TECHNISCHE UNIVERSITÄT MÜNCHEN
FAKULTÄT FÜR INFORMATIK

WS 10/11

Einführung in die Informatik II **Übungsblatt 9**

Univ.-Prof. Dr. Andrey Rybalchenko, A. Herz, K. Apinis 17.12.2010

Dieses Blatt behandelt Kapitel 1 - 4 aus dem Buch zur Vorlesung.

Aufgabe 1 (syntax) Geben Sie die Baumdarstellungen der folgenden durch Zeichendarstellungen beschriebenen Phrasen an.

a) `int * real → real * int`

b) `('a → 'b) → 'a list → 'b list`

c) `if ~x = y then g 3 else #3 p`

d) `val x = f (x + y)`

e) `fun p (x:int,n:int) = if n>0 then x*p(x,n-1) else 1`

Aufgabe 2 (list) Schreiben Sie eine Prozedur

a) `takeWhile : ('a → bool) → 'a list → 'a list`

welche alle Elemente aus der übergebenen Liste, die vor dem ersten Element sind, für die die übergebene Funktion false liefert, zurück gibt.

`takeWhile (fn x⇒true) [2,3,4,1]` $\rightsquigarrow [2,3,4,1]$
`takeWhile (fn x⇒false) [2,3,4,1]` $\rightsquigarrow []$
`takeWhile (fn x⇒x<=3) [2,3,4,1]` $\rightsquigarrow [2,3]$

b) `repl : int list → int list`

welche jede Zahl aus der Liste n mal wiederholt, wobei n gleich der Zahl aus der Liste ist:

`repl [2,3,1]` $\rightsquigarrow [2,2,3,3,3,1]$
`repl [4]` $\rightsquigarrow [4,4,4,4]$

Aufgabe 3 (verstehen) Geben Sie den Wert von result im folgenden Programm an:

```
val my_list = [1,2,9,3,4];

val rec something_else = fn
  nil ⇒ raise Empty
  | (x::xs) ⇒ fn 0 ⇒ x
               |  n ⇒ something_else xs (n-1);

val do_something_with_a_list = foldl (fn ((a:int),(b:int list)) ⇒ a::b) nil;

val nth_3_tuple =
  fn (a,b,c) ⇒
    fn 1 ⇒ a
     | 2 ⇒ b
```

```
    | 3 ⇒ c
    | _ ⇒ raise Empty;

val result = something_else (do_something_with_a_list (nth_3_tuple ([1,2,3,4],
  my_list@my_list, do_something_with_a_list my_list) 2))3;
```

Aufgabe 4 (foldl/foldr) Implementieren sie die Funktionen `List.filter`, `List.map`, `List.concat` und `List.partition` mithilfe von `foldl` oder `foldr`.

Aufgabe 5 (rekursion+fallunterscheidung) Schreiben Sie eine rekursive SML Prozedur, welche die n-te Fibonacci Zahl berechnet und nicht divergiert falls $n < 0$.

$$F(n) = \begin{cases} 0 & \text{if } n = 0 \\ 1 & \text{if } n = 1 \\ F(n-1) + F(n-2) & \text{otherwise} \end{cases}$$

a) Nur mit if Statatements,

b) mit Prozeduren und Mustern

c) mit **case .. of**.

Aufgabe 6 (verstehen/typen)

a) Geben Sie den Wert von `res4` im folgenden Programm an:

```
fun g x f = f x
fun h f g = f 3
fun p x y = x + y
fun d x y = y div y
val res1 = g
val res2 = g (g 1 g p)
val res3 = g (g 1 g p) h
val res4 = g (g 1 g p) h d
val res5 = h
val res6 = d
val res7 = h d
```

b) Geben Sie die Typen der Werte res1 .. res7 an.

Aufgabe 7 (strings) Schreiben Sie eine Prozedur `remove: string → char → string` welche mittels einer der beiden fold Prozeduren alle Vorkommen des übergebenen Zeichens aus dem String löscht. Die Reihenfolge der Zeichen im String soll dabei erhalten bleiben.
Beispiel: `remove "abcdc" \#"c" = "abd"`

```
foldl : ('a * 'b → 'b) → 'b → 'a list → 'b
foldr : ('a * 'b → 'b) → 'b → 'a list → 'b
```

Aufgabe 8 (verstehen) Evaluieren sie die folgenden Ausdrücke:

a) `if 3>4 then 5+5 else 2`

b)
```
let fun p x = x mod 2 = 0
in p 5 andalso p 6 end
```

c)
```
let fun p 0 = true
      | p x = not (p (x-1))
in p 500 end
```

d)
```
let fun p x = if x>100 then x else p (x*2+15)
in p 13 end
```

e)
```
let fun p x 0 = []
      | p x n = x :: p (x-1) (n-2)
in p 7 8 end
```

f)
```
map (fn x ⇒ 2*x) [~1,0,1]
```

g)
```
foldl (fn (x,a) ⇒ a*10+x) 0 [7,7,4]
```

Aufgabe 9 (typen)

Geben Sie die Typschemen an, mit denen die folgenden Bezeichner typisiert werden.

a) `fun f x y z = if x=y then z else 0`

b) `fun g x y z = ((x,y,3.0),z)`

c) `fun h x y z = z (x,y)`

d) `fun p x y z = y x (z,2)`

e) `fun q x y z = Math.pow (x, y) + z`

Aufgabe 10 (typen+tupel) Geben Sie die Typen der folgenden Funktionen an:

```
fun f (x,y,z) t = if x = y then z + 1 else if x>y then z else y+t;
fun w x y z = x (y z);
```

Aufgabe 11 (typen+tupel) Schreiben Sie eine Prozedur welche die Hauptdiagonale einer quadratischen Matrix, welche als Liste von Listen von ints dargestellt wird, zurück gibt.
Hinweis: Matrix

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

ist die darstellung für `[[1,2],[3,4]]`