Aufgabe 12-1

Wir betrachten einfache Termstrukturen über einer Menge V von *Variablensymbolen* und einer Menge S von *Funktionssymbolen*. Ein Term sei dabei entweder eine *Variable* $v \in V$, oder ein *Funktionsterm* $f(t_1, \ldots, t_n)$, mit $f \in S$ und Termen $t_1, \ldots, t_n$.

a) Definieren Sie in PVS einen induktiven Datentyp `Term` zur Modellierung solcher Termstrukturen.

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

c) 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.

d) 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`.

e) 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.

f) Optional: Definieren Sie eine Funktion `unify`, die den allgemeinsten Unifikator von `s` und `t` berechnet.

g) Optional: Zeigen Sie die Korrektheit Ihrer Unifikationsfunktion.

Aufgabe 12-2

Wir betrachten Fixpunkte (monotoner) Prädikatentransformer in PVS. Dazu seien die folgenden Typvereinbarung gegeben:

```
T           : TYPE+
D           : TYPE = pred[T]
Transformer : TYPE = [D -> D]
```

a) Definieren Sie eine geeignete Halbordnungsrelation `<=` auf `D`. Zeigen Sie auch, dass `<=` in der Tat die Halbordnungseigenschaft besitzt.

b) Seien `s : VAR D` und `A : VAR set[D]`. Definieren Sie Prädikate `unS(s,A)` und `grUnS(s,A)` die angeben, wann `s` eine (größte) untere Schranke von `A` in `D` ist.

c) Definieren Sie ein Prädikat `bot`, das das kleinste Prädikat bzgl. `<=` ist. Zeigen Sie:

```
bot_is_bot : FACT
  FORALL (p:D): bot <= p

bot_is_unS : FACT
  FORALL (A:set[D]): unS(bot,A)
```

d) Definieren Sie eine Funktion `inf : [set[D] -> D]`, so dass `inf(A)` die größte untere Schranke von `A` ist. Zeigen Sie diesen Sachverhalt:

```
inf_is_grUnS : FACT
  FORALL (A:set[D]): grUnS(inf(A),A)
```

e) Sei `t : VAR Transformer`. Definieren Sie ein Prädikat `FP?(t) : pred[D]`, so dass `FP?(t)(p)` genau dann wahr ist, wenn `p` ein Fixpunkt von `t` ist.

Definieren Sie analog dazu ein Prädikat `klFP?(t) : pred[D]`, das den kleinsten Fixpunkt von `t` charakterisiert.

f) Definieren Sie ein Prädikat `monoton`, das für einen Transformer `t` angibt, ob `t` monoton ist.

g) Für monotone Prädikatentransformer `tau` sei definiert:

```
p       : VAR D
tau     : VAR (monoton?)
mu(tau) : D = inf({p | tau(p) <= p})
```

h) Zeigen Sie das Parksche Lemma:

```
parks_lemma : LEMMA
  tau(p) <= p IMPLIES mu(tau) <= p
```

i) Zeigen Sie, dass `mu(tau)` der kleinste Fixpunkt von `tau` ist:

```
mu_is_klFP : THEOREM
  klFP?(tau)(mu(tau))
```

Hinweis: Unterteilen Sie den Beweis mit Hilfe geeigneter Lemmata in kleinere Schritte. Versuchen Sie, das Parksche Lemma sowie die Monotonie-Eigenschaft von `tau` auszunutzen.

j) Definieren Sie die transitive Hülle einer binären Relation als Fixpunkt eines geeigneten Transformers. Zeigen Sie auch die Korrektheit der so definierten Hülle.

Hinweis: Erstellen Sie für die transitive Hülle eine separate Theorie und *importieren* Sie die Fixpunktmodellierung wie folgt:

```
fixpunkte [T:TYPE+] : THEORY
BEGIN
  ...
END fixpunkte

transclosure [T:TYPE+] : THEORY
BEGIN

  IMPORTING fixpunkte[[T,T]]

  ...
END transclosure
```