Ausgabe a hat:
`correctAnswers (*2) [(1,2), (6,12), (-3, -6)] = True`