Aufgabe 9-1

Zur Modellierung einfacher Termstrukturen sei folgende Datentyp-Deklaration in PVS gegeben:

```
S: TYPE    arity: [S -> nat]   V: TYPE

Term : DATATYPE
BEGIN
   vterm(vsym: V): vterm?
   fterm(fsym: S, args: [below(arity(fsym)) -> Term]): fterm?
END Term
```

Der Typ `S` modelliert eine Menge von *Funktionssymbolen*, welchen durch die Funktion `arity` jeweils eine *Stelligkeit* zugeordnet wird. Der Typ `V` bezeichnet die Menge an *Variablensymbolen*.

Die betrachteten Terme bestehen also entweder nur aus einem Variablensymbol `vsym`, ausgedrückt durch den Konstruktor `vterm`, oder einem Funktionsymbol `fsym` und der zur Stelligkeit von `fsym` entsprechenden Anzahl von Argumenttermen `args`.

Beispiel: Sei f ein 2-stelliges Funktionssymbol, d. h. `arity(f) = 2`, und x und y Variablensymbole. Der Term $f(x, y)$ kann somit durch den Ausdruck

```
fterm(f,lambda (i:below(2)): IF i = 0 THEN vterm(x) ELSE vterm(y) ENDIF)
```

dargestellt werden.

a) Eine Funktion `s:[V -> Term]` nennen wir *Variablenbelegung*. Definieren Sie eine Funktion `subst(s:[V->Term])(t:Term) : Term`, die gemäß der Variablenbelegung `s` Variablen durch die zugeordneten Terme ersetzt.

b) Definieren Sie ein Prädikat `substitution? : pred[[Term -> Term]]`, das beschreibt, wann es sich bei einer Funktion `sigma:[Term -> Term]` um eine *Substitution* im oben beschrieben Sinne handelt. Definieren Sie damit einen Typ `Substitution` der Substitutionsfunktionen.

c) Definieren Sie ein Prädikat `unifier?(s,t) : pred[Substitution]`, das beschreibt, wann es sich bei einer Substitution `sigma` um einen *Unifikator* der Terme `s` und `t` handelt. Definieren Sie damit einen Typ `Unifikator(s,t:term)` der Unifikatoren zweier Terme `s` und `t`.

d) Definieren Sie ein Prädikat `MGU?(s,t:Term) : pred[Unifikator(s,t)]`, das beschreibt, wann es sich bei einem Unifikator `sigma` von `s` und `t` um den allgemeinsten Unifikator handelt.

e) Definieren Sie eine Funktion `unify(s,t:Term) : set[Unifikator(s,t)]`, die den allgemeinsten Unifikator von `s` und `t` berechnet. Das Resultat der Funktion soll eine einelementige Menge bestehend aus dem allgemeinsten Unifikator sein, bzw. die leere Menge, falls `s` und `t` nicht unifizierbar sind.

f) Zeigen Sie die Korrektheit Ihrer Unifikationsfunktion:

```
unify_correct : THEOREM
   FORALL (s,t:Term, sigma:Unifikator(s,t)):
      member(sigma,unify(s,t)) => MGU?(s,t)(sigma)
```