

Übungen zu Informatik I (Lösungsvorschlag)

Aufgabe 7-1 Multiplikation in curried Form (keine Abgabe)

a) Eine mögliche Herleitung ist die folgende:

- | | | |
|-----|---|-------------------------------------|
| (1) | $\{x : \text{int}\} \triangleright x : \text{int}$ | Typaxiom |
| (2) | $\{y : \text{int}\} \triangleright y : \text{int}$ | Typaxiom |
| (3) | $\emptyset \triangleright * : \text{int} * \text{int} \rightarrow \text{int}$ | Typaxiom |
| (4) | $\{x : \text{int}, y : \text{int}\} \triangleright x * y : \text{int}$ | (R2) mit Prämissen (1), (2) und (3) |
| (5) | $\{x : \text{int}\} \triangleright \text{fn } y \Rightarrow x * y : \text{int} \rightarrow \text{int}$ | (R4) mit Prämisse (4) |
| (6) | $\emptyset \triangleright \text{fn } x \Rightarrow \text{fn } y \Rightarrow x * y : \text{int} \rightarrow \text{int} \rightarrow \text{int}$ | (R4) mit Prämisse (5) |

b) Um die gewünschte Herleitung zu erhalten, verlängern wir die Herleitung aus Teilaufgabe (a):

- | | | |
|-----|--|--------------------------------|
| (7) | $\emptyset \triangleright 5 : \text{int}$ | Typaxiom |
| (8) | $\emptyset \triangleright (\text{fn } x \Rightarrow \text{fn } y \Rightarrow x * y) 5 : \text{int} \rightarrow \text{int}$ | (R5) mit Prämissen (6) und (7) |

Aufgabe 7-2 Standardausnahmen (keine Abgabe)

a) (* Nochmals die Fakultätsfunktion aus der Vorlesung *)

```
fun fak n = if n=0 then 1 else n*fak(n-1)
```

```
(* Die Funktion safe_fak *)
```

```
fun safe_fak n = fak n handle overflow => ~1;
```

b)

```
fun fast_exp (n, m) =  
  if m = 0  
  then 1  
  else if m mod 2 = 0  
  then let val k = fast_exp(n, m div 2)  
        in k * k  
        end  
  else n * fast_exp(n, m-1);;
```

```
fun safe_exp(n, m) =  
  fast_exp(n, m)  
  handle overflow => ~1;;
```

Die im Folgenden definierte Funktion *bad_exp* führt nur dann eine Fehlerbehandlung durch, wenn der Überlauf bei der Berechnung von $n * \text{bad_exp}(n, m - 1)$ auftritt. Außerdem rechnet die Funktion nach dem Auftreten einer Ausnahme mit dem Wert -1 weiter: z.B. $\text{bad_exp}(7, 11) = -1$, $\text{bad_exp}(7, 22) = 1$, $\text{bad_exp}(7, 23) = 7$.

```

fun bad_exp(n, m) = if m = 0
  then 1
  else if m mod 2 = 0
    then let val k = bad_exp(n, m div 2)
         in k * k
        end
    else n * bad_exp(n, m-1)
  handle overflow => ~1;;

```

Aufgabe 7-3

Mustervergleich

(keine Abgabe)

a) Die gesuchte Funktion *qzt_pm* ist wie folgt mit Mustervergleich definierbar:

```

fun qzt_pm (n, 0) = n = 0
  | qzt_pm (n, k) = n = k*k orelse qzt_pm (n, k-1)

```

b) Die Funktion *teilersumme_pm* ist wie folgt definiert:

```

fun teilersumme_pm(n, 0) = 0
  | teilersumme_pm(n, k) = (if n mod k = 0 then k else 0) + teilersumme_pm(n, k-1);

```

c) Die gewünschte Funktion *hatteiler_pm* kann man wie folgt mit Mustervergleich definieren:

```

fun hatteiler_pm (n, 1) = false
  | hatteiler_pm (n, k) = n mod k = 0 orelse hatteiler_pm (n, k-1)

