Prolog

Begriffe:

Begriff	Definition	Beispiele
Atom	*Text, klein geschrieben oder in Hochkomma*	ding , 'Alter'
true, false	*Wahrheitswerte*	
Klausel	*Kopf.* oder *Kopf:-Rumpf.* *(Punkt am Ende!)*	blau(X):-rot(X).
Prädikat	*alle Klauseln mit dem gleichen Namen*	
Kopf	*name(Parameterliste)*	blau(rose) , blau(A,B)
Parameterliste	*Liste von Parametern, durch Komma getrennt*	
Parameter	*Atom oder Variable oder _ (ohne Wirkung)*	
Variable	*Wort, groß geschrieben, steht für Atome*	
Rumpf	*Liste von Prädikaten, durch Komma getrennt*	
Fakt	*Klausel ohne Prädikate im Rumpf*	blau(rose) , blau(rose):-true

Eine Klausel hat den Wahrheitswert true, wenn es keinen Rumpf gibt oder durch Ersetzen von Paramtern durch Atome alle Prädikate im Rumpf den Wert true erhalten. Eine Klausel ohne Rumpf darf als Parameter keine Variablen enthalten!

Beispiele:

schlau(peter).	*immer wahr*
oma(K,O):- kind(K,V,_) , kind(V,_,O).	*oma ist wahr, wenn K einen Vater V hat und dieser Kind von O ist*
oma(K,O):- kind(K,_,M) , kind(M,_,O).	*oma ist wahr, wenn K eine Mutter M hat und diese Kind von O ist*

Funktionsweise:

Ein Prolog-Programm wird gestartet, indem man ein Prädikat aufruft. Man kann dabei Variable durch konkrete Werte ersetzen. Werden alle Variablen ersetzt, so erhält man als Rückmeldung, ob das Prädikat eine wahre Aussage wurde. Andernfalls versucht der Rechner, selbstständig Ersetzungen für die Variablen zu finden, so dass eine wahre Aussage entsteht. Dies geschieht, indem der Rechner Prädikate geeignet durchläuft, bis sich der Wahrheitswert klären lässt. In diesem Fall werden die entsprechenden Ersetzungen für die Anfangsparameter ausgegeben.

Beispiel:

erg(X,Y,E):- zahl(X),zahl(Y),E is X+Y.	*X,Y sind Zahlen und E das Ergebnis ihrer Addition, dabei ist zahl() ein Prädikat, das vorher definiert wurde.*
erg(2,4,6)	*Ausgabe: true*
erg(2,Y,6).	*Ausgabe: Y=4*

Theoretisch werden die Prädikate gleichzeitig untersucht, praktisch im Programmtext von oben nach unten, die Klauseln von links nach rechts. Welche Prädikate der Rechner bei der Lösung verwendet und in welcher Reihenfolge, wird durch ihre Reihenfolge im Programmtext nur beeinflusst, aber nicht festgelegt wie z.B. in Java.
Prolog ermöglicht zudem, während des Programmablaufs selbstständig neue Klauseln zu erzeugen, die in den Ablauf eingreifen.

Funktionen/Operatoren (Auswahl)**:**

Ablaufsteuerung:

fail	*Klausel ist immer falsch*
true	*Anfang der Klausel ist immer wahr*
A,!,B	*nach A wird B getestet, aber bei Rekursion nicht A nach B (Ventilfunktion)*
assert(T)	*Erzeugen einer neuen Klausel T am Ende des Programmtextes*

asserta(T) *Erzeugen einer neuen Klausel T am Anfang des Programmtextes*
clause(K,R) *Testet, ob ein Prädikat mit Kopf K und Rumpf R existiert*

Vergleiche:

A = B *A wird gleichgesetzt mit B*
A == B *A gleich B in allen Eigenschaften*
A \== B *A ungleich B*
A =:= B *Zahlenwerte von A und B gleich*
A =\= B *Zahlenwerte von A und B verschieden*
A < B *A kleiner B*
A =< B *A kleiner oder gleich B*
A > B *A größer B*
A >= B *A größer oder gleich B*

Logik:

