



Übungen zur „Praktischen Informatik III“, WS 2008/09

Prof. Dr. R. Loogen · Fachbereich Mathematik und Informatik · Hans-Meerwein-Straße, D-35032 Marburg

Nr. 3, Abgabe: 5. November 2008 vor der Vorlesung

6. Listenabstraktionen

4 Punkte

Definieren Sie die folgenden Funktionen mit Hilfe von Listenabstraktionen:

- (a) `paarAdd :: Num a => [(a,a)] -> [a]` addiert jedes Zahlenpaar einer Liste. / 1
 Beispiel: `paarAdd [(1,2), (3,2), (3,4)] =>* [3,5,7]`
- (b) `ordPaare :: Ord a => [(a,a)] -> [(a,a)]` bestimmt die Teilliste der geordneten Paare einer Liste, d.h. der Paare, bei denen die erste Komponente kleiner oder gleich der zweiten Komponente ist. / 1
 Beispiel: `ordPaare [(1,2), (3,2), (3,4)] =>* [(1,2), (3,4)]`
- (c) `domino :: Eq a => [(a,a)] -> Bool` testet, ob in der Paarliste die zweite Komponente eines Paares jeweils identisch zu der ersten Komponente des nächsten Paares ist. Beispiel: `domino [(1,2), (3,2), (3,4)] =>* False`
`domino [(1,2), (2,3), (3,4)] =>* True` / 2
 Hinweis: Verwenden Sie neben der Listenabstraktion die Funktionen
`zip :: [a] -> [b] -> [(a,b)]` und `null :: [a] -> Bool`.

7. Algebraische Datenstrukturen

8 Punkte

Die folgende Datenstruktur repräsentiert aussagenlogische Formeln in Haskell (siehe Modul `Formula.hs` auf der Vorlesungsseite):

```
data Formula    = Var String | Not Formula
               | And Formula Formula | Or Formula Formula
```

Der Term `(And (Var "A") (Not (Var "B")))` stellt z.B. die Formel $A \wedge (\text{not } B)$ dar.

Eine Belegung von Variablen mit Wahrheitswerten kann als Liste von Paaren dargestellt werden: `type Assignment = [(String, Bool)]`.

- (a) Schreiben Sie eine Funktion `vars :: Formula -> [String]`, die eine Liste aller in einer Formel vorkommenden Variablen (ohne Duplikate) bestimmt. / 2
- (b) Definieren Sie eine Funktion `truthValue :: Assignment -> Formula -> Bool`, die den Wahrheitswert einer Formel in Abhängigkeit einer Belegung der in der Formel vorkommenden Variablen mit Wahrheitswerten bestimmt. / 2
- (c) Definieren Sie eine Funktion `assignments :: [String] -> [Assignment]`, die zu einer Liste von Variablen alle möglichen Belegungen der Variablen mit Wahrheitswerten berechnet. Zum Beispiel könnte `assignments ["A","B"]` die folgende Liste liefern: `[("A",True), ("B",True)], [("A",True), ("B",False)],`
`[("A",False), ("B",True)], [("A",False), ("B",False)]`. / 2
- (d) Schreiben Sie eine Funktion `truthTable :: Formula -> [(Assignment, Bool)]`, die zu einer Formel die Wahrheitstabelle in der durch den Typ festgelegten Form bestimmt. Zum Beispiel könnte `truthTable (And (Var "A") (Not (Var "B")))` die folgende Liste liefern:
`[(("A",True), ("B",True)), False], [("A",True), ("B",False)), True],`
`[(("A",False), ("B",True)), False], [("A",False), ("B",False)), False]` / 2