```

Aufgabe 7-4

Typisierung mit lokalen Konstanten

(keine Abgabe)

Wir verwenden neben der Typisierungsregel (R6) auch die Regel (R5) und erhalten:

- | | | |
|------|---|-------------------------------------|
| (1) | $\{x : \mathbf{int}\} \triangleright x : \mathbf{int}$ | Typaxiom |
| (2) | $\emptyset \triangleright 1 : \mathbf{int}$ | Typaxiom |
| (3) | $\emptyset \triangleright + : \mathbf{int} * \mathbf{int} \rightarrow \mathbf{int}$ | Typaxiom |
| (4) | $\{x : \mathbf{int}\} \triangleright x + 1 : \mathbf{int}$ | (R2) mit Prämissen (1), (2) und (3) |
| (5) | $\{x2 : \mathbf{int}\} \triangleright x2 : \mathbf{int}$ | Typaxiom |
| (6) | $\{x : \mathbf{int}, x2 : \mathbf{int}\} \triangleright x2 + x + 1 : \mathbf{int}$ | (R2) mit Prämissen (4), (5) und (2) |
| (7) | $\{x : \mathbf{int}\} \triangleright \mathbf{fn} x2 \Rightarrow x2 + x + 1 : \mathbf{int} \rightarrow \mathbf{int}$ | (R4) mit Prämisse (6) |
| (8) | $\emptyset \triangleright 2 : \mathbf{int}$ | Typaxiom |
| (9) | $\emptyset \triangleright * : \mathbf{int} * \mathbf{int} \rightarrow \mathbf{int}$ | Typaxiom |
| (10) | $\{x : \mathbf{int}\} \triangleright x * 2 : \mathbf{int}$ | (R2) mit Prämissen (1), (8) und (9) |
| (11) | $\{x : \mathbf{int}\} \triangleright (\mathbf{fn} x2 \Rightarrow x2 + x + 1)(x * 2) : \mathbf{int}$ | (R5) mit Prämissen (7) und (10) |
| (12) | $\{x : \mathbf{int}\} \triangleright \mathbf{let} \mathbf{val} x2 = x * 2 \mathbf{in} x2 + x + 1 \mathbf{end} : \mathbf{int}$ | (R6) mit Prämisse (11) |

Hinweis: Die in (R6) angegebene Beziehung zwischen Funktionsabstraktion und lokaler Bindung gilt nicht nur für die Typisierung, es ist auch $W((\mathbf{fn} x \Rightarrow s)(t)) = W(\mathbf{let} \mathbf{val} x = t \mathbf{in} s \mathbf{end})$.

Aufgabe 7-5

Benutzerdefinierte Ausnahmen

(keine Abgabe)

(* Loesung für alle Teilaufgaben *)

(* Teilaufgabe (a) *)

```

exception Prozent_Kleiner_0;;
exception Prozent_Groesser_100;;

fun nichtwaehler pw =
  if pw < 0.0
  then raise Prozent_Kleiner_0
  else if pw > 100.0
  then raise Prozent_Groesser_100
  else 100.0 - pw;;

(* Teilaufgabe (b) *)

fun constantly (n:int) =
  fn x => n;;

fun klassifiziere (f, n) =
  (constantly 0 (f n))
  handle Prozent_Kleiner_0 => ~1
  | Prozent_Groesser_100 => 1;;

(* oder *)

fun klassifiziere_2 (f, n) =
  ((constantly 0 (f n))
   handle Prozent_Kleiner_0 => ~1)
  handle Prozent_Groesser_100 => 1;;

(* Teilaufgabe (c) *)

klassifiziere(nichtwaehler, 20.0);
klassifiziere(nichtwaehler, ~20.0);
klassifiziere(nichtwaehler, 125.0);

```