A und B A , B.
A oder B *zwei Klauseln:* A.
B.
nicht A not(A) *Umkehrung des Wahrheitswertes*

Arithmetik:

A is Term Term *wird berechnet und der Wert A zugewiesen*

Verknüpfungen: A + B , A - B , A * B , A / B , A // B *(ganzzahlige Division)* , A mod B *(Divisionsrest)*
Tests: integer(A), float(A)
Funktionen: min, max, abs, sign, exp, log, sqrt, sin, cos, tan, asin, acos, atan, sinh, cosh, tanh
/\ *(bitweise and)*, \/ *(bitweise or)*, \ (bitweise not), xor

Zeichenketten (Strings):

string_concat(A,B,C) *Zusammenfügen von String A und B zu C*
string_length(A,B) *Zeichenzahl von String A in B*
string_to_list(A,B) *Umwandeln von String A in Zeichenliste B*
sub_string(A,P,L,R,C) *Teilstring C von String A, beginnend an Position P mit Länge L, Reststring R*
text_to_string(A,B) *Text A umwandeln in String B*
string_codes(A,B) *Umwandeln von String A in Liste der ASCII-Kodes B*
string_chars(A,B) *Umwandeln von String A in Liste der Buchstaben B*
get_string_code(P,A,B) *ASCII-Kode B von Zeichen an der Position P im String A*

Listen:

[A,B,C,...] *Liste mit den Elementen A,B,C,...*
[A|R] *A steht für das erste Listenelement, R für die Restliste*
[A,B|R] *A,B stehen für die ersten beiden Listenelemente, R für die Restliste*
append(A,B,E) *Vereinigen der Listen A und B zur Liste E*
select(A,B,R) *Suchen von Element A in Liste B, R nachfolgende Restliste nach A*

Ein-,Ausgabe:

write(A) *Ausgabe von A*
nl *neue Zeile*
read(A) *Einlesen in A in Klauselform*
readln(A) *Einlesen in A in Listenform*

Windows-Anbindung:

Windows arbeitet prozedural, die Abläufe sind durch die Reihenfolge in den zugehörigen Programmtexten vorgegeben. Dieses Konzept verträgt sich nicht mit Prolog, so dass dort eine Schnittstelle geschaffen werden musste, die beide Umgebungen miteinander verbindet. Dazu werden Windowsaktionen in dem Rumpf einer Klausel eingebettet, die dann von links nach rechts abgearbeitet werden. Das Auslösen einer Aktion ge-

schieht durch das Prädikat send, das eine passende „Botschaft“ an Windows sendet. Windows organisiert dann den Ablauf der Aktion unabhängig von Prolog.

new(K,dialog(N))	*Es wird unter Windows ein neues Dialogfenster angelegt mit dem Titel N, K ist eine von Windows benötigte Kennung des Fensters für weitere Aktionen und darf damit von Prolog aus nicht verändert werden.*
send(K,append,El)	*Botschaft an das Windows-Fenster mit der Kennung E, das Element El einzurichten (vgl. unten)*
send(K,open)	*Öffnen, Anzeigen des Fensters K*
send(K?frame,free)	*Schließen des Fensters K, Einstellungen bewahrt*
send(K,close)	*Schließen des Fensters K*

send-Elemente EL:

new(K,WE)	*neues Eingabeelement WE mit der Kennung K*
text(T)	*Ausgabe von Text T*
text_item(V)	*Eingabefeld mit einleitendem Text V*
text_item(V,T)	*Eingabefeld mit einleitendem Text V, Ausgabe/Vorgabe von Text T*
button(T,message(..))	*Button mit der Aufschrift T, bei Betätigen löst message eine Reaktion aus*

message(K?frame,free) *Schließen des Fensters K (vgl. send)*
message(@prolog,P,Werteliste)
Aufruf des Prädikats P mit Ersetzen der Parameter durch aufgelistete Werte:
K?selection *(Eintrag in Fenster mit Kennung K)*

Hinweis: Welcher Parameter in P durch einen Wert der Werteliste ersetzt wird, entscheidet die Reihenfolge in der Parameter- und der Werteliste, nicht der Name!