EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Fachbereich Informatik

Programmiersprachen und Softwaretechnik

Prof. Dr. Klaus Ostermann

Leitung des Übungsbetriebs
Philipp Schuster
philipp.schuster@uni-tuebingen.de

Programmiersprachen II

Hausaufgabe 3 – WS 16

Tübingen, 2. November 2016

Abgabe Geben Sie diese Hausaufgabe bis Donnerstag den 10. November 2016 ab. Entweder bis 12:00 per Email an Philipp Schuster (philipp.schuster@uni-tuebingen.de) oder zu Beginn der Übung auf Papier.

Gruppen Sie können in Gruppen von bis zu 2 Personen arbeiten. Schreiben Sie in jedem Fall die Namen und Matrikelnummern aller Gruppenmitglieder mit auf die Hausaufgabe / in die Email. Wenn Sie in einer Gruppe arbeiten, achten Sie darauf, dass alle Mitglieder der Gruppe den Stoff verstehen. Nur dann sind die Hausaufgaben eine gute Vorbereitung auf die Prüfung.

Punkte Sie können für die Aufgaben dieser Woche jeweils zwischen 0 und 2 Punkten bekommen. Insgesamt also zwischen 0 und 6 Punkten. Sie bekommen für die Aufgaben jeweils:
1 Punkt, wenn Ihre Abgabe zeigt, daß Sie sich mit der Aufgabe ernsthaft beschäftigt haben.
2 Punkte, wenn Sie die Aufgabe weitgehend korrekt gelöst haben.
Um zur Klausur zugelassen zu werden müssen Sie mindestens 50% der maximal möglichen Punkte in den Hausaufgaben erreichen. Mit 60% bis 100% der möglichen Hausaufgabenpunkte erhalten Sie einen Bonus von 0% bis 20% der Klausurpunkte in der Klausur.

Aufgabe 1: Evaluation

In der Vorlesung wurde die Reduktionsrelation für das Lambda-Kalkül vorgestellt. Reduzieren Sie die folgenden Terme nach den Ableitungsregeln der Reduktionsrelation, bis sie in Normalform sind:

1. $(\lambda a.\ \lambda b.\ a)\ (\lambda x.\ x)$
2. $(\lambda a.\ \lambda b.\ a\ b)\ (\lambda x.\ \lambda y.\ x\ y)\ (\lambda z.\ z)$
3. $(\lambda z.\ z\ z)\ (\lambda f.\ f\ (\lambda a.\ a))$

Ableitungsbäume müssen Sie nicht angeben. Geben Sie alle Reduktionsschritte an. Zum Beispiel würden Sie für den Term $(\lambda f.\ f\ (\lambda x.\ x))\ (\lambda x.\ x)$ folgende Reduktionsschritte angeben:

$$(\lambda f.\ f\ (\lambda x.\ x))\ (\lambda x.\ x) \longrightarrow (\lambda x.\ x)\ (\lambda x.\ x) \longrightarrow (\lambda x.\ x)$$

Aufgabe 2: Church-Kodierung

In der Vorlesung haben wir natürliche Zahlen als Folds im Lambda-Kalkül kodiert. In dieser Aufgabe kodieren wir Listen als Folds. Die grundlegenden Listenfunktionen sehen dann folgendermaßen aus:

$\mathrm{nil} = \lambda f.\ \lambda z.\ z$
$\mathrm{singleton} = \lambda x.\ \lambda f.\ \lambda z.\ f\ x\ z$
$\mathrm{cons} = \lambda h.\ \lambda t.\ \lambda f.\ \lambda z.\ f\ h\ (t\ f\ z)$
$\mathrm{fold} = \lambda f.\ \lambda z.\ \lambda l.\ l\ f\ z$

Unter dieser Kodierung ist die Liste mit den Elementen c_1, c_2, c_3:

$\mathrm{onetwothree} = \lambda f.\ \lambda z.\ f\ \ \mathrm{c}_1\ \ (f\ \ \mathrm{c}_2\ \ (f\ \ \mathrm{c}_3\ \ z))$

Schreiben Sie folgende typische Funktionen auf Listen für diese Repräsentation:

1. sum
2. map
3. length

Sie dürfen die definierten Makros aus der Vorlesung verwenden, insbesondere plus, c_0, c_1, c_2, ...

Aufgabe 3: Monotonie

Die Reduktionsrelation $\longrightarrow$ und die Funktion size sei für Terme des ungetypten Lambda-Kalküls definiert wie in der Vorlesung. Beweisen oder widerlegen Sie folgende Aussage: wenn $t \longrightarrow t'$ ableitbar ist, dann gilt $\mathrm{size}(t) > \mathrm{size}(t')$.