Übungen zur Rekursion

(*Rekursion*)

Stichwörter: Funktionale Programmierung mit Haskell

Implementiere in der Dateilists.hs eine Funktion `mylength(liste)`, die die Länge der übergebenen Liste berechnet. Die Funktion `myconcat(liste1, liste2)` soll als Ergebnis die Konkatenation der beiden Listen liefern. Mit `myappend(liste, elem)` soll das Element an das Ende der Liste angehängt werden. Die Funktion `listSum(liste)` soll die Summe aller Werte in der Liste zurückliefern. Verwenden Sie in dieser Aufgabe keine spezialisierten Listenfunktionen (wie z. B. den ++-Operator) außer dem :-Operator.

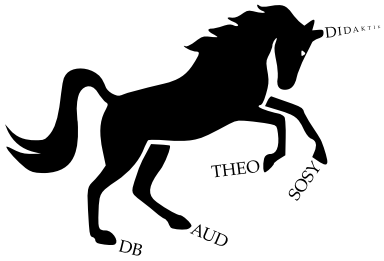
Lösungsvorschlag

```
myLength:: [a] -> Int
myLength [] = 0
myLength (x:xs) = 1+myLength(xs)

myconcat:: ([a],[a]) -> [a]
myconcat ([],ys) = ys
myconcat (xs,[]) = xs
myconcat ((x:xs),ys) = x : myconcat(xs,ys)

myappend:: ([Int],Int) -> [Int]
myappend ([],y) = [y]
myappend ((x:xs),y) = x:myappend(xs,y)

mySum:: [Int] -> Int
mySum [] = 0
mySum (x:xs) = x+mySum(xs)
```



Die Bschlangaul-Sammlung
Hermine Bschlangaul and Friends

Eine freie Aufgabensammlung mit Lösungen von Studierenden für Studierende zur Vorbereitung auf die 1. Staatsexamensprüfungen des Lehramts Informatik in Bayern.

Hilf mit! Die Hermine schafft das nicht allein! Das ist ein Community-Projekt! Verbesserungsvorschläge, Fehlerkorrekturen, weitere Lösungen sind herzlich willkommen - egal wie - per Pull-Request oder per E-Mail an hermine.bschlangaul@gmx.net.Der TEX-Quelltext dieses Dokuments kann unter folgender URL aufgerufen werden: `https://github.com/bschlangaul-sammlung/examens-aufgaben/blob/main/Module/60_FUMUP/30_Funktionale-Programmierung/Aufgabe_Rekursion.tex`