5. Übung **Informatik A** **WS 07/08**

Klaus Kriegel **Abgabe: 27.11.2007, 12:00 Uhr**

Allgemeine Anforderungen bei Programmieraufgaben:

Die Haskell–Dateien mit dem Code müssen per email an den Tutor geschickt werden. Zusätzlich sollte der Code gedruckt abgegeben werden. Alle Lösungen sollten die folgenden Forderungen erfüllen:

- Für alle neu definierten Funktionen ist die Signatur anzugeben.
- Die Programme müssen nicht nur lauffähig sein, sie sollten auch mit verschiedenen Eingabebeispielen getestet sein.
- Die Lösungen sind kurz zu begründen. Das kann durch Kommentare im Code oder durch einen Begleittext erfolgen.

Aufgabe 0: **Hugs** (0 Punkte)

Machen Sie sich mit Hugs vertraut! Implementieren Sie einige der Beispiele, die in Vorlesung und Übung besprochen wurden, bauen Sie absichtliche Fehler ein (Funktionsname mit Großbuchstaben am Anfang, Wächtersymbol nicht eingerückt, Parameterliste widersprüchlich zur Signatur, ...) und sehen Sie sich die Fehlermeldungen an.

Aufgabe 1: **Fallunterscheidungen** (8 Punkte)

Implementieren Sie die folgenden Funktionen in Haskell. Dazu sind einfache, zum Teil auch etwas komplexere Fallunterscheidungen notwendig.
a) Die Funktion `avgIncluded :: Int -> Int -> Int -> Bool` erhält drei beliebige `Int`–Werte als Eingabe und soll `True` ausgeben, wenn einer der Werte der Durchschnitt dieser drei Werte ist, sonst `False`.
b) Die Funktion `span` erhält drei beliebige `Float`–Werte als Eingabe und soll die Differenz aus dem größten und kleinsten Wert ausgeben.
c) Die Funktion `median` erhält drei beliebige `Float`–Werte als Eingabe und soll den zweitgrößten Wert aus dieser Menge ausgeben.
d) Die Funktion `antivalOf4` erhält vier beliebige `Bool`–Werte und soll die Antivalenz aus diesen Werten bestimmen.

Aufgabe 2: **Polynome** (2 + 2 + 4 Punkte)

Ein reelles Polynom vom Grad n ist ein Ausdruck der Form
$a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0$
wobei alle Koeffizienten a_i reelle Zahlen sind und $a_n \neq 0$ ist. Man kann ein Polynom an einer Stelle $r \in \mathbb{R}$ auswerten, indem die Variable x durch die Zahl r ersetzt und der Wert des so entstandenen arithmetischen Ausdrucks ausgerechnet wird.

a) Implementieren Sie eine Haskell-Funktion
`eval3 :: Float -> Float -> Float -> Float -> Float -> Float`
zur Auswertung von Polynomen vom Grad 3, wobei die erste Eingabestelle dem Wert r und die folgenden Stellen die Koeffizienten in der Reihenfolge a_3, a_2, a_1, a_0 angeben.

b) Ähnlich wie beim Schulverfahren zur Division von ganzen Zahlen kann man auch eine Polynomdivision mit Rest ausführen (Tutorien): Das Ergebnis der Divsion von $p(x)$ durch $s(x)$ ist ein Quotientenpolynom $q(x)$ und ein Restpolynom $r(x)$ dessen Grad kleiner als der Grad von $s(x)$ ist und so, dass $p(x) = q(x) \cdot s(x) + r(x)$ gilt.
Folglich ist bei der Division durch ein Polynom der Form $s(x) = (x+b)$ ist der Rest immer ein Polynom vom Grad 0, also eine Zahl. Üben Sie das Verfahren an den Beispielen $(2x^2 + 2x - 10) : (x - 3)$ und $(3x^3 + x^2 - 5x + 2) : (x + 2)$.

c) Implementieren Sie zwei Haskell-Funktionen
`polyDivRest2 :: Float -> Float -> Float -> Float -> Float` und
`polyDivRest3 :: Float -> Float -> Float -> Float -> Float -> Float`
deren Eingabe an erster Stelle den Parameter b und danach die Koeffizientenfolge eines Polnoms $p(x)$ vom Grad 2 bzw. 3 enthält und die den Rest der Polynomdivision von $p(x)$ durch $(x+b)$ berechnet. Zu diesem Aufgabenteil gehören nicht nur die funktionierenden Programme, sondern auch eine Begründung, warum sie das Geforderte leisten.