Logikprogrammierung Blatt 3

Christoph Hübner 711836
Max Winkler 718541

Aufgabe 1: Unifikation

Unifizieren Sie die folgenden Strukturen und geben Sie die dabei ggf. erzeugten Variablenbindungen an.

```
b(x,y)              b(Y,X)
t(r,i)              t(Z,Z)
h(g(F,k),g(k,F))    h(g(m,H),g(H,m))
m(X,c(g),h(X))      m(t(r,s),c(u),h(g(T)),t)
false               not(true)
False               not(true)
```

Unter Verwendung des unify-operators:

```
1 ?- b(x,y) = b(Y,X) .
Y = x,
X = y.

2 ?- t(r,i) = t(Z,Z) .
false.

3 ?- h(g(F,k),g(k,F)) = h(g(m,H),g(H,m)) .
F = m,
H = k.

4 ?- m(X,c(g),h(X)) = m(t(r,s),c(u),h(g(T)),t) .
false.

5 ?- false = not(true) .
false.

6 ?- False = not(true) .
False = not(true).
```

Aufgabe 2: Erreichbarkeit über Reisedaten

a) Schreiben sie ein zweistelliges Prädikat `travel/2`, das angibt, ob man direkt oder indirekt von einem Ort zum anderen Ort gelangen kann. Zum Beispiel sollte die Anfrage `?- travel(valmont,raglan).` mit `true` beantwortet werden.

```
direct(Start, Ziel) :- byTrain(Start, Ziel); byCar(Start, Ziel); byPlane(Start,
Ziel).
```

```
travel(Start, Ziel) :- direct(Start, Ziel).

travel(Start, Ziel) :- direct(Start, Zwischenziel), travel(Zwischenziel, Ziel).
```

b) Schreiben Sie ein erweitertes Prädikat `travel/3`, das auch die Route in einer verschachtelten Struktur mit einem dreistelligen Funktionsnamen `go` ausgibt.

```
travel(X, Y, go(X, Y)) :- direct(X, Y).

travel(X, Y, go(X, Z, Route)) :-
 direct(X, Z),
 travel(Z, Y, Route).
```

c) Erweitern Sie `travel/3` so, dass auch die benötigten Transportmittel angegeben werden.

Zunächst wird ein `direct/3`-Prädikat erstellt.

```
direct(Start, Ziel, train)  :- byTrain(Start, Ziel).
direct(Start, Ziel, car)    :- byCar(Start, Ziel).
direct(Start, Ziel, plane)  :- byPlane(Start, Ziel).
```

```
travel(X, Y, go(X, Y, Means)) :- direct(X, Y, Means).

travel(X, Y, go(X, Z, Means, Route)) :-
 direct(X, Z, Means),
 travel(Z, Y, Route).
```

Aufgabe 3: Peano-Arithmetik

Definieren Sie folgende Prädikate für die PEANO-Arithmetik und testen Sie sie mit geeigneten Beispielen:

a) ein Prädikat, das zwei PEANO-Zahlen im Hinblick auf die Relation ***größer oder gleich*** vergleicht;

```
bigger_than(s(_),0).
bigger_than(0,0).
bigger_than(s(Big),s(Small)) :- bigger_than(Big,Small).
```

b) ein Prädikat für die Subtraktion zweier PEANO-Zahlen;

```
substract(R,0,R).
substract(s(X),s(Y),R) :- substract(X,Y,R).
```

c) Prädikate `odd/1` und `even/1` für ungerade und ungerade Peanozahlen; Hinweis: Definieren sie parallel beide Prädikate durch indirekte Rekursion

```
even(0).
even(s(X)) :- odd(X).
odd(s(X)) :- even(X).
```

d) ein Prädikat `int2peano(+nicht-negative-Integerzahl,?Peanozahl)`, das eine nicht-negative Integerzahl in eine Peanozahl umwandelt.

```
decr(X,NX) :- NX is X-1.
int2peano(0,0).
int2peano(I,s(P)) :- I>0,Q is I-1,int2peano(Q,P).
```