universität innsbruck

Funktionale Programmierung	WS 2026/2027	LVA 703206/703025
Übungsblatt 0, 0 Punkte		Abgabefrist: wird im 1. Proseminar besprochen

Aufgabe 1 *Erste Schritte mit Haskell* **0 P.**

1. Installieren Sie Haskell (inkl. Cabal), wie auf https://cl-teaching.uibk.ac.at/26w/fp/ghc_setup.php beschrieben. Empfohlen ist die Installation mittles `ghcup` (https://www.haskell.org/ghcup/).

   Führen Sie Schritt 1 am besten schon vor dem Proseminar am 14. Oktober durch, da die Installation etwas Zeit benötigt. Alle weiteren Schritte werden dann im Proseminar besprochen.

2. Starten Sie einen kurzen Test mit Haskell.
   - Öffnen Sie eine Konsole (Powershell, terminal, ...), entweder auf dem eigenen Rechner, oder auf dem Computer vor Ihnen. Im zweiten Fall wird Ihr(e) Proseminar-LeiterIn Ihnen die richtigen Anweisungen für die Computer in Ihrem Raum nennen.
   - Starten Sie den *G*lasgow *H*askell *C*ompiler im *I*nteraktiven Modus (GHCI) mit dem Befehl `ghci`.
   - Werten Sie einen Ausdruck aus, etwa `3 + 4 * 2`.
   - Beenden Sie GHCI mittels `:q`

   Das Resultat sollte dann etwa wie folgt aussehen:

   ```
   shell$ ghci
   GHCi, version 9.12.2: https://www.haskell.org/ghc/  :? for help
   Loaded GHCi configuration from /Users/rene/.ghci
   ghci> 3 + 4 * 2
   11
   ghci> :q
   Leaving GHCi.
   shell$
   ```

3. Testen Sie Haskell mit eigenen Programmen.
   - Erstellen Sie ein einfaches Haskell Programm, indem Sie eine Datei names `Test1.hs` mit folgendem Inhalt anlegen.

     ```
     square x = x * x
     ```

   - Starten Sie GHCI mit dem Programm und berechnen Sie 748939403^2. Beachten Sie, dass das Programm und die Konsole sich im gleichen Verzeichnis befinden müssen. Nutzen Sie `cd` oder `pwd`, um zum richtigen Verzeichnis zu wechseln.
   - Das Resultat bis jetzt sollte in etwa so aussehen.

     ```
     shell$ ghci Test1.hs
     GHCi, version 9.12.2: https://www.haskell.org/ghc/  :? for help
     Loaded GHCi configuration from /Users/rene/.ghci
     [1 of 2] Compiling Main             ( Test1.hs, interpreted )
     Ok, one module loaded.
     ghci> square 748939403
     560910229365996409
     ghci>
     ```

- Ergänzen Sie Ihr Programm, in dem Sie eine weitere Zeile einfügen, die x^4 berechnet.

  ```
  pow4 x = undefined -- replace "undefined" with your own definition
  ```

- Speichern Sie Ihre Datei, laden Sie das Programm erneut mittels `:r`, und berechnen Sie $(9032^4)^2$.

  ```
  ghci> :r
  [1 of 2] Compiling Main             ( Test1.hs, interpreted ) [Source file changed]
  Ok, one module reloaded.
  ghci> square (pow4 9032)
  4428650737869115537342550612377 6
  ghci> :q
  shell$
  ```

Aufgabe 2 *Erste Schritte mit Cabal und OLAT* **0 P.**

Für jedes Übungsblatt gibt es vorgefertigte Haskell-Dateien, die Sie auf der FP-Webseite herunterladen können:

https://cl-teaching.uibk.ac.at/26w/fp/exercises.php

1. Laden Sie die ZIP-Datei für Woche 00 herunter und extrahieren diese in einem Verzeichnis Ihrer Wahl. Wechseln Sie in das Verzeichnis `fp_ps_00`.

2. Die Datei `src/Template_00.hs` ist immer die Datei, die Sie hauptsächlich bearbeiten müssen. Der obere Teil der Datei darf *nicht* verändert werden.

3. Editieren und lösen Sie die Aufgabe 1 im Template: definieren Sie `poly` so, dass $x^3 + 2x^2 - 5$ berechnet wird.

4. Sie können jederzeit Ihre Bearbeitung mittels `ghci src/Template_00.hs` und eigenen Aufrufen testen.

5. Am Ende können Sie noch von FP-Team bereitgestellte Tests ausführen, um zu sehen, ob sich Ihr Programm richtig verhält. Dazu starten Sie folgenden Befehl.

   ```
   shell$ cabal run
   ```

   Programmteile, die Sie nicht bearbeitet oder gelöst haben, lassen Sie einfach im Template unverändert (hier: Aufgabe 2).

6. Abschließend laden Sie *nur* die Datei `Template_00.hs` in OLAT hoch und setzen ein Kreuz bei Aufgabe 1.

7. Wichtig: Ihre Template-Datei muss ohne Fehler compilierbar sein, d.h. `ghci src/Template_00.hs` sollte zu keiner Fehlermeldung führen. Notfalls entfernen Sie Teile des Programms oder nutzen Dummy-Implementierungen wie

   ```
   knifflig x = error "Leider nicht geschafft"
   ```

8. Hinweis: es müssen nicht alle Tests erfolgreich bestanden werden, damit Sie ein Kreuz bei einer Aufgabe setzen können. Sie sollten aber in der Lage sein, Ihr Programm zu erklären, und zumindest einige Tests bestehen.