Geordnete Binärbäume. Die linken Nachkommen eines Knotens sind stets kleiner, die rechten stets größer (oder gleich).

```
data OrdBintree a = Ext | In a (OrdBintree a) (OrdBintree a)
  deriving Show
insert :: Ord a => a→OrdBintree a→OrdBintree a
insert x Ext = In x Ext Ext
insert x (In y yl yr) =
  if x < y
  then In y (insert x yl) yr
  else In y yl (insert x yr)
listtotree :: Ord a => [a]→OrdBintree a
listtotree [ ] = Ext
listtotree (x : xs) = insert x (listtotree xs)
```

Durch *Ord a => a* wird ein beliebiger Typ bezeichnet, der aber geordnet ist. Sonst kann $<$ nicht verwendet werden.

Funktionen höherer Ordnung

Eine besonders mächtiges Werkzeug der funktionalen Programmierung sind Funktionen, die Funktionen auf Funktionen abbilden. Wir nennen diese auch Funktionen höhrerer Ordnung.

Einfaches Beispiel:

Wandle eine Funktion $f : A \times B \to C$ in eine Funktion $g : B \times A \to C$ um, so daß $g(x, y) = f(y, x)$ gilt.

```
umdrehen :: (a→b→c)→(b→a→c)
umdrehen f x y  =  f y x
loesche :: Integer→[Integer]→[Integer]
loesche x [ ]  =  [ ]
loesche x (y : ys)  =
  if x == y
  then loesche x ys
  else y  :  loesche x ys
uloesche  =  umdrehen loesche
```

Mit *toweroftwo* 4 können wir $2^{2^{2^2}}$ berechnen.

```
zweimal :: (a→a)→(a→a)
zweimal f x = f (f x)

kmal :: Integer→(a→a)→(a→a)
kmal 0 f x = x
kmal n f x = f (kmal (n − 1) f x)

zweierpotenz :: Integer→Integer
zweierpotenz k = kmal k (∗2) 1

toweroftwo :: Integer→Integer
toweroftwo k = kmal k zweierpotenz 1
```

Mit *nach* können wir aus f und g ihre Komposition $f \circ g$ berechnen.

```
nach :: (b→c)→(a→b)→(a→c)
nach f g = \x → f (g x)
```

Beispiel:

$f = nach\ (*2)\ (+4)$ definiert die Funktion

$$f : n \mapsto 2(n+4).$$

Anstatt $f = nach\ g\ h$ können wir auch $f = g$ `` `nach` `` h schreiben.

Eine solche Funktion ist schon vordefiniert: $f = g\ .\ h$

lmap wendet eine Funktion auf alle Elemente einer Liste an.

```
lmap :: (a→b)→[a]→[b]
lmap f [ ]  =  [ ]
lmap f (x : xs)  =  f x  :  lmap f xs
```

Beispiel:

lmap $(*2)$ $[1,2,3,4,5,6]$

lmap head $[[1,2],\ [3,4],\ [5,6]]$

doppeln $=$ *lmap* $(*2)$

Auch *lmap* ist schon vordefiniert und heißt einfach *map*.

Lambda-Ausdrücke

Wir können eine Funktion $n \mapsto 5n$ definieren, ohne ihr einen Namen zu geben:

$\backslash n \rightarrow 5 * n$

Dies ist an den λ-Kalkül angelehnt. Dort schreibt man $\lambda n.5n$.

Beispiel:

$map\ (\backslash x \rightarrow 2 * x + 5)\ [1, 2, 3, 4, 5, 6]$

$umgekehrt\ f\ =\ \backslash x\ y \rightarrow f\ y\ x$

Frage:

Was ergibt $(\backslash x\ y \rightarrow (/)\ y\ x)\ 2\ 3$?