Aufgabe 7-1

Gegeben sei der Datentyp eines Binärbaums über einem Grundtyp `T`, auf dem eine totale Ordnung `<=` gegeben ist:

```
BinTree[T:TYPE+, <=:(total_order?[T])] : DATATYPE
BEGIN
  empty : empty?
  node(key:T, left:BinTree, right:BinTree) : node?
END BinTree
```

Ein Binärbaum sei sortiert, wenn für alle Knoten gilt, dass alle Werte im linken Teilbaum des Knotens höchstens so groß wie der Wert des Knoten und alle Werte im rechten Teilbaum mindestens so groß wie der Wert des Knotens sind.

- Definieren Sie eine Funktion `sort`, die einen beliebigen Binärbaum sortiert. Beweisen Sie die Korrektheit Ihrer Funktion!

Hinweis: Eine mögliche Vorgehensweise ist, die Sortierfunktion in zwei Phasen zu unterteilen: $sort = mktree \circ flatten$. Die Funktion *flatten* traversiert dabei den Baum und liefert seine Elemente in einer Liste zurück, welche die Funktion *mktree* sukzessive in einen zu Beginn leeren Baum sortiert einfügt.

Unterteilen Sie analog den Korrektheitsbeweis, indem Sie geeignete Lemmata formulieren.

Aufgabe 7-2

a) Definieren Sie in PVS eine rekursive Funktion `mod(n:nat,k:posnat)`[1] zur Berechnung des Rests einer natürlichen Zahl `n` bei der Division durch `k`.

b) Zeigen Sie mittels *starker* Induktion:

```
n,m : VAR nat
k   : VAR posnat

mod_defn : PROPOSITION
  EXISTS m: n = m*k + mod(n,k)
```

Hinweis: Um in einem Induktionsbeweis das starke Induktionsprinzip anzuwenden, muss dem Befehl

[1] Beachte den Typ des zweiten Arguments!

```
(induct/$ var &optional (fnum 1) name)
```

über das optionale Argument `:name` der Name des starken Induktionsprinzips, hier `NAT_induction` übergeben werden. Genaueres siehe `HELP` Kommando.

Aufgabe 7-3

a) Definieren Sie in PVS eine rekursive Funktion `ggT(x,y)`, die den größten gemeinsamen Teilers zweier positiver natürlicher Zahlen nach dem Differenzen-Algorithmus berechnet:

$$ggT(x,y) = \begin{cases} x & \text{falls } x = y \\ ggT(x-y,y) & \text{falls } x > y \\ ggT(x,y-x) & \text{sonst} \end{cases}$$

b) Zeigen Sie mittels *Maßinduktion*: `ggT(x,y) = ggT(y,x)`

Hinweise:

- Wagemutige können die in PVS vordefinierte lexikographische Ordnung `lex2` zusammen mit der auf Ordinalzahlen definierten Relation `<` als `MEASURE` benutzen. Schlaue definieren eine einfachere Maßfunktion!
- Beweise durch Maßinduktion werden in PVS mit dem Befehl

  ```
  (MEASURE-INDUCT+ measure vars)
  ```

  eingeleitet, wobei `measure` eine geeignete Maßfunktion ist, die freie Variablen aus `vars` enthält. Genaueres siehe `HELP` Kommando.

Aufgabe 7-4

Beweisen Sie die Gültigkeit des Prinzips der Noetherschen Induktion:

```
noethersch [T:TYPE] : THEORY
BEGIN
  noethersche_induktion: LEMMA
    FORALL (<:pred[[T,T]]):
      well_founded?[T](<) IMPLIES
        FORALL (P:pred[T]):
          (FORALL (x:T): (FORALL (y:T): y<x IMPLIES P(y)) IMPLIES P(x))
        IMPLIES
          FORALL (x:T): P(x)
END noethersch
```