Übung

Man definiere OCAML-Signaturen für die folgenden Rechenstrukturen:

- Stapel
- Binärbaum
- Reihung
- Schlange
- Ring (Menge mit Konstanten 0, 1 und Operationen $+, \times$)
- Endliche Funktionen (implementierbar etwa mit Assoziationslisten)
- Endliche Mengen

Anwendung: Breitenordnung

Man gebe die Knoten eines Binärbaums in der *Breitenordnung* aus, d.h., Knoten mit kleinerer Höhe zuerst, bei Knoten gleicher Höhe die weiter links stehenden zuerst.

```
let t = Build(6,Build(3,Build(2,Empty,Empty),
                         Build(8,Empty,Build(5,Empty,Empty))),
              Build(8,Build(4,Empty,Empty),Empty)
             )
linbreit t;;
- : int list = [6; 3; 8; 2; 8; 4; 5]
```

Lösung: man schreibe eine allgemeinere Funktion, die die Knoten einer *Schlange* von Bäumen in der Breitenordnung ausgibt.

Bemerkung: ersetzt man “Schlange” durch “Stapel”, so erhält man eine effiziente Implementierung der Vorordnung.

Überschattung

- Variablen (Funktionsparameter, let-gebundene Variablen, etc.) haben einen bestimmten *Gültigkeitsbereich* in dem sie unter ihrem Namen verwendbar sind.
- Ausnahme: innerhalb ihres Gültigkeitsbereichs können Variablen durch andere Variablen des gleichen Namens *verschattet* (auch *überschattet*) werden.
- Die in Kapitel 3 eingeführte Semantik trägt diesen Umständen bereits Rechnung.
- Parameter einer übergeordneten Funktion bleiben in einer untergeordneten Funktion sicht- und verwendbar. Man muss sie daher nicht explizit übergeben. [Kröger] bezeichnet das als *Parameterunterdrückung*.

Beispiel: Suche in Reihungen

```
let enthalten1(x,a) =
   let rec enthallg(x,k,a) =
          if k>dim(a) then false
          else a=get(x,k) || enthallg(x,k+1,a)
   in enthallg(x,1,a)
let enthalten1'(x,a) =
   let rec enthallg k =
          if k>dim(a) then false
          else a=get(x,k) || enthallg(k+1)
   in enthallg 1
```