Sortieren

- **Sortieren durch Einfügen** (lineare Rekursion)

```
fun insert (x,nil) = [x]
  | insert (x,y::yr) = if x<=y
                          then x::y::yr else y::insert(x,yr);
fun isort xs = foldl insert nil xs
```

- **Sortieren durch Mischen** (Baumrekursion)

```
fun merge (nil, ys) = ys
  | merge (xs, nil) = xs
  | merge (x::xr, y::yr) = if x<=y
                          then x::merge(xr, y::yr)
                          else y::merge(x::xr, yr)
fun msort [] = []
  | msort [x] = [x]
  | msort xs = let val (ys,zs) = split(xs)
                          in merge (msort ys, msort zs) end
```

Ordnungen

- Typ `order`, 3 Werte: `LESS, EQUAL, GREATER`
- Vergleichsprozeduren:

```
Int.compare:   int * int -> order
Real.compare:   real*real -> order
String.compare:   string*string -> order
```

- **Polymorphes sortieren:**

```
fun pisort compare =
let
   fun insert (x,nil) = [x]
   | insert(x,y::yr) = case compare(x,y) of
     GREATER => y::insert(x,yr)
     _ => x::y::yr
in foldl insert nil
```