

Programmierung 1 (Wintersemester 2012/13)

Zusatzübungsblatt 4

(Kapitel 3 und 4)

Aufgabe 4.1 (*Offizielle Übungsaufgabe aus dem Wintersemester 11/12*)

Deklariieren Sie eine Identitätsprozedur $"a \text{ ideq} : "a \rightarrow "a$, deren Typschema auf Typen mit Gleichheit eingeschränkt ist. Verzichten Sie dabei auf explizite Typangaben.

Aufgabe 4.2 (*Offizielle Übungsaufgabe aus dem Wintersemester 11/12*)

Geben Sie zunächst die geschachtelte Paardarstellung von folgenden Listen an. Überlegen Sie sich dann, wie Sie die Liste nur unter Verwendung von nil und $::$ aufschreiben können.

- (a) $[1, 2, 3, 4]$
- (b) $[1.0, 2.0, 3.0]$
- (c) $[(1, 4), (4, 1), (6, 2), (3, 8)]$
- (d) $[[4, 2], [3, 2, 1], []]$

Aufgabe 4.3 (*Offizielle Übungsaufgabe aus dem Wintersemester 11/12*)

Schreiben Sie folgende Prozeduren:

- (a) Eine Prozedur $listEvenOdd : int \text{ list} \rightarrow (int * bool) \text{ list}$, welche zu einer Liste von Zahlen folgendes zurückgibt:
 $listEvenOdd[x_1, x_2, \dots, x_n] = [(x_1, b_1), (x_2, b_2), \dots, (x_n, b_n)]$, wobei b_i jeweils $true$ sein soll, wenn x_i gerade ist, ansonsten $false$ (für alle $i \in \{1, \dots, n\}$).
- (b) Eine Prozedur $countEvenOdd : int \text{ list} \rightarrow int * int$, welche zu einer Liste von Zahlen das Tupel (e, o) zurückgibt, sodass e die Anzahl gerader Zahlen in der Liste und o die Anzahl ungerader Zahlen in der Liste angibt.

Aufgabe 4.4 (*Offizielle Übungsaufgabe aus dem Wintersemester 11/12*)

Das Sieb des Eratosthenes ist ein Algorithmus zur Berechnung von Primzahlen bis einschließlich einem gegebenen Index n . Dazu wird eine Liste oder Tabelle von Zahlen von 2 bis n verwendet, aus der nach und nach mehr Zahlen rausgestrichen werden. Solange es noch Elemente in der Liste gibt, betrachtet man jeweils die kleinste Zahl, die in der Liste ist, nimmt sie aus der Liste raus und markiert sie als Primzahl. Dann streicht man alle Vielfachen von ihr aus der Liste. Die Menge der markierten Zahlen ist dann gerade die Menge aller Primzahlen zwischen 2 und n . Im Folgenden soll dieses Prinzip in SML umgesetzt werden.

- (a) Schreiben Sie eine Prozedur $listevonzahlen : int \rightarrow int \rightarrow int \text{ list} \rightarrow int \text{ list}$, die zu von einem Startindex m an alle Zahlen einschließlich n aufsteigend geordnet in eine Liste schreibt.
- (b) Schreiben Sie eine Prozedur $loesche : int \rightarrow int \text{ list} \rightarrow int \text{ list}$, die aus einer gegebenen Liste alle Zahlen entfernt, die durch eine gegebene Zahl n teilbar sind.
- (c) Schreiben Sie nun eine Prozedur $sieb' : int \text{ list} \rightarrow int \text{ list} \rightarrow int \text{ list}$, die aus der einer Liste xs das erste Element x entfernt, es an die andere Liste ys anhängt und alle Zahlen aus xs entfernt, die durch x teilbar sind.
- (d) Fügen Sie nun die drei Hilfsprozeduren zum Sieb des Eratosthenes zusammen.

Aufgabe 4.5 (Offizielle Übungsaufgabe aus dem Wintersemester 11/12)

In dieser Aufgabe sollen Sie Ihren bisherigen Punkteschnitt aus den Tests berechnen lassen. Schreiben Sie dazu eine Prozedur $average : real\ list \rightarrow real$, die mithilfe von *foldl* das arithmetische Mittel aller Zahlen einer Liste berechnet wie folgt:

- (a) Unter Verwendung einer Hilfsprozedur $lengthReal : \alpha\ list \rightarrow real$, die die Länge einer Liste als *real* ausgibt
- (b) Ohne Verwendung von *lengthReal* mithilfe von Faltung.

Aufgabe 4.6

Sei folgende Prozedur bereits deklariert:

```
fun f ((2, 4.0)::xr) = 42
  | f [(1, y), (_, 3.0), (_, 4.0)] = 5
  | f [(x, _), (u, w)] = u + x
  | f [(44, 11.0), (12, 3.0)] = 42
  | f (x::xr) = #1(hd xr)
```

Was ergeben die folgenden Prozeduranwendungen?

- (a) $f [(1, 5.0), (4711, 3.0), (0815, 4.0)]$
- (b) $f [(2, 4.0), (4, 2.0)]$
- (c) $f [(2, 4.0)]$
- (d) $f [(44, 11.0), (12, 3.0)]$
- (e) $f\ nil$
- (f) $f [(2, 4.1)]$
- (g) $f [(2, 4.000000000000000001)]$

Aufgabe 4.7 (Offizielle Übungsaufgabe aus dem Wintersemester 11/12)

Sie wurden bei VIVA eingestellt, um einen neuen Algorithmus für den *Love-Generator* zu implementieren. Dieser berechnet die Kompatibilität zwischen zwei Menschen anhand ihrer Namen und gibt die Beziehungsaussichten in Prozent an. Ihre Aufgabe ist es nun, folgenden, wissenschaftlich höchst fundierten Algorithmus zu implementieren: Schreiben Sie eine Prozedur $loveGen : string * string \rightarrow int$, die für jeden der Strings die Ordinalzahlen aufaddiert, daraus die Summe bildet und davon den Rest der Division mit 100 zurückgibt.

- (a) Schreiben Sie zunächst eine Hilfsprozedur $characterStrength : string \rightarrow int$, die die Summe der Ordinalzahlen aller Buchstaben eines Strings zusammenzählt. Nutzen Sie dafür *foldl*.
- (b) Nutzen Sie die Hilfsprozedur *characterStrength*, um *loveGen* wie beschrieben zu implementieren.
- (c) Um den Lovegenerator auch international etablieren zu können, benötigt VIVA eine angepasste Version, um die Kompatibilität eines ganzen Harems zu berechnen. Schreiben Sie nun eine Prozedur $loveGenHarem : string\ list \rightarrow int$, die die Kompatibilität eines ganzen Harems berechnet. Nutzen Sie dafür erneut *characterStrength* und *foldl*, jedoch keine sonstigen Hilfsprozeduren.

Aufgabe 4.8 (Schriftliche Aufgabe aus dem Wintersemester 11/12)

Welches Thema der Vorlesung hat Sie bisher am meisten fasziniert oder überrascht? Versuchen Sie den Leser mit Ihrer Faszination für das Thema anzustecken, indem Sie erklären, was Ihre Faszination oder Überraschung ausgelöst hat.

Aufgabe 4.9 (Schriftliche Aufgabe aus dem Wintersemester 11/12)

Erklären Sie die folgenden Begriffe ausführlich in eigenen Worten und setzen Sie sie zueinander in Beziehung.

Polymorphe Typisierung: Typschemen, freie und quantifizierte Typvariablen, Instanziierung und Instanzen; polymorphe, monomorphe und ambige Deklarationen; Spezifikation polymorpher Prozeduren.