

Programmierung 1 (Wintersemester 2012/13)

Lösungsblatt 1

(Kapitel 3)

Hinweis: Dieses Zusatzübungsblatt wird von den Tutoren der Vorlesung “Programmierung 1” (WS 2012/13) erstellt. Es enthält unter anderem Aufgaben von den Übungsblättern der gleichen Vorlesung aus dem Wintersemester 2011/12 gehalten von Prof. Hermanns, sowie des Nachklausurtutoriums aus dem gleichen Jahr. Außerdem enthält es einige neue, von den Tutoren erstellte Aufgaben.

Die hier gestellten Aufgaben und die damit abgedeckten Themenbereiche sind weder für die Klausur relevant, noch irrelevant. Sie dienen vor allem dazu Ihnen andere Blickrichtungen auf den Stoff zu präsentieren.

Aufgabe 1.1 (Offizielle Übung aus dem Wintersemester 11/12)

Schreiben Sie eine Prozedur

$itercond : int \rightarrow \alpha \rightarrow (\alpha \rightarrow bool) \rightarrow (\alpha \rightarrow \alpha) \rightarrow \alpha$,

$$\text{wobei } itercond\ n\ s\ d\ t\ f = \begin{cases} s & n = 0 \\ itercond\ (n-1)\ (t(s))\ d\ t\ f & d(s) = true \wedge n \neq 0 \\ itercond\ (n-1)\ (f(s))\ d\ t\ f & d(s) = false \wedge n \neq 0 \end{cases}$$

Lösung 1.1:

```
fun itercond n s d t f = if n = 1 then s else (if d s then itercond (n-1) (t s) d t f else itercond
(n-1) (f s) d t f)
```

Aufgabe 1.2 (Offizielle Übung aus dem Wintersemester 11/12)

Geben Sie geschlossene Abstraktionen an, die die folgenden Typen haben. Die Abstraktionen sollen nur mit Prozeduranwendungen, Tupeln und Bezeichnern gebildet werden. Konstanten und Operatoren sollen nicht verwendet werden.

- a) $(unit * int) \rightarrow unit \rightarrow int$
- b) $int \rightarrow (int \rightarrow real * real) \rightarrow unit * unit \rightarrow real$
- c) $(int * real * unit) \rightarrow real \rightarrow int * real$
- d) $int \rightarrow unit \rightarrow real \rightarrow int * int$

Lösung 1.2:

- a) `fn(a:unit,b:int) => (fn (c:unit) => b)`
- b) `fn (x:int) => (fn (f : int -> real * real) => (fn (a:unit, b:unit) => #1(f x)))`
- c) `fn (a:int, b:real, c:unit) => (fn (d:real) => (a,b))`
- d) `fn(a:int) => (fn(b:unit) => (fn (c:real) => (a,a)))`

Aufgabe 1.3

Deklariieren Sie polymorphe Prozeduren, die die folgenden Typschemata besitzen:

- (a) $\forall \alpha \beta. (\alpha * \alpha \rightarrow \beta) \rightarrow \alpha \rightarrow \beta$
- (b) $\forall \alpha \beta \gamma. \alpha * \beta * \gamma \rightarrow \beta$
- (c) $\forall \alpha \beta \gamma. \alpha * \beta * \gamma \rightarrow \alpha * \gamma$
- (d) $\forall \alpha \beta \gamma. (\alpha \rightarrow \beta \rightarrow \gamma) \rightarrow \alpha \rightarrow \beta \rightarrow \gamma$

Lösung 1.3:

- (a) `fun prozedur f n = f (n, n)`
- (b) `fun prozedur (a, b, c) = b`
- (c) `fun prozedur (a, b, c) = (a, c)`
- (d) `fun prozedur f a b = f a b`

Aufgabe 1.4 (Nachklausurtutorium WS 12/13)

Schreiben Sie eine Prozedur, die die zweite oder dritte Zahl, die ein Prädikat erfüllt findet, unter Verwendung von *first*.

Können Sie auch eine Prozedur schreiben, die ganz allgemein die *k*-te Zahl findet, die ein Prädikat erfüllt?

Lösung 1.4:

```
fun find k p = iter k 0 (fn x=> first (x+1) p)
```

Aufgabe 1.5 (Schriftliche Aufgabe aus dem Wintersemester 11/12)

Sie wissen nun, wie Ihr Interpreter für SML arbeitet. Kann man auch einen Menschen als Interpreter für menschliche Sprache betrachten? Welche der Verarbeitungsphasen eines Interpreters finden auch beim Menschen statt und inwieweit unterscheiden sie sich.

Aufgabe 1.6 (Schriftliche Aufgabe aus dem Wintersemester 11/12)

Alle Phrasen der Sprache SML lassen sich auch in Baumdarstellung angeben. Begründen Sie ob man dies auch für natürliche Sprachen wie das Deutsche tun kann. Macht dies Sinn?

Aufgabe 1.7 (Schriftliche Aufgabe aus dem Wintersemester 11/12)

Erklären Sie das Konzept der Höherstufigkeit von Prozeduren in eigenen Worten. Wenn möglich, überlegen Sie sich einen anschaulichen Vergleich, wo Höherstufigkeit als grundlegendes Konzept im Alltag (eines Informatikers) vorkommt. Gehen Sie gezielt auf den genauen Zusammenhang der Begriffe *höherstufige Prozedur* und *kaskadierte Prozedur* ein.