Hinweise zu Organisation und Ablauf des Übungsteil der LVA 185.A03 Funktionale Programmierung im WS 2017/18

Programmiermöglichkeit im Labor "Argentinierstr."

Für die Bearbeitung, Lösung und Abgabe der Programmieraufgaben stehen Ihnen (im Rahmen der Kapazitäten) die Rechner im Labor im Erdgeschoss des Institutsgebäudes Argentinierstraße 8 zur Verfügung. Sie erreichen dieses Labor über den kleinen Innenhof im Erdgeschoss. Einen Lageplan finden Sie auf

`http://www.complang.tuwien.ac.at/ulrich/p-1851-E.html` .

Um die Aufgaben zu lösen, rufen Sie bitte den Hugs 98-Interpretierer durch Eingabe von `hugs` in der Kommandozeile einer Shell auf. Falls Sie die Übungsaufgaben auf Ihrem eigenen Rechner bearbeiten möchten, müssen Sie zunächst Hugs 98 installieren. Hugs 98 ist beispielsweise unter `www.haskell.org/hugs/` für verschiedene Plattformen verfügbar. Der Aufruf der jeweiligen Interpretierervariante ist dann vom Betriebssystem abhängig.

Ausgabe und Abgabe von Aufgaben

Neue Aufgabenblätter werden regelmäßig mittwochs auf der Webseite

`http://www.complang.tuwien.ac.at/knoop/lehre/ws1718/fp185A03`

zur Lehrveranstaltung ausgegeben und können dort heruntergeladen werden.

Die Abgabe der Lösungen zerfolgt regelmäßig am auf den Ausgabetag folgenden Mittwoch bis spätestens 15:00 Uhr, ausschließlich auf der Maschine

`g0.complang.tuwien.ac.at`,

jeweils in Ihrem Homeverzeichnis in einer Datei, deren Name auf dem jeweiligen Aufgabenblatt vorgegeben ist.

Für Zweitabgaben siehe Abschnitt "Zweitabgabe von Aufgabenlösungen".

Benachrichtigung über erreichte Punkte

Je Aufgabenblatt sind insgesamt **100** Punkte zu erreichen. Die Ergebnisse der Bewertung Ihrer abgegebenen Lösungen wird in einer Datei in Ihrem Homeverzeichnis abgelegt.

Zweitabgabe von Aufgabenlösungen

Für jedes Aufgabenblatt können Sie eine zweite Lösung nach ggf. vorgenommenen Verbesserungen abgeben.

Erhalten Sie die Bewertung für die Erstabgabe zu einem Aufgabenblatt bis spätestens 9:00 Uhr am auf den Ausgabetag folgenden Montag, können Sie eine verbesserte Lösung bis spätestens 15:00 Uhr am unmittelbar darauf folgenden Mittwoch abgeben.

Erhalten Sie die Bewertung Ihrer Erstabgabe nicht bis zu diesem Zeitpunkt, verlängert sich die Zeit für die Zweitabgabe um eine Woche auf den Mittwoch der Folgewoche, ebenfalls um 15:00 Uhr.

Wichtig: Vorhalten von Aufgabenlösungen zu Abgabeterminen

Zu beiden Abgabeterminen für ein Aufgabenblatt und bis zum Erhalt der jeweiligen Bewertung (gleich ob Erst- oder Zweitabgabe für ein Aufgabenblatt) müssen Sie ihre Lösungen der Programmieraufgaben in einer entsprechend der Aufgabenvorgabe benannten Datei im Homeverzeichnis Ihres Accounts auf dem Übungsrechner `g0` zum automatischen Absammeln abgelegt haben und bereit halten. Ihre Abgabedatei wird aus ihrem Homeverzeichnis zu den genannten Zeitpunkten automatisch kopiert. Findet sich dort keine Abgabedatei, erhalten Sie für diesen Abgabetermin 0 Punkte.

Ihre Abgabedatei darf deshalb **keinesfalls** in einem Unterverzeichnis Ihres Homeverzeichnisses stehen oder anders benannt sein als auf dem Aufgabenblatt vorgegeben.

