Institut für Informatik **Ludwig-Maximilians-Universität München**
Prof. François Bry, Thomas Prokosch
Kilian Rückschloß

Programmierung und Modellierung, SoSe 21
Übungsblatt 1

Abgabe: Mo, 19. April 2021 08:00 Uhr

Im Rahmen der Übungen werden wir Ihnen immer wieder Dateien zur Verfügung stellen. Da dies, in der Regel, mehrere pro Übungsblatt sind, geschieht dies gesammelt als ZIP-Archiv. Für dieses Übungsblatt handelt es sich um die Datei `u01.zip`.

Entpacken Sie diese Datei mit
`unzip u01.zip`
Dies erstellt ein Verzeichnis `u01`, in dem sich pro Teilaufgabe ein Unterverzeichnis befindet. In diesen Unterverzeichnissen können Sie die Teilaufgaben bearbeiten. Wenn Sie mit der Bearbeitung fertig sind, packen Sie mit
`zip -r u01.zip u01`
wieder alles zusammen und geben die so erhaltene Datei `u01.zip` ab.
Die Abgaben sollten nur Haskell-Programm Dateien (`.hs`), Dateien im Plain Text (`.txt`), Markdown (`.md`) oder im Portable Document Format (`.pdf`) enthalten. Abgaben mit in anderen Dateiformaten werden nicht korrigiert.

Aufgabe 1-1 Software Installation

Installieren Sie auf Ihrem Computer die Haskell-Plattform (Glasgow Haskell Compiler – GHC und GHCI) und einen passenden Editor. Auf der Webseite der Vorlesung finden Sie entsprechende Anleitungen unter dem Punkt "7. Software".

Aufgabe 1-2 Ausdrücke und Typen

Werten Sie folgende Ausdrücke aus und bestimmen Sie deren Typen.

Hinweis: Sie können jede Zeile in den Interpreter GHCI eingeben und die Reaktion beobachten.

a) `50 * 100 - 4999`

b) `50 * (100 - 4999)`

c) `1-200/0`

d) `let fuenf = 5;`

e) `let acht = 008;`

f) `let x10 = 0x10;`

g) `let fuenf' = -(-5)` Wie finde ich heraus, wie eine Funktion definiert ist (und auf welchen Datentypen)?

Hoogle

h) `let fuenftel = 1.0 / fromIntegral(fuenf)`

i) `(fuenf == acht) == (fuenf > acht)`

j) `fuenf /= 5`

k) `let a = 'a'`

l) `let bc = "c"`

m) `let abc = a:bc`

n) `let begruessung = "hallo " ++ "welt!"`

o) `let begruessung' = begruessung ++ " gut gem" ++ show(acht)`

p) `read(show(fuenftel)) + fuenftel`

Aufgabe 1-3 Erste Funktionen

Hinweis: Sie können die Funktionen in die Datei `erste-funktionen.hs` im Verzeichnis `u01/1-3/` einfügen. Um diese zu laden, geben Sie im `ghci` den Befehl
`Prelude> :l erste-funktionen.hs`
ein. Danach können Sie die Funktionen aufrufen.

a) Definieren Sie eine Funktion '`dreifach :: Integer -> Integer`', die einen Integer Wert als Eingabe erhält und dessen dreifachen Wert zurück gibt.

b) Definieren Sie ein Funktion '`vierfach :: Integer -> Integer`', die einen Integer Wert als Eingabe erhält und dessen vierfachen Wert zurück gibt.

Aufgabe 1-4 Listen

Werten Sie folgende Ausdrücke aus und bestimmen Sie deren Typen.

a) `[1,2,3,4] ++ [9,10,11,12]`

b) `[3,4,2] > [2,4]`

c) `[1,2,3,4,5,6,7,8] !! 5 == 5`

d) `null [1,2,3]`

e) `length [5,4,3,2,1]`

f) `head []`

g) `tail [2,3,4]`

h) `'H':"-Milch"`

i) `"Haskell" !! 2`

j) Definieren Sie eine Funktion '`kopf`', die das erste Element eines Strings ausgibt.

k) Definieren Sie eine Funktion '`ende`', die das letzte Element eines Strings ausgibt.

l) Definieren Sie eine Funktion '`rest`', die bis auf das erste Element alle Elemente eines Strings ausgibt.

Aufgabe 1-5 List-Comprehension

Gegeben sind die zwei untenstehenden Listen 'substantive' und 'adjektive'. Berechnen Sie mit Hilfe einer list-comprehensions alle grammatikalisch sinnvollen Tupel-Kombinationen der Elemente der Listen.

```
let substantive = ["Student","Professor","Tutor"]
let adjektive = ["fauler","fleissiger","hilfreicher"]
```

Grundsätzliche Erklärung der List-Comprehension Syntax, wie sie auf der letzten Vorlesungsfolie besprochen wird; Prädikate und (v.a.) Fallunterscheidungen nur am Rand, werden auf späterem Übungsblatt besprochen.