PHILIPPS-UNIVERSITÄT MARBURG
Fachbereich Mathematik und Informatik
Prof. Dr. R. Loogen

D-35032 Marburg
Hans Meerwein Straße
Lahnberge

Übungen zur „Praktischen Informatik III“, WS 2005/06

Nr. 6, Abgabe: 6. Dezember 2005 vor der Vorlesung

A. Hausaufgaben

Die Lösungen sollten grundsätzlich schriftlich, Programme zusätzlich auf Diskette oder per E-Mail an Ihre Tutorin abgegeben werden. Die Abgabe ist in Gruppen bis zu zwei Personen erlaubt.

24. `map` – `filter` – `foldr` **2 Punkte**

    (a) Drücken Sie die Listenabstraktion `[ f x | x <- xs, p x ]` mit den Funktionen `map` und `filter` aus. / 1

    (b) Definieren Sie die Funktion `filter` mittels `foldr`. / 1

25. Insertion Sort **4 Punkte**

    (a) Definieren Sie die in der Vorlesung vorgestellte Funktion `isort` (siehe Seiten 31 und 32 im Skript) mit `foldr`. / 1

    (b) Welches Problem tritt auf, wenn man auch die Hilfsfunktion `insert` als Instanz von `foldr` beschreiben möchte? / 1

    (c) Definieren Sie `insert` unter Zuhilfenahme der Funktion / 2

    ```
    insert' a bs = (bs, insert a bs)
    ```

    als Instanz von `foldr`. Welchen Typ hat die Funktion `insert'`?

26. Wertetabelle zu aussagenlogischer Formel **6 Punkte**

    Ausagenlogische Formeln können in Haskell durch die folgende Datenstruktur repräsentiert werden:

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

    Die Formel $A \wedge (A \vee B)$ wird durch den folgenden Term dargestellt:

    ```
    And (Var "A") (Or (Var "A") (Var "B"))
    ```

    (a) Schreiben Sie eine Funktion `vars :: Formula -> [String]`, die zu einer Formel eine Liste aller in der Formel vorkommenden Variablen bestimmt. / 1

    (b) Eine Belegung von Variablen mit Wahrheitswerten kann als Liste von Paaren dargestellt werden:

    ```
    type Assignment = [(String,Bool)]
    ```

Definieren Sie eine Funktion `assignments :: [String] -> [Assignment]`, die zu einer Liste von Variablen alle möglichen Belegungen der Variablen mit Wahrheitswerten berechnet. / 2

Zum Beispiel soll der Aufruf `assignments ["A","B"]` die folgende Liste liefern:

```
[ [("A",True), ("B",True)],
  [("A",True), ("B",False)],
  [("A",False), ("B",True)],
  [("A",False), ("B",False)] ]
```

(c) 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. / 1

(d) Schreiben Sie eine Funktion `truthTable :: Formula -> [(Assignment,Bool)]`, die zu einer Formel die Wahrheitstabelle in der durch den Typ festgelegten Form bestimmt. / 1

(e) Schreiben Sie eine Faltungsfunktion `foldFormula` und drücken Sie die Funktionen `vars` und `truthValue` als Instanzen dieser Funktion aus. / 1

B. Mündliche Aufgaben

27. Listenspiegelung als Listenfaltung

Die Listenspiegelung `reverse :: [a] -> [a]` kann auch mit `foldl` definiert werden:

```
reverse xs = foldl revOp [] xs
```

(a) Geben Sie die Definition des hierzu benötigten Faltungsoperators `revOp` an.

(b) Zeigen Sie die Korrektheit der so definierten Listenspiegelung.

(c) Analysieren Sie die Zeitkomplexität dieser Definition.

28. Zeilennummern

Schreiben Sie eine Funktion `addLineNumbers :: String -> String`, die einen Text um Zeilennummern erweitert.

Hinweis: Verwenden Sie die vordefinierten Funktionen `lines`, `unlines`, `zip` und `map`.

29. `curry` – `uncurry`

Definieren Sie unter Verwendung von `curry`, `uncurry` und der Funktionskomposition eine Funktion

```
sprod :: Num a => [a] -> [a] -> a
```

die das Skalarprodukt zweier als Zahlenlisten dargestellter Vektoren berechnet.