

Logische Programmierung und deduktive Datenbanken — Blatt 3: Listen in Prolog —

Aufgabe 3

8 Punkte

Schreiben Sie vier Prädikate zur Listenverarbeitung in Prolog:

- Definieren Sie ein Prädikat `last_element/2`, das das letzte Element einer Liste liefert. Z.B. soll `last_element([1,2,3], 3)` gelten. Der Aufruf `last_element([], X)` soll dagegen fehlschlagen.
- Definieren Sie ein Prädikat `min/2`, das das minimale Element aus einer Liste von Integers liefert. Z.B. soll `min([3,2,5], 2)` gelten. Der Aufruf `min([], X)` soll fehlschlagen. Sie können sich darauf verlassen, daß das Prädikat mit einer Liste von Integers aufgerufen wird, nicht mit irgendwelchen anderen Datentypen. Falls Sie aber testen wollen, ob ein Listenelement ein Integer ist, so geht das mit dem eingebauten Prädikat `integer/1`. Falls Sie allgemein auf Zahlen testen wollen, können Sie das Prädikat `number/1` verwenden.
- Schreiben Sie ein Prädikat `ncopies(C,N,L)`, das L an eine Liste bestehend aus N Elementen C bindet. Z.B. soll `ncopies(a,3,[a,a,a])` gelten.
- Definieren Sie ein Prädikat `like/2`, das dem SQL-Prädikat zum Stringvergleich mit den Wildcards “_” und “%” entspricht. Um Schwierigkeiten mit der Bedeutung von “%” als Kommentar-Kennung zu vermeiden, ersetzen Sie es bitte durch “*”. Z.B. sollen folgende Aufrufe erfolgreich sein: `like("abc","a_c")` und `like("abc","a*")`. (Zur Erinnerung: “_” passt auf ein einzelnes beliebiges Zeichen und “*” passt auf beliebig viele beliebige Zeichen, inklusive der leeren Zeichenkette). In klassischen Prologs (wie z.B. `Gnu-Prolog`) werden Strings (z.B. “abc”) als Listen von Integers (ASCII-Codes) behandelt. Die ASCII-Codes von Zeichen können Sie als `0’x` schreiben. Wenn Sie SWI-Prolog verwenden, sollten Sie die Eingaben wohl zuerst per `string_to_list/2` in diese Darstellung überführen.

Präsenzaufgabe

Machen Sie einen Vorschlag, wie Attributnamen in eine Prolog/Datalog-Variante integriert werden könnten, oder wie man vollständig auf den Tupelkalkül umstellen könnte. Normalerweise werden in Prolog Argumente ja durch Position identifiziert. Wenn man aber auf Datenbank-Relationen zugreift (mit eventuell sehr vielen Spalten), möchte man die Argumente durch Namen identifizieren können, und keine Platzhalter (“_”) für nicht benötigte Spalten angeben müssen. Illustrieren Sie Ihren Vorschlag anhand einer der Aufgaben von Blatt 1.