Christian-Albrechts-Universität zu Kiel
Institut für Informatik

Prof. Dr. M. Hanus, F. Reck

Institut für Informatik

1. Übung zur Vorlesung „Deklarative Programmiersprachen“

Abgabe am Dienstag, 28. April - 10:15

Hinweis zur Abgabe der Übungsaufgaben

Die Lösungen zu den Übungsaufgaben sollen über die elektronische Übungsverwaltung *iLearn* abgegeben werden, die unter der folgenden URL zu finden ist:

https://www-ps.informatik.uni-kiel.de/iLearn

Die Teilnehmerdaten werden aus der *StudiDB* übernommen. Dabei werden, soweit nötig, neue *iLearn*-Accounts generiert. Ihre Accountdaten werden Ihnen dann per Mail zugesandt. Falls nicht bereits geschehen, melden Sie sich bitte bis spätestens Mittwoch, den 22.04.09 in der *StudiDB* zu dieser Veranstaltung an.

Bitte schließen Sie sich in *iLearn* zu zweit zu einer Kleingruppe zusammen, *bevor* Sie eine Abgabe tätigen.

Hinweise zur Benutzung des Hugs-Systems

Die Programmieraufgaben in dieser Übung sollen mit Hilfe des Hugs-Systems realisiert werden. Dabei es handelt es sich um einen Interpreter für die Programmiersprache Haskell. Es gibt frei verfügbare Versionen von Hugs u.a. für Linux und Windows, so dass Sie die Übungen auch zu Hause bearbeiten können. Informationen dazu sowie Links auf die Dokumentation etc. finden Sie auf der Webseite zur Übung. Die wichtigsten Informationen zum Loslegen werden hier zusammengefasst:

- Starten von Hugs auf den SUNs des Instituts durch `/home/haskell/bin/hugs`. Um für Aufgabe 3 und 4 die Anzahl der Reduktionsschritte und den Speicherbedarf zu ermitteln, starten Sie Hugs mit dem Parameter `+s`.

  Der Eingabeprompt `Prelude>` zeigt an, dass alle vordefinierten Funktionen aus der Prelude bereits geladen wurden.
- `:l test` lädt das Haskell-Program `test.hs` aus dem aktuellen Verzeichnis. Der Eingabeprompt ändert sich daraufhin in `Main>`, um anzuzeigen, dass ein Programm geladen wurde.
- `:r` lädt das zuletzt geladene Program erneut.
- `:t func` gibt den Typ der Funktion `func` aus.
- Um eine Berechnung zu starten, wird einfach ein entsprechender Funktionsaufruf in der Kommandozeile eingegeben, z.B.

  ```
  1+1
  ```

  oder

  ```
  length [1,2,3,4]
  ```

  zum Berechnen der Länge der Liste `[1,2,3,4]`.
- `:q` beendet Hugs.

Aufgabe 1 - Auswertung

3 Punkte

Die Funktion `double` sei wie folgt definiert:

```
double x = x+x
```

Geben Sie alle möglichen Auswertungen von `double (3+2)` an.

Aufgabe 2 - fac

3 Punkte

Geben Sie die in der Vorlesung vorgestellten Funktionsdefinitionen für die Fakultätsberechnung ein und lassen Sie in Hugs die Funktionsaufrufe `fac 0`, `fac 5` und `fac 10` berechnen.

Geben Sie für alle Funktionsdefinitionen das Ergebnis der Berechnung, die Anzahl der Reduktionsschritte[1] (entspricht Zeitbedarf) und den benötigten Speicherbedarf an.

Steigt der Zeitbedarf linear oder quadratisch an?

Woran könnte es liegen, dass die Versionen unterschiedlich gut abschneiden?

Aufgabe 3 - fib

4 Punkte

Definieren Sie die Fibonacci-Funktion über `if-then-else`, bedingte Gleichungen und Pattern-Matching.

Berechnen Sie `fib 5`, `fib 10` und `fib 20` und geben Sie wieder die Zahl der benötigten Reduktionsschritte und den benötigten Speicherbedarf an.

Bestimmen Sie anhand der benötigten Reduktionsschritte die Zeitkomplexität Ihres Algorithmus (`fib` liegt in $O(??)$).

[1] Hinweis: Der Aufruf `fac 0` beispielsweise benötigt eigentlich nur einen Reduktionsschritt. Die sehr viel höhere Zahl, die Hugs Ihnen ausgibt, wird z.B. durch die Verwaltung interner Strukturen verursacht.