Lösungen zu den Aufgaben zur Vorlesung "*Grundlagen der Informatik*"

Lösung Aufgabe 1:

(a) Folgende Prozedur leistet das Gewünschte:

```
procedure both(l1, l2);
    return { x in l1 | x in l2 };
end both;
```

(b) Folgende Prozedur leistet das Gewünschte:

```
procedure pythagoras(n);
    return { [ x, y, z ] : x in {1..n}, y in {1..n}, z in {1..2*n}
                          | x*x + y*y = z*z and x * y < n
           };
end pythagoras;
```

Lösung Aufgabe 2:

(a) Folgende Prozedur leistet das Gewünschte:

```
procedure compose(A1, A2);
    return { [[x,z], d1 + d2] :
             [[x,y1], d1] in A1, [[y2,z], d2] in A2 | y1 = y2 };
end compose;
```

(b) Folgende Prozedur leistet das Gewünschte:

```
procedure closure(A);
    C    := A;
    OldC := {};
    while C /= OldC loop
        OldC := C;
        C := C + compose(C, C);
        print(C);
    end loop;
    return C;
end closure;
```

Lösung Aufgabe 3: Die folgende Implementierung leistet das Gewünschte:

```
procedure toLists(M);
    if M = {} then
        return { [] };
    end if;
    return { [ x ] + l : x in M, l in toLists(M - {x}) };
end toLists;
```

Lösung Aufgabe 4:

(a) Wir führen die folgenden aussagenlogischen Variablen als Abkürzungen ein.

1. Linker Motor fällt aus: l
2. Rechter Motor fällt aus: r
3. Flugzeug stürzt ab: a

Die gesuchte Schluss-Regel ist:

$$\frac{l \vee r \rightarrow a \qquad a \qquad \neg l}{r}$$

(b) Die Schluss-Regel ist genau dann korrekt, wenn die Formel

$$(l \vee r \rightarrow a) \wedge a \wedge \neg l \rightarrow r$$

eine Tautologie ist. Wir formen diese Formel in KNF um:

$$\begin{aligned}
& (l \vee r \rightarrow a) \wedge a \wedge \neg l \rightarrow r \\
\Leftrightarrow \quad & \neg\big((l \vee r \rightarrow a) \wedge a \wedge \neg l\big) \vee r \\
\Leftrightarrow \quad & \neg\big((\neg(l \vee r) \vee a) \wedge a \wedge \neg l\big) \vee r \\
\Leftrightarrow \quad & \neg(\neg(l \vee r) \vee a) \vee \neg a \vee l \vee r \\
\Leftrightarrow \quad & ((l \vee r) \wedge \neg a) \vee \neg a \vee l \vee r \\
\Leftrightarrow \quad & (l \vee r \vee \neg a \vee l \vee r) \wedge (\neg a \vee \neg a \vee l \vee r) \\
\Leftrightarrow \quad & (l \vee r \vee \neg a)
\end{aligned}$$

Diese Formel ist keine Tautologie. Ein Gegenbeispiel ist durch die Belegung

$$\mathcal{I} := \big\{\langle l, \texttt{false}\rangle, \langle r, \texttt{false}\rangle, \langle a, \texttt{true}\rangle\big\}$$

gegeben. Diese Belegung hätte man auch ohne die Rechnung angeben dürfen.

Lösung Aufgabe 5:

(a) Die gesuchte Formel ist

$$(q \wedge p) \wedge (r \vee \neg q) \rightarrow r \wedge p.$$

(b) Wir formen diese Formel in konjunktive Normalform um:

$$\begin{aligned}
& (q \wedge p) \wedge (r \vee \neg q) \rightarrow r \wedge p \\
\Leftrightarrow \quad & \neg\big((q \wedge p) \wedge (r \vee \neg q)\big) \vee (r \wedge p) \\
\Leftrightarrow \quad & \big(\neg(q \wedge p) \vee \neg(r \vee \neg q)\big) \vee (r \wedge p) \\
\Leftrightarrow \quad & \big((\neg q \vee \neg p) \vee (\neg r \wedge q)\big) \vee (r \wedge p) \\
\Leftrightarrow \quad & \big((\neg q \vee \neg p \vee \neg r) \wedge (\neg q \vee \neg p \vee q)\big) \vee (r \wedge p) \\
\Leftrightarrow \quad & \big((\neg q \vee \neg p \vee \neg r) \wedge \top\big) \vee (r \wedge p) \\
\Leftrightarrow \quad & (\neg q \vee \neg p \vee \neg r) \vee (r \wedge p) \\
\Leftrightarrow \quad & (\neg q \vee \neg p \vee \neg r \vee r) \wedge (\neg q \vee \neg p \vee \neg r \vee p) \\
\Leftrightarrow \quad & \top \wedge \top \\
\Leftrightarrow \quad & \top \\
\Leftrightarrow \quad & \{\}
\end{aligned}$$

