technische universität dortmund

FLDIT

Lehrstuhl 1
Logik in der Informatik

Prof. Dr. Peter Padawitz

Jos Kusiek (*jos.kusiek@tu-dortmund.de*)
Lukasz Czajka (*lukasz.czajka@tu-dortmund.de*)

Wintersemester 2018/2019

Übungen zu Funktionaler Programmierung
Präsenzblatt

Ausgabe: 8.10.2018, **Abgabe:** keine Abgabe, **Block:** N.N.

Aufgabe 0.1 *Einführung in den* `GHCi`

Installieren Sie die Haskell-Platform (`https://www.haskell.org/platform/`) auf ihrem Rechner. Stellen Sie dabei sicher, dass `ghc` und `ghci` zu ihrer Pfadvariablen hinzugefügt sind.

a) Öffnen Sie den Texteditor Ihrer Wahl und tippen Sie folgendes Programm ab:

```
f :: Int -> Int -> Int -> Int
f x y z = x + y + z * z
```

b) Speichern Sie das Programm in einer Datei mit der Endung `.hs`. Den Pfad zu der Datei nennen wir im Folgenden `file.hs`.

c) Öffnen Sie die Kommandozeile und laden Sie die Datei mit dem interaktiven Modus des Glasgow Haskell Compiler (GHCi genannt), wie folgt: `ghci file.hs`
Sie sollten nun die folgende Ausgabe erhalten:

```
[1 of 1] Compiling Main             ( file.hs, interpreted )
Ok, modules loaded: Main.
*Main>
```

d) Rufen Sie nun die Funktion `f` auf, indem Sie zum Beispiel `f 1 2 3` eingeben und mit ENTER bestätigen. Das Ergebnis wird ausgegeben und Sie können weitere Funktionsaufrufe auswerten lassen.

Folgende Kommandos des GHCi haben sich als nützlich erwiesen:

- `:load file` (kurz `:l`) lädt die Datei `file` in den GHCi.
- `:reload` (kurz `:r`) lädt die aktuelle Datei neu ein. Nachdem Änderungen an dem Quelltext vorgenommen wurden, kann die aktuelle Datei mit `:r` leicht neu geladen werden.
- `:type ausdruck` (kurz `:t`) zeigt den Typ des Ausdruckes `ausdruck` an, z. B. `:t f` oder `:t f 1 2 3`.
- `:kind typ` (kurz `:k`) zeigt den Kind des Typs `typ` an, z. B. `:k Int` oder `:k []`.
- `:info name` (kurz `:i`) zeigt umfangreiche Informationen zu `name` an, z. B. `:i True`, `:i Bool` oder `:i Eq`.
- `:help` (kurz `:h`) öffnet die Hilfe mit weiteren nützlichen Befehlen.
- `:quit` (kurz `:q`) beendet den GHCi.

Aufgabe 0.2 *Fehlermeldungen des GHCi*

Die folgende Aufgabe enthält eine Reihe von fehlerhaften Haskell-Ausdrücken. Ziel dieser Aufgabe ist, dass Sie sich mit den Fehlermeldungen des GHCi vertraut machen. Laden Sie dazu die Datei aus Aufgabe 1 und interpretieren Sie die folgenden Ausdrücke mit dem GHCi. Versuchen Sie die Fehlermeldungen nachzuvollziehen.

a) `f 3 1 True`

b) `f 4 3 2 1`

c) `f 2 1`

d) `foo 3 2 1`

Aufgabe 0.3 *Painter-Paket*
Laden und Entpacken Sie das *Painter-Paket* von der Moodle-Vorlesungsseite (`https://moodle.tu-dortmund.de/course/view.php?id=12903`) im Abschnitt *Übungen*. Das Modul `Examples` stellt die meisten in der Vorlesung vorgestellten Definitionen bereit.

a) Laden Sie die Datei `Examples.hs` in den GHCi.

b) Sie können Module mit der Anweisung `import` laden. Diese Anweisung kann im GHCi ausgeführt werden oder am Anfang einer Haskell-Datei (`.hs`) stehen. Legen Sie eine Haskell-Datei an. Beginnen Sie die Datei mit `import Examples` und laden Sie die Datei in GHCi. Alle Dateien müssen sich in dem gleichen Ordner befinden.

Aufgabe 0.4 *Einführung in den GHC*
Mit dem Glasgow Haskell Compiler kann man auch ausführbare Dateien erzeugen. Dazu *muss* eine Funktion `main` vom Typ `IO ()` als Einstiegspunkt existieren. Speichern Sie folgendes Programm in einer Haskell-Datei:

```
main :: IO ()
main = putStrLn "Hello, world!"
```

Übersetzen Sie das Programm mit `ghc file.hs`. Es entsteht eine ausführbare Datei mit gleichem Namen (`file`) und der Dateiendung `.exe` bzw. keiner Endung, je nach Betriebssystem. Führen Sie diese Datei aus.

Aufgabe 0.5 *Hackage*
Besuchen Sie die Seite `https://hackage.haskell.org/`. Suchen Sie dort nach dem Paket `base`. Finden Sie in dem Paket das Modul `Prelude`. Hier finden Sie die Dokumentationen zu allen Funktionen, Datentypen, etc. die Ihnen automatisch in Haskell zur Verfügung stehen.

Sie können weitere Module aus dem Paket `base` oder anderen Paketen mit der Anweisung `import` nutzen. Eine vollständige Liste aller Pakete und damit aller Module der Haksell-Plattform finden Sie unter `https://www.haskell.org/platform/contents.html`. In der Veranstaltung werden lediglich die Module aus dem *Painter-Paket* und dem *base-Paket* (`https://hackage.haskell.org/package/base`) benutzt.

Aufgabe 0.6 *Hoogle*
Besuchen Sie die Seite `https://www.haskell.org/hoogle/`. Hier können Sie nach Funktionen, Datentypen und mehr suchen. Finden Sie heraus, was der Operator `$` macht.