Prof. Dr. Uwe Schmidt

fhwedel
UNIVERSITY OF APPLIED SCIENCES

4. Februar 2020

Aufgaben zur Klausur **Grundlagen der funktionalen Programmierung** im WS 2019/20 (BInf)

Zeit: 90 Minuten erlaubte Hilfsmittel: keine

Bitte tragen Sie Ihre Antworten und fertigen Lösungen ausschließlich an den gekennzeichneten Stellen in das Aufgabenblatt ein. Ist ihre Lösung wesentlich umfangreicher, so überprüfen Sie bitte nochmals Ihren Lösungsweg.

Nutzen Sie die Rückseiten der Klausur zur Entwicklung der Lösungen und übertragen die fertigen Lösungen in das Aufgabenblatt.

Sollten Unklarheiten oder Mehrdeutigkeiten bei der Aufgabenstellung auftreten, so notieren Sie bitte, wie Sie die Aufgabe interpretiert haben.

Viel Erfolg!

Diese Klausur besteht einschließlich dieses Deckblattes aus 7 Seiten.

Aufgabe 1:

Definieren Sie eine Funktion

$$dec2int :: [Int] \rightarrow Int$$

die eine Liste von Ziffern in eine Dezimalzahl konvertiert. Ein Beispiel:

$$dec2int\ [2, 3, 4, 5] = 2345$$

Entwickeln Sie eine erste Version der Funktion, ohne Funktionen höherer Ordnung zu verwenden. Tipp: Nutzen Sie eine Hilfsfunktion mit einem zusätzlichen Parameter zum Akkumulieren des Resultats.

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

Entwickeln Sie eine zweite Version, die mit `foldl` arbeitet.

. . . . . .

. . . . . .

. . . . . .

Aufgabe 2:

Die Funktion `zip` verarbeitet zwei Listen, indem Sie aus den Elementen der beiden Listen Paare bildet. Die Länge der Resultatliste ist das Minimum der Längen der beiden Argumentlisten. Entwickeln Sie diese Funktion.

$zip :: [a] \to [b] \to [(a, b)]$

. . . . . .

. . . . . .

. . . . . .

. . . . . .

Verallgemeinern Sie diese Funktion zu einer `zipWith`-Funktion, bei der die Elemente mit Hilfe einer zweistelligen Funktion zu einem neuen Wert verknüpft werden.

Die Signatur (der Typ) von `zipWith`:

. . . . . .

Die Funktionsdefinition von `zipWith`:

. . . . . .

. . . . . .

. . . . . .

Reimplementieren sie `zip` mit `zipWith`:

$zip' :: [a] \to [b] \to [(a, b)]$

. . . . . .

Aufgabe 3:

Gegeben sei der folgenden Datentyp für einen Baum mit zwei Arten von inneren Knoten und einer Form von Blättern.

$$\begin{aligned}&\textbf{data}\ E\ a = Add\ (E\ a)\ (E\ a)\\ &\quad \mid Mul\ (E\ a)\ (E\ a)\\ &\quad \mid Val\ a\end{aligned}$$

Entwickeln Sie eine Funktion `mapE` analog zu `map` für Listen und `mapTree` für das Baum-Beispiel aus der Vorlesung (einschließlich des Typs der Funktion).

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

Mit diesem Datentyp lassen sich arithmetische Ausdrücke mit $+$ und $*$ und Zahlen repräsentieren.

Beispiel: $5 * (3 + 4) + 1$ kann durch `e1` repräsentiert werden (Skizze):

$$\begin{aligned}&e1 :: Num\ a \Rightarrow E\ a\\ &e1 = Add\ (Mul\ (Val\ 5)\ (Add\ (Val\ 3)\ (Val\ 4)))\ (Val\ 1)\end{aligned}$$

Entwickeln Sie eine Auswerte-Funktion `evalE`, mit der solche Ausdrücke ausgewertet werden können.

$$evalE :: Num\ a \Rightarrow E\ a \rightarrow a$$

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

Entwickeln Sie eine `show`-Funktion `showE`, die einen Ausdruck in eine lesbare Textform transformiert. Die Ausdrücke sollen dabei vollständig geklammert werden. `showE e1` liefert den String `s1 = "((5*(3+4))+1)"`. Es gilt hier: `length s1 == 13`.

Die Funktion `showE` einschließlich Typ:

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

Die beiden Funktionen `evalE` und `showE` haben eine ähnliche Struktur. Die Ausdrücke werden auf die gleiche Art und Weise traversiert, sie unterscheiden sich nur in den Aktionen an den Blättern und der Kombination der Resultate an den inneren Knoten (analog zum Baum-Beispiel aus der Vorlesung).

Entwickeln Sie eine Funktion `foldE` (analog zu `foldTree`), die einen Ausdrucksbaum zu einem Ergebnis zusammen faltet, bei der die Aktionen an den einzelnen Knoten aber als zusätzliche Parameter mitgegeben werden.

Die Funktion `foldE` benötigt neben dem zu verarbeitenden Baum drei zusätzliche Parameter. Warum genau drei zusätzliche Parameter?

. . . . . .

Welchen Typ besitzt `foldE`?

. . . . . .

. . . . . .

. . . . . .

Die Definition von `foldE`:

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

Definieren Sie `evalE` mit Hilfe von `foldE`:

. . . . . .

. . . . . .

. . . . . .