



Funktionale Programmierung, WS 2009/10, Blatt 1

Dr. Jan Schwinghammer, Prof. Dr. Gert Smolka

www.ps.uni-sb.de/courses/proseminar-ws09/

Grundlagen funktionaler Programmierung, Auswertungsstrategien, Streams

Aufgabe 1.1 (Rückblick) Erinnern Sie sich an die zentralen Ideen zur funktionalen Programmierung aus der Vorlesung Programmierung 1:

- Polymorphismus,
- höherstufige Prozeduren,
- Muster und Konstruktoren,
- Listen, Bäume, Faltungen,
- (abstrakte) Datenstrukturen.

(Dies entspricht in etwa den Kapiteln 3, 4, 6, 7 und 14 aus *Programmierung – Eine Einführung in die Informatik mit Standard ML* von Gert Smolka.)

Aufgabe 1.2 (Programmieren in Haskell) Lesen Sie Kapitel 1–3 in *A Gentle Introduction to Haskell* (www.haskell.org/tutorial/) von Hudak, Peterson und Fasel, insbesondere Abschnitte 3.3 und 3.4 zu nicht-strikter ('lazy' oder 'by need') Auswertung. Verwenden Sie einen Haskell Interpreter, um mit einigen Beispielen aus dem Text zu experimentieren.

Aufgabe 1.3 (Streams) Wir bezeichnen eine unendliche Liste als *stream*. Abschnitt 3.4 in *A Gentle Introduction...* zeigt, wie streams als zyklische Listen definiert werden können. Geben Sie ähnliche Definitionen

- a) für $twos = [2, 2, 2, \dots]$,
- b) für $powers = [1, 2, 4, 8, 16, \dots]$, bestehend aus den Potenzen 2^n ,
- c) für $fact = [1, 1, 2, 6, 24, \dots]$, bestehend aus den n -ten Fakultäten $n!$.

Aufgabe 1.4 (Sieve of Eratosthenes) Ein Verfahren, mit dem Primzahlen bestimmt werden können, ist das folgende: Ausgehend von $[2, 3, 4, \dots]$ werden zunächst die Vielfachen $4, 6, 8, \dots$ von 2 gestrichen. Anschliessend werden die Vielfachen $6, 9, 12, \dots$ von 3 (der nächst größeren Zahl der verbleibenden Elemente) gestrichen, sofern sie noch vorkommen, dann die Vielfachen von 5... Wird dieser Prozess fortgeführt, so ergibt sich als 'Grenzwert' der stream $[2, 3, 5, 7, \dots]$ aller Primzahlen.

Realisieren Sie dieses Verfahren in Haskell mittels zweier Deklarationen $primes :: [Int]$ und $sieve :: [Int] \rightarrow [Int]$.

Aufgabe 1.5 (Hamming sequence) Deklarieren Sie in Haskell den stream $hamming = [1, 2, 3, 4, 6, 8, 9, 12, \dots]$ aller Vielfachen $2^n 3^m$ von 2 und 3, aufsteigend angeordnet.

Hinweis 1 Nutzen Sie aus, dass $1 = 2^0 3^0$ ein Element des streams ist, und dass k enthalten ist genau dann wenn $2k$ und $3k$ ebenfalls enthalten sind.

Hinweis 2 Verwenden Sie eine Hilfsprozedur $merge :: [Int] \rightarrow [Int] \rightarrow [Int]$, zum Mischen zweier sortierter streams.

Aufgabe 1.6 (Auswertungsstrategien in Standard ML) Sei eine Prozedurdeklaration $fun\ p\ x = e$ gegeben. Erinnern Sie sich daran, dass die Auswertungsreihenfolge in Standard ML im Gegensatz zu Haskell 'strikt' ist: Bei einer Prozeduranwendung pa wird das Argument a zu einem Wert v ausgewertet, bevor der Prozedurrumpf e ausgeführt wird (mit dem formalen Parameter x der Prozedur ersetzt durch v).

- a) Überlegen Sie sich an dem Beispiel $fun\ const1\ x = 1$, dass bedarfsgesteuerte und strikte Auswertung zu unterschiedlichem Verhalten führen können.
- b) Lässt sich bedarfsgesteuerte Auswertung in Standard ML simulieren?