PHILIPPS-UNIVERSITÄT MARBURG
Fachbereich Mathematik und Informatik
Prof. Dr. R. Loogen
M. Dieterle

D-35032 Marburg
Hans Meerwein Straße

17.Oktober 2007

1. Übung zu „Grundlagen des Compilerbaus“, WS 2007/08

Abgabe der Aufgaben:	**Mi, 24.Oktober 2007 (vor der Vorlesung)**
Besprechung der Aufg.:	**Fr, 26.Oktober 2007 in der Übung**

Hinweise:

- Die Lösungen aller Aufgaben sollen grundsätzlich schriftlich abgegeben werden, sofern die Aufgaben nicht ausdrücklich als mündliche Aufgaben gekennzeichnet sind. Programme sollen ausgedruckt und *zusätzlich* per E-Mail an den jeweiligen Tutor geschickt werden. Die Abgabe ist in Gruppen bis zu zwei Personen erlaubt.
- Mündliche Aufgaben sollen (wenn vorhanden) zum angegebenen Tutoriumstermin vorbereitet werden, sie werden im Tutorium präsentiert und besprochen.
- Die Präsentation von mündlichen und schriftlichen Aufgaben im Tutorium ist Bedingung für den Erwerb eines Leistungsnachweises. Bitte achten Sie selbst (mit) darauf, dass Sie diese Bedingung erfüllen können.
- Ein kompakter Überblick über Haskell-Funktionen und Datentypen ist z.B. unter http://zvon.org/other/haskell/Outputglobal/index.html zu finden.

Aufgaben

1.1 **T-Diagramme** — 4 Punkte

Sie haben die Aufgabe, einen Java-Übersetzer zu entwickeln, der auf einer SGI-Workstation läuft und Code für die Sony Playstation 3 erzeugt. Ihr Auftraggeber verlangt, dass neu zu implementierende Teile in Java geschrieben sind.
Ihnen stehen zwei auf SGI lauffähige C-Übersetzer zur Verfügung, welche Maschinencode für SGI bzw. für die Playstation erzeugen. Geben Sie in Form von T-Diagrammen an, wie Sie (mit möglichst wenig Aufwand) vorgehen können und an welchen Stellen “Handarbeit” erforderlich ist.

1.2 **Entfernung von Whitespace** — 3 Punkte

Schreiben Sie ein Haskell-Programm, das aus einer gegebenen Zeichenkette Reihen von Leerzeichen, Tabulatoren (`\t`) und Zeilenumbrüchen (`\n`) entfernt und durch jeweils ein Leerzeichen ersetzt.

(a) Schreiben Sie zunächst eine Funktion `isWhiteSpace :: Char -> Bool` die erkennt, ob der Argumentbuchstabe zu den oben genannten Zeichen gehört.

Bitte wenden!

(b) Schreiben Sie nun eine Funktion `compress :: [Char] -> [Char]` die unter Benutzung von `isWhiteSpace` Ketten von mehr als einem Leerzeichen zu einem einzigen Leerzeichen reduziert.

Beispiel: `"Emil und   die Detektive\n\n \t\t von Erich Kaestner    "`

wird konvertiert zu `"Emil und die Detektive von Erich Kaestner "`

1.3 Scannen eines Strings

5 Punkte

(a) Definieren Sie eine Haskell-Funktion `getStrings :: String -> [String]`, die einen `String` erhält und alle zusammenhängenden Zeichenketten extrahiert, welche nur aus Kleinbuchstaben, Großbuchstaben, Zahlen, Leerzeichen und dem '_' - Symbol bestehen. Sie können dazu die Funktionen `isLower, isUpper, isDigit, isSpace :: Char -> Bool` benutzen.

(b) Schreiben Sie ein Haskell-Programm, das die Funktion `getStrings` auf die kompilierte Version des eigenen Codes anwendet. Die von `getStrings` erstellte Wortliste soll ausgegeben werden. Kompilieren Sie das Programm zu Testzwecken mit "ghc". Nützliche Funktionen sind:

```
getProgName :: IO String
readFile    :: String -> IO String
putStrLn    :: String -> IO ()
mapM        :: Monad a => (b -> a c) -> [b] -> a [c]
```