Lösung Aufgabe 6:

(a) Die folgende Prozedur leistet das Gewünschte:

```
procedure allVars(E);
    Operators := {"+", "-", "*", "/"};
    case
        when is_string(E)      => return { E };
        when E(2) in Operators => return allVars(E(1)) + allVars(E(3));
        otherwise              => abort("Error in allVars(" + E + ")");
    end case;
end allVars;
```

(b) Die folgende Prozedur leistet das Gewünschte:

```
procedure countVars(E, x);
    count := 0;
    Ops   := {"+", "-", "*", "/"};
    case
        when x = E        => count := count + 1;
        when is_string(E) => count := count;
        when E(2) in Ops  => return countVars(E(1),x) + countVars(E(3),x);
        otherwise         => abort("Error in countVars(" + E + ")");
    end case;
    return count;
end countVars;
```

(c) Die folgende Prozedur leistet das Gewünschte:

```
procedure singleVars(E);
    return { x in allVars(E) | countVars(E,x) = 1 };
end singleVars;
```

Lösung Aufgabe 7: Die folgende Kette von Anwendungen der Schnitt-Regel weist die Behauptung nach:

1. $\{\neg t, \neg p\}, \{\neg t, p\} \vdash \{\neg t\}$,
2. $\{\neg t\}, \{t, \neg q\} \vdash \{\neg q\}$,
3. $\{\neg q\}, \{q, \neg p\} \vdash \{\neg p\}$,
4. $\{\neg p\}, \{t, p, \neg r\} \vdash \{t, \neg r\}$,
5. $\{t, \neg r\}, \{\neg t\} \vdash \{\neg r\}$,
6. $\{\neg r\}, \{t, q, r\} \vdash \{t, q\}$,
7. $\{t, q\}, \{\neg t\} \vdash \{q\}$,
8. $\{q\}, \{\neg q\} \vdash \{\}$.

Lösung Aufgabe 8: Da die Menge M keine Unit-Klausel enthält, beginnen wir mit einer Fall-Unterscheidung. Wir führen die Fall-Unterscheidung bezüglich p durch:

1. Als erstes betrachten wir daher die Menge $M_0 = M \cup \big\{\{p\}\big\}$. Wir berechnen

   $$M_1 := \mathit{reduce}(M_0, p) = \big\{\ \{r, \neg t\}, \{\neg r, q\}, \{\neg q, s\}, \{\neg r, \neg s\}, \{\neg s\}, \{q, t\}, \{p\}\ \big\}.$$

   M_1 enthält die neue Unit-Klausel $\{\neg s\}$. Daher berechnen wir nun

   $$M_2 := \mathit{reduce}(M_1, \neg s) = \big\{\ \{r, \neg t\}, \{\neg r, q\}, \{\neg q\}, \{q, t\}, \{p\}, \{\neg s\}\ \big\}.$$

   M_2 enthält die neue Unit-Klausel $\{\neg q\}$. Daher berechnen wir nun

   $$M_3 := \mathit{reduce}(M_2, \neg q) = \big\{\ \{r, \neg t\}, \{\neg r\}, \{t\}, \{p\}, \{\neg s\}, \{\neg q\}\ \big\}.$$

   M_3 enthält die neue Unit-Klausel $\{t\}$. Daher berechnen wir nun

   $$M_4 := \mathit{reduce}(M_3, t) = \big\{\ \{r\}, \{\neg r\}, \{p\}, \{\neg s\}, \{\neg q\}, \{t\}\ \big\}.$$

   Da M_4 sowohl die Unit-Klausel $\{r\}$ als auch die Unit-Klausel $\{\neg r\}$ enthält, ist M_4 nicht lösbar.

2. Nun betrachten wir die Menge $M \cup \big\{\{\neg p\}\big\}$.

   $$M_5 := \mathit{reduce}(M_0, \neg p) = \big\{\ \{q\}, \{r, s\}, \{r, \neg s\}, \{\neg q, s\}, \{\neg r, \neg s\}, \{\neg p\}\ \big\}$$

   M_5 enthält die neue Unit-Klausel $\{q\}$. Daher berechnen wir nun

   $$M_6 := \mathit{reduce}(M_5, q) = \big\{\ \{r, s\}, \{r, \neg s\}, \{s\}, \{\neg r, \neg s\}, \{\neg p\}, \{q\}\ \big\}$$

   M_6 enthält die neue Unit-Klausel $\{s\}$. Daher berechnen wir nun

   $$M_7 := \mathit{reduce}(M_6, s) = \big\{\ \{r\}, \{\neg r\}, \{\neg p\}, \{q\}, \{s\}\ \big\}$$

   Da M_7 sowohl die Unit-Klausel $\{r\}$ als auch die Unit-Klausel $\{\neg r\}$ enthält, ist M_7 nicht lösbar.

