HTWK Leipzig, Fakultät IM

Prof. Dr. Sibylle Schwarz `sibylle.schwarz@htwk-leipzig.de`

Arbeitpaket zu KW 17 zum Modul „Fortgeschrittene Programmierung“
Sommersemester 2020

In dieser Woche geht es darum, wie in Haskell einfache Funktionen deklariert, aufgerufen und diese Aufrufe ausgewertet werden. Wichtig dabei sind die Termersetzung als das der funktionalen Programmierung zugrundeliegende Berechnungsmodell und Pattern Matching zur Feststellung, welche Ersetzungsregeln dabei angewendet werden.

Lesen und Verstehen

Kapitel 3 Haskell-Grundlagen (außer 3.3, das kommt später noch ausführlicher)
im Buch „Haskell-Intensivkurs - Ein kompakter Einstieg in die funktionale Programmierung“
(`https://link.springer.com/content/pdf/10.1007%2F978-3-642-04718-3.pdf`)
Überprüfen Sie dabei wieder alle Beispiele mit `ghci`.
Da das Thema Termersetzung im Buch zu kurz kommt, konsultieren Sie dazu bitte die Folien zur Termersetzung auf der Homepage zum Modul.

Begriffe

Substitution, Termersetzungsregel, Termersetzungssystem (TRS), Normalform,
Konstruktor, definiertes Symbol, Konstruktor-TRS
Pattern, Pattern Matching, Guard,
rechtsassoziativ, linksassoziativ,

Testfragen

1) Für jeden der Funktionstypen `f :: Int -> Bool -> Char`,
   `g :: (Int -> Bool) -> Char`, `h :: Int -> (Bool -> Char)`
   Wieviele Argumente hat jede Funktion des Typs und welche Typen haben diese Argumente? Geben Sie je ein Beispiel für eine Aufruf dieser Funktionen an.

2) Was ist Pattern Matching ?

3) Welche Formen von Pattern Matching können in Funktionsdeklarationen vorkommen?

4) Welche Wirkung hat die Reihenfolge der Zeilen in einer Funktionsdeklarationen aus mehreren Mustern ?

5) Wofür steht `_` in Funktionsdeklarationen ?

6) Was bedeutet `=>` in Typdeklarationen ?

7) Haben Einrückungen am Zeilenbeginn in Haskell-Programmen eine semantische Bedeutung ?

8) Welche Vor- und Nachteile hat dies ?

9) Stellen Sie Vor- und Nachteile lokaler Deklarationen durch `where` und `let` gegenüber.

10) Was sind Guards? Wie und wozu werden sie eingesetzt?

11) Was bedeutet `otherwise` ? Wie wird es ausgewertet ?

12) Welche Möglichkeiten gibt es, Fallunterscheidungen in Haskell darzustellen ?

13) Welche Möglichkeiten gibt es in Haskell, mehrere Funktionen desselben Typs zu deklarieren ?

14) Was bedeutet Assoziativität, Rechtsassoziativität und Linksassoziativität zweistelliger Funktionen ? Wie wirken sich diese Eigenschaften auf die Darstellung der mehrstelligen Erweiterung dieser Funktionen aus ?

15) Welche Assoziativität erfüllt `->` in Haskell-Typen?

16) Welche Assoziativität erfüllt die logische Implikation $\rightarrow$?

17) Lassen sich Präfix-definierte Funktionen auch in Infix-Form aufrufen ? Wenn ja, wie ?

18) Lassen sich Infix-definierte Funktionen auch in Präfix-Form aufrufen ? Wenn ja, wie ?

Übungsaufgaben

Serie 2

Autotool

tolle Aufgaben zur Termersetzung