

Logica Computatională Laboratorul numărul 2

Noțiuni generale

Un program Prolog este alcătuit din **predicate**. Fiecare predicat este definit de *numele* său și de un număr numit *aritatea* sa. Aritatea reprezintă un număr fix de argumente ale predicatului. Două predicate care au același nume, dar un număr diferit de argumente se consideră că sunt predicate diferite. Fiecare predicat dintr-un program este definit de existența uneia sau mai multor **clauze**.

Exemplu:

femeie(elena).

parinte(andrei, cristi).

parinte(andrei, elena).

În exemplul de mai sus, predicatul *parinte* are două clauze, iar predicatul *femeie* are o singură clauză. O clauză poate fi un **fapt** sau o **regulă**. Un fapt are forma

nume_relatie(arg1, ..., argN).

iar o regulă

nume_rel(arg1, ..., argN) :- rel1(...), ..., relM(...).

unde arg1, ..., argN reprezintă argumentele. Ca argument, poate fi folosit orice **termen** al limbajului Prolog. Termenii de bază sunt:

- Întregii – numere pozitive sau negative
- Atomii – text care începe cu literă mică
- Variabilele – încep cu literă mare sau *underline* (_)
- Structuri compuse

Întregii în Prolog, sunt reprezentați de cifre (0 - 9).

Atomii sunt alcătuiți de obicei din litere și cifre, având primul caracter o literă mică.

Exemplu de atomi:

salut

douaCuvinteAlaturate

un_atom

a2

Următoarele elemente nu reprezintă atomi:

nu-este-atom

5nu

_faraunderline

Litramare

Dacă avem caractere introduse între apostrofură formăm atomi:

'acesta-este-atom'

'inca un atom'

'Atom'

Variabilele sunt asemănătoare atomilor, cu excepția că ele încep cu literă mare sau cu underline (_).

Exemple:

Variabila

_variabila

Alta_vaRiabila2

În sfârșit, *structurile compuse* sunt de forma următoare:

persoana(maria, inginer, adresa("Calea Bucuresti", "Bloc A10", "Ap 11")).

Variabila anonimă

Variabilele se notează, după cum afirmam mai sus, cu șiruri de caractere care încep cu literă mare sau cu șiruri de caractere ce încep cu ' _ '. Acestea din urmă se numesc variabile anonime. Care este rolul lor? Vom observa, urmărind exemplul de mai jos.

Avem predicatul: *parinte(numeCopil, numePărinte), barbat(numeBărbat), femeie(numeFemeie)* și vrem să afișăm pentru o persoană toți copiii și toți nepoții. Pentru aceasta, definim predicatul *copii(+numePărinte, -numeCopil)* și *nepoti(+numeBunic, -numeNepot)*.

Observație: Notația *nume_predicat(+term1, -term2)* reprezintă faptul că termenul *term1* este termen de intrare, iar *term2* este termen de ieșire.

Presupunem că avem introduse faptele *parinte(...), barbat(...)* și *femeie(...)*. Scriem în continuare regulile de calcul pentru afișarea copiilor și a nepoților.

copii(Tata) :- parinte(Copil, Tata), tab(2), write_ln(Copil), fail.

nepoti(Bunic) :- parinte(Tata, Bunic), parinte(Nepot, Tata), tab(2), write_ln(Nepot), fail.

Putem testa aceste predicate și observăm că afișează fiecare copiii, respectiv nepoții pentru persoana dată de noi. Spuneam însă că vrem să afișăm pentru o persoană atât copiii, cât și nepoții în cadrul aceleiași interogări. Pentru aceasta, construim un nou predicat:

descendenti(X) :- write_ln('Copiii:'), copii(X), write_ln('Nepotii:'), nepoti(X).

Ce observăm însă? Acesta nu afișează decât copiii! Am văzut că, individual, fiecare regulă funcționează: atunci care este problema? În rândurile următoare o să vedem de ce. Ei bine, într-adevăr funcționează separat dar pentru fiecare, după ce afișează elementele cerute, ne răspunde cu *No* datorită predicatului implicit *fail*. Pentru a putea folosi

predicatul *copii/1* în conjuncție cu alte predicate, nu trebuie decât să mai adăugăm o clauză *copii/1* care răspunde întotdeauna cu *Yes*:

```
copii(Tata) :- parinte(Copil, Tata), tab(2), write_ln(Copil), fail.  
copii(OriceVariabila).
```

În acest caz, atunci când prima clauză va întoarce *No* (pentru că nu mai găsește alți copii), sistemul va încerca cea de a doua clauză *copii/1*. Devreme ce argumentul celei de a doua clauze este o variabilă, sistemul va întoarce întotdeauna răspunsul *Yes*.

