Programmierparadigmen und Compilerbau
Übungsblatt 3

Sommersemester 2024
Prof. Visvanathan Ramesh
Gamze Akyol
Alperen Kantarci

Abgabe: 12.6.2024

Lesen Sie die folgenden Informationen sorgfältig durch!

- Gruppenarbeit ist **nicht erlaubt!** Sie können mit Ihren Freunden über die Aufgaben sprechen und diskutieren, aber Sie sollten die Aufgaben **individuell** lösen. Plagiate werden nicht toleriert.
- Sie können ChatGPT oder andere Webquellen verwenden, aber Sie dürfen nicht die ganze Frage stellen. Sie sollten in der Lage sein, jeden Schritt Ihres Codes zu erklären.
- Legen Sie alle Dateien, die Sie einreichen möchten, in eine einzige .zip-Datei auf Moodle.
- Laden Sie alle Dateien, die Sie einreichen möchten, in **eine einzelne .zip-Datei auf Moodle** hoch. Bitte stellen Sie sicher, dass Sie nur .pdf- und .hs-Dateien einreichen.
- Bitte schreiben Sie Ihre **Matrikelnummer**, **Name** und **Gruppennummer** zu jeder Datei, die Sie einreichen.
- Der Dateiname sollte dem Muster `IhrName_IhreGruppennummer_Ubungsblatt3.zip` entsprechen.
- Sie sollten ToDo-Teile bearbeiten oder je nach Übung Code für jede Datei hinzufügen und den Rest der Datei so belassen, wie er ist.

Übungsblatt 3

Dieses Übungsblatt basiert auf dem Haskell MOOC Kurs der Universität von Helsinki: `https://github.com/moocfi/haskell-mooc/`

Zwei Dateien `Set5a.hs`, `Set5b.hs` sind in unserer Moodle-Seite enthalten.

- Set5a.hs enthält Übungsabschnitte zur Umsetzung verschiedener mathematischer Funktionen. Sie müssen mindestens 6 der 12 geforderten Aufgaben korrekt lösen, um die volle Punktzahl zu erhalten. Beachten Sie, dass einige Programmieraufgaben von anderen Aufgaben abhängig sein können, und dass es empfehlenswert ist, alle Aufgaben zu lösen.

- Set5b.hs hat Übungsabschnitte zu Listen und funktionaler Programmierung. Sie müssen mindestens 5 der 10 geforderten Aufgaben richtig lösen, um die volle Punktzahl zu erhalten. Beachten Sie, dass einige Programmieraufgaben von anderen Aufgaben abhängig sein können, und es wird empfohlen, alle Aufgaben zu lösen.

Um Ihre Lösungen zu testen, müssen Sie die folgenden Befehle in Ihrer Docker-Umgebung ausführen:

```
stack build
stack runhaskell Set5aTest.hs
stack runhaskell Set5bTest.hs
```

Nachfolgend finden Sie ein Beispiel, um zu sehen, wie Sie die Übungen bearbeiten und testen können:

Beispiel:

```
-- Exercise set 5a
module Set5a where

import Mooc.Todo

--
    ------------------------------------------------------------------------

-- Ex 1: Define the type Vehicle that has four constructors:
     Bike,
-- Bus, Tram and Train.
--
-- The constructors don't need any fields.
-- You will write your code to empty lines below


--
    ------------------------------------------------------------------------

-- Ex 3: Here's the definition for a datatype ShoppingEntry
    that
-- represents an entry in a shopping basket. It has an item
    name (a
-- String), an item price (a Double) and a count (an Int).
    You'll also
-- find two examples of ShoppingEntry values.
--
-- Implement the functions totalPrice and buyOneMore below.

totalPrice :: ShoppingEntry -> Double
totalPrice = todo
```

Sie ändern NUR die todo-Teile oder schreiben Definitionen zu leeren Teilen wie in Set5a.hs Übung 1 und testen dann Ihre Lösung mit Testdateien: `stack runhaskell Set5aTest.hs`, und `stack runhaskell Set5bTest.hs`