Arbeit mit Hugs und GHCi; Bewertung ausschließlich unter Hugs

Sie können Ihre Aufgabenlösungen sowohl mit Hugs als auch mit GHCi vorbereiten. Die **Bewertung** Ihrer Aufgabenlösungen erfolgt jedoch **ausschließlich unter Hugs, in der auf dem Übungsrechner `g0` installierten Version.**

Wenn Sie Ihre Aufgabenlösungen nicht direkt auf der `g0` (im Labor "Argentinierstr." oder über eine `ssh`-Verbindung) unter Hugs entwickeln, müssen Sie sich deshalb nach erfolgter Übertragung Ihrer Abgabedatei auf die `g0` (z.B. über eine `scp`-Verbindung) **unbedingt vergewissern**, dass Ihre möglicherweise mit einem anderen Werkzeug entwickelte Lösung **auch unter Hugs auf dem Übungsrechner `g0` wie von Ihnen erwartet arbeitet.**

Machen Sie sich rechtzeitig mit der technischen Abwicklung von Aufgabenabgabe und Überprüfung des erwarteten Programmverhaltens auf der `g0` **vertraut, in jedem Fall rechtzeitig vor dem ersten Abgabetermin.**

Online-Tutorien zu Haskell und Hugs

Unter `www.haskell.org/hugs/pages/hugsman/` finden Sie ein *Online*-Manual für Hugs 98. Lesen Sie in jedem Fall die ersten Abschnitte des Manuals, darunter Abschnitt 3 zum Thema „Hugs for beginners" und probieren Sie die darin beschriebenen Beispiele aus. Machen Sie sich so weit mit dem Haskell-Interpretierer vertraut, dass Sie problemlos einfache Ausdrücke auswerten lassen können.

Ein weiteres *Online*-Tutorial zu Haskell finden Sie auf `haskell.org/tutorial/`. Hinweise zur Arbeit mit Hugs finden Sie auch in Kapitel 4 aus:

- H. Conrad Cunningham. *Notes on Functional Programming with Haskell*. Course Notes, University of Mississippi, 2007 (Chapter 4, Using the Hugs Interpreter). `citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.114.2822&rep=rep1&type=pdf`

Fragen von allgemeinem Interesse zu Vorlesung und Übung können Sie auch über das TISS-Forum zur Lehrveranstaltung zur Diskussion stellen.

Hinweise zu Haskell und möglichen (Anfangs-) Klippen und Hürden

Die Syntax von Haskell birgt im großen und ganzen keine besonderen Fallstricke und ist zumeist intuitiv, wenn auch im Vergleich zu anderen Sprachen anfangs möglicherweise ungewohnt und deshalb "gewöhnungsbedürftig".

Eine Hürde für neu mit Haskell beginnende Programmierer sind Einrückungen. Einrükkungen tragen in Haskell Bedeutung für die Bindungsbereiche und damit für die Programmbedeutung und müssen deshalb unbedingt eingehalten werden (vgl. z.B. Kap. 3.7 der Vorlesung zu „Funktions- und Programmlayout, Abseitsregel"). Alles, was zum selben Bindungsbereich gehört, muss in derselben Spalte beginnen. Diese in ähnlicher Form auch in anderen Sprachen wie etwa `Occam` vorkommende Konvention erlaubt es, Strichpunkte und Klammern einzusparen.

Ein Anwendungsbeispiel in Haskell: Wenn eine Funktion mehrere Zeilen umfasst, muss alles, was nach dem „=" steht, in derselben Spalte beginnen oder noch weiter eingerückt sein als in der ersten Zeile. Anderenfalls liefert Hugs dem Haskell-Programmierbeginner (scheinbar) unverständliche Fehlermeldungen wegen fehlender Strichpunkte.

Weiterhin sollen alle Funktionsdefinitionen und Typdeklarationen in Haskellprogrammen in derselben Spalte (also ganz links) beginnen. Verwenden Sie keine Tabulatoren oder stellen Sie die Tabulatorgröße auf acht Zeichen ein.

Achten Sie auf richtige Klammerung, auch wenn Haskell vielfach keine Klammern verlangt, da sie gemäß geltender Prioritätsregeln automatisch ergänzt werden können. Im Zweifelsfall ist es gute Praxis, ggf. überflüssig zu klammern, um „Überraschungen" zu vermeiden.

Beachten Sie, dass außer „−" (Minus) alle Folgen von Sonderzeichen als Operatoren interpretiert werden; Minus wird kontextabhängig als Infix- oder Prefix-Operator interpretiert.

Achtung: Der Funktionsaufruf „`potenz 2 -1`" entspricht „`potenz 2 - 1`", also „`(potenz 2) - (1)`" und nicht, wie man möglicherweise erwarten könnte, „`potenz 2 (-1)`".

Operatoren haben immer eine niedrigere Priorität als das Hintereinanderschreiben von Ausdrücken (d.h. Funktionsanwendungen). Ein Unterstrich „_" zählt zu den Kleinbuchstaben. Wenn Sie nicht sicher sind, verwenden Sie lieber mehr Klammern als (möglicherweise) nötig.

Funktionsdefinitionen und Typdeklarationen können Sie nicht direkt im Haskell-Interpretierer schreiben, sondern nur aus Dateien laden. Genauere Hinweise zur Syntax finden Sie z.B. unter `haskell.org/tutorial/`.