Da sowohl $M \cup \big\{\{p\}\big\}$ als auch $M \cup \big\{\{\neg p\}\big\}$ unlösbar sind, ist auch die Menge M unlösbar. $\square$

Lösung Aufgabe 9: Wir setzen $\mathcal{U} := \{a, b\}$ und definieren die Interpretation $p^{\mathcal{J}}$ des Prädikats-Zeichens p als

$$p^{\mathcal{J}} := \big\{\langle a, a\rangle, \langle b, b\rangle\big\}.$$

Dann gibt es in U zwar für jedes x ein y, so dass $p(x, y)$ gilt, denn wir können $y = x$ setzen. Aber es gibt kein y, so dass für alle x die Formel $p(x, y)$ gilt. Wollten wir dies mit dem in der Vorlesung gezeigten Programm testen, so könnten wir folgendes definieren:

```
a := "a";
b := "b";
U := { a, b };
pJ := { [[a,a], true], [[a,b], false], [[b,a], false], [[b,b], true] };
J := { [ "p", pJ ] };
S := [ U, J ];
I := { [ "x", a ], [ "y", b ] };
f := parse("(forall x: exists y: p(x,y)) -> exists y: forall x: p(x,y)");
```

Dann würde der Aufruf "`evalFormula(f, S, I)`" den Wert `false` liefern.

Lösung Aufgabe 10:

$$\begin{array}{lll}
 & \Big(\forall x : \neg\exists y : \big(q(x) \rightarrow p(x, y)\big)\Big) \rightarrow \big(\forall z : q(z)\big) & \\
\Leftrightarrow & \neg\Big(\forall x : \neg\big(\exists y : q(x) \rightarrow p(x, y)\big)\Big) \vee \big(\forall z : q(z)\big) & \\
\Leftrightarrow & \Big(\exists x : \exists y : \big(q(x) \rightarrow p(x, y)\big)\Big) \vee \big(\forall z : q(z)\big) & \\
\Leftrightarrow & \exists x : \Big(\exists y : \big(q(x) \rightarrow p(x, y)\big) \vee \big(\forall z : q(z)\big)\Big) & \\
\Leftrightarrow & \exists x : \exists y : \Big(\big(q(x) \rightarrow p(x, y)\big) \vee \big(\forall z : q(z)\big)\Big) & \\
\Leftrightarrow & \exists x : \exists y : \forall z : \Big(\big(q(x) \rightarrow p(x, y)\big) \vee q(z)\Big) & \\
\Leftrightarrow & \exists y : \forall z : \Big(\big(q(s_1) \rightarrow p(s_1, y)\big) \vee q(z)\Big) & \text{mit der Skolem-Konstante } s_1 \text{ für } x \\
\Leftrightarrow & \forall z : \Big(\big(q(s_1) \rightarrow p(s_1, s_2)\big) \vee q(z)\Big) & \text{mit der Skolem-Konstante } s_2 \text{ für } y \\
\Leftrightarrow & \forall z : \neg q(s_1) \vee p(s_1, s_2) \vee q(z) & \\
\Leftrightarrow & \big\{\{\neg q(s_1),\ p(s_1, s_2),\ q(z)\}\big\} &
\end{array}$$

Lösung Aufgabe 11: Wir rechnen wie folgt:

$$
\begin{aligned}
& \Big\langle \big\{ f(x_1, x_2, h(x_1)) \doteq f(h(c), h(x_3), x_3) \big\}, [] \Big\rangle \\
\leadsto\;& \Big\langle \big\{ x_1 \doteq h(c), x_2 \doteq h(x_3), h(x_1) \doteq x_3 \big\}, [] \Big\rangle \\
\leadsto\;& \Big\langle \big\{ x_1 \doteq h(c), x_2 \doteq h(x_3), x_3 \doteq h(x_1) \big\}, [] \Big\rangle \\
\leadsto\;& \Big\langle \big\{ x_1 \doteq h(c), x_2 \doteq h(h(x_1)) \big\}, [x_3 \mapsto h(x_1)] \Big\rangle \\
\leadsto\;& \Big\langle \big\{ x_1 \doteq h(c) \big\}, [x_3 \mapsto h(x_1), x_2 \mapsto h(h(x_1))] \Big\rangle \\
\leadsto\;& \Big\langle \big\{ \big\}, [x_3 \mapsto h(h(c)), x_2 \mapsto h(h(h(c))), x_1 \mapsto h(c)] \Big\rangle
\end{aligned}
$$

Damit lautet die Lösung des oben gegebenen syntaktischen Gleichungs-Systems

$$\mu = [x_3 \mapsto h(h(c)), x_2 \mapsto h(h(h(c))), x_1 \mapsto h(c)].$$

Lösung Aufgabe 12:

```
% is_contained(+List(T), +List(T)).

is_contained([], L).

is_contained([ X | R ], L) :-
    member(X, L),
    is_contained(R, L).
```