Praktische Informatik 3 WS 12/13

Christoph Lüth
Marcus Ermler
Christian Maeder
Martin Ring
Diedrich Wolter

0. Übungsblatt

Ausgabe: 17.10.12
Abgabe: —

Dieses Übungsblatt dient zum Warmwerden mit Haskell. Eine Abgabe ist nicht vorgesehen, Punkte gibt's auch nicht. Nutzen Sie es dazu, sich am besten schon im Tutorium mit der Benutzung der Sprache (Syntax, Interpreter/-Compiler, Fehlermeldungen etc.) vertraut zu machen.

Erste Schritte mit Haskell

Der Unterschied zwischen Theorie und Praxis ist in der Praxis größer als in der Theorie. Wir empfehlen, die folgenden Schritte zu befolgen, um eine lauffähige Haskell-Umgebung auf Ihrem System (bzw. einem Uni-Rechner) einzurichten und die Übungsaufgaben lösen zu können.

1. Laden Sie unter `http://hackage.haskell.org/platform/` die *Haskell Platform* herunter.
   - Für Windows: Wählen Sie die Windows Installer-Variante (`HaskellPlatform-2012.2.0.0-setup.exe` o.ä.) und befolgen Sie die Schritte. Achten Sie darauf, den `PATH` während der Installation (oder danach) auf das Installationsverzeichnis der Haskell Platform (in dem `ghci.exe` liegt) zu erweitern.
   - Für Linux/Mac: GHC per Paketmanager wie unter `http://www.haskell.org/` beschrieben installieren.
   - Für Nutzer des FB3-Rechnerpools: Der GHCi ist vorinstalliert und kann insbesondere auf x-Rechnern per `ghci` direkt gestartet werden.
2. Erstellen Sie ein Verzeichnis, in dem Sie die übungsrelevanten Dateien ablegen (z. B. `~/PI3/Uebung00`).
3. Erstellen Sie mit einem Editor Ihrer Wahl (Vorschläge: Emacs, gedit (Linux), Notepad++ (Windows)) eine Datei `Aufgabe00.hs` im erzeugten Verzeichnis.
4. Fügen Sie folgenden Code (aus der Vorlesung) ein:

   ```
   fac :: Int -> Int
   fac n = if n <= 0 then 1 else n * fac (n - 1)
   ```

   Fügen Sie später ggf. weitere Definitionen hinzu.
5. Starten Sie eine Kommandozeile/Terminal/Konsole (Start… Ausführen… `cmd.exe` unter Windows, `bash` o.ä. unter Linux)
6. Wechseln Sie in das erzeugte Verzeichnis (`cd .../Uebung00`)
7. Starten Sie den Haskell-Interpreter `ghci`
8. Laden Sie Ihre Testdatei `Aufgabe00.hs` in den Interpreter: `:load Aufgabe00` bzw. kürzer `:l Aufgabe00`.
9. Der Interpreter erlaubt Ihnen, die in der geladenen Datei definierten Funktionen und Variablen auszuwerten. Geben Sie hierzu entsprechende Haskell-Ausdrücke auf der Kommandozeile des Interpreters ein. Beispiele:

   ```
   17 + 4                100 / 7
   sqrt 100 - 1          (-1) * (-1)
   fac 10                fac (-3)
   fac "kolibri"         fac 9 - 20
   ```

   Beachten Sie auch die hilfreichen (teilweise noch unverständlichen) Fehlermeldungen bei fehlerhaften Ausdrücken.

10. Hilfe bekommen Sie durch Eingabe von `:help`. Die meisten Optionen sind anfangs noch nicht relevant; als hilfreich werden sich aber die Kommandos `:reload` (Abk.: `:r`), `:type` (`:t`) und `:browse` (Sie erkennen das Schema – `:b`) erweisen. Ein üblicher Entwicklungszyklus sieht dann so aus: Editieren und Speichern im Editor, (neu) laden im Interpreter (`:l` bzw. `:r`), Fehlermeldungen analysieren bzw. Testeingaben machen, zurück zum Editor.

11. Die PI3-Webseite bietet nützliche Links zu den Themen Haskell lernen, API-Dokumentation, etc.

So gewappnet können Sie jetzt die folgende Aufgabe angehen (am besten gemeinsam im Tutorium):

0.1 *Größter Gemeinsamer Teiler* *0 Punkte*

Der größte gemeinsame Teiler zweier natürlicher Zahlen kann rekursiv wie folgt berechnet werden:

$$ggt(m,n) = \begin{cases} ggt(n,m) & \text{wenn } m < n \\ m & \text{wenn } n = 0 \\ ggt(n, m \bmod n) & \text{sonst} \end{cases}$$

1. Implementieren Sie eine Funktion `ggt` in Haskell. Welche Signatur muss diese Funktion haben, und wie behandeln Sie ungültige Eingabewerte (gibt es welche)?

2. Speichern Sie Ihre Definition in der Datei `Aufgabe00.hs` (oder einem beliebigen anderen Namen) und laden Sie die Datei.

3. Überlegen Sie sich Testfälle (Testwerte) für Ihre Implementation, und führen Sie diese aus, indem Sie die Auswertung auf der Kommandozeile wie in den *Ersten Schritten* beschrieben nutzen.