13. Übung	**Informatik A**	**WS 07/08**
Klaus Kriegel		**Abgabe: 05.02.2008, 12:00 Uhr**

Das ist der letzte abgabepflichtige Übungszettel, mit dem man seinen Punktestand noch einmal verbessern kann. Korrigiert werden die Aufgaben 2 und 3, aber auf das Soll werden nur 8 Punkte angerechnet.

Aufgabe 1: **Polymorphe Typbestimmung:** (0 Punkte)

Führen Sie für die folgenden Definitionen eine Typprüfung durch. Natürlich kann man das auch von hugs erledigen lassen, aber Sie sollten üben, es selbst zu machen. Stellen Sie fest, ob die Definitionen konsistent sind, und geben Sie bei positivem Ergebnis die Signatur an:

```
f1 x
   | x > 0        = True
   | otherwise    = 12

f2 n
   | ord n >= 6   = [n]
   | otherwise    = "n"++[n]

f3 x y = show (div x 3) ++ y

f4 (x,y,z) = (x,['1'..y],x++[z])

f5 x y z = [(x, mod x y)]++z
```

Aufgabe 2: **Lazy Evaluation** ($3 + 1 + 2$ Punkte)

a) Welche Liste von Zahlen ergibt der Aufruf
`[ 10*x + y | x <- [0..9], y <- [0..9], z <- [1,4..16], x + y == z]` ?
Geben Sie eine einfache Beschreibung dieser Liste und begründen Sie Ihre Antwort.

b) Wieviele Aufrufe des Operators == aus der letzten Bedingung erfolgen bei der Berechnung der gesamten Liste?

c) Wieviele Aufrufe des Operators == erfolgen bei der Berechnung von
`head [ 10*x + y | x <- [0..9], y <- [0..9], z <- [1,4..16], x + y == z]`
und bei der Berechnung von
`head [ 10*x + y | x <- [1..9], y <- [1..9], z <- [1,4..16], x + y == z]` ?

Aufgabe 3: **Unendliche Listen** ($4 + 4$ Punkte)

Eine Zahl, die sich als Produkt aus den Faktoren $1, 2, 3$ und 5 darstellen lässt, wobei jeder Faktor beliebig oft auftreten kann, nennt man eine Hamming-Zahl. Hier ist ein

Anfangsstück der geordnete Liste der Hamming-Zahlen:
$1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, 16, 18, 20, \ldots$.

a) Erzeugen Sie die (unendliche) geordnete Liste der Hamming-Zahlen mit List-Comprehension und schätzen Sie die Laufzeit ab, die Ihr Algorithmus benötigt, um alle Hamming-Zahlen $\leq n$ zu finden. Es reicht aus, wenn Sie für $T(n)$ die Anzahl der Vergleichsoperationen `==, /=, <=, >=, <, >` zählen.

b) Wenn Ihre Analyse eine quadratische oder noch größere Laufzeit ergibt, sollte der Algorithmus (mit geeigneten Hilfsfunktionen) so zu modifiziert werden, dass er mit $O(n \cdot (\log_2 n)^k)$ Vergleichsoperationen auskommt, wobei für $k \in \mathbb{N}$ ein möglichst kleiner Wert erreicht werden soll.