În această a doua clauză, pe noi nu ne interesează ce variabilă folosim. Pentru asemenea situații, folosim *variabila anonimă* care este notată prin underline () sau printr-un șir de caractere care începe cu underline:

```
copii(_).
```

Astfel, obținem următorul program:

```
copii(Tata) :- parinte(Copil, Tata), tab(2), write_ln(Copil), fail.  
copii(_).
```

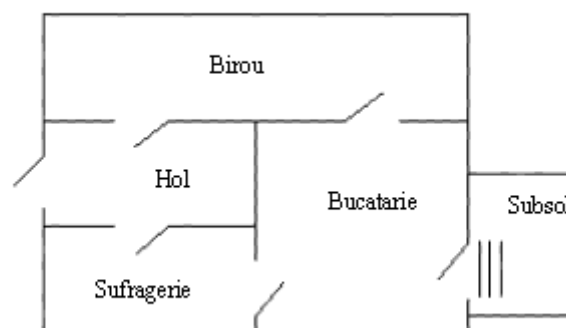
```
nepoti(Bunic) :- parinte(Tata, Bunic), parinte(Nepot, Tata), tab(2), write_ln(Nepot), fail.  
nepoti(_).
```

```
descendenti(X) :- write_ln('Copiii:'), copii(X), write_ln('Nepotii:'), nepoti(X).
```

Observație: *nume_predicat/n* reprezintă faptul că *nume_predicat* este un predicat de aritate *n*.

Exercițiu

Presupunem că avem următoarea schemă a unei case:



În fiecare cameră se găsesc următoarele lucruri:

- în birou: masă, computer, scaun;
- în hol: cuier;
- în sufragerie: televizor, bibliotecă, fotolii, canapea;

- în bucătărie: aragaz, frigider, masa, scaune, biscuiți, mere;
- la subsol: mașina de spălat, rufe murdare.

Definiți legăturile dintre camere (eventual printr-un predicat numit *ușă*), locurile unde sunt situate diferitele obiecte, locul unde se află subiectul.

Dacă apelez un predicat numit *start* care să primească un argument ce reprezintă o camera în care este plasat subiectul, sistemul să spună care este camera în care este situat, care sunt lucrurile pe care le poate vedea în camera în care se află și care sunt camerele în care poate merge.

Exemplu:

?- start(bucatarie).

Esti in bucatarie

Poti vedea:

aragaz
frigider
masa
scaune
biscuiti
mere

Poti merge in:

birou
subsol
sufagerie

Operații matematice

Pentru evaluarea unei expresii aritmetice, în Prolog folosim predicatul predefinit *is*. Sintaxa acestui operator este următoarea:

X is <expresie aritmetică>.

Variabila X va lua valoarea expresiei aritmetice. Important este faptul că variabila X nu trebuie să aibă anterior o valoare asignată. Expresia aritmetică arată ca o expresie aritmetică din orice alt limbaj de programare.

Exemplu:

?- X is $2 + 2$.

$X = 4$

?- X is $3 * (4 + 1)$.

$X = 15$

În afară de predicatul *is*, putem folosi operatori pentru compararea numerelor: $<$, $>$, $=<$, $>=$, $==$, $=\backslash=$. Ultimii doi operatori sunt pentru a verifica egalitatea dintre două numere, respectiv pentru a verifica dacă două numere sunt diferite.

Exemplu:

?- X is $2 + 2$, $X > 3$.

X = 4

Yes

Operatorii pot fi folosiți și în cadrul regulilor:

suma(N1, N2, S) :- S is N1 + N2.

Important

Trebuie să înțelegem faptul că scrierea "2+3" în Prolog nu reprezintă o instrucțiune care păstrează rezultatul acestei adunări; ea reprezintă mai degrabă "adunarea lui 2 cu 3". Astfel, "2+3" este un termen diferit de "4+1" și bineînțeles diferit de "5*1". Pentru a înțelege aceasta, iată următorul exemplu:

numar(3).

numar(4).

numar(5).

Presupunem că avem aceste fapte într-un program Prolog. Dacă adresăm interogarea:

? – numar(2 + 1).

ni se va răspunde cu *No* pentru că termenul $2 + 1$ nu poate fi *unificat* cu vreunul din termenii din baza de cunoștințe. La o interogare de forma :

? – X is 2 + 1, numar(X).

vom primi următorul răspuns:

X = 3

Yes

În acest caz, s-a realizat întâi calculul, (datorită predicatului *is*) iar apoi a fost realizată interogarea numar(3).

Interogări

Introduceți următoarele interogări și asigurați-vă că înțelegeți de ce primiți rezultatele oferite de sistem în fiecare caz:

- N is 1+1.
- N is 1+1, P is N*2, Q is P+Q.
- N is X+1.
- I is I+1.
- I is 6, I is I+1.
- I is 6, J is I+1.

Exerciții

1. Definiți un predicat care să adauge o unitate la un număr dat.
2. Patrutul unui număr.
3. Maximul a două numere.
4. Maximul a trei numere.
5. Valoarea absolută a unui număr.

6. Afisati daca un numar este par sau impar.
7. Afisati daca un numar este pozitiv sau negativ.
8. Calculati valoarea functiei $f_1(x) = 3x^2 - 4$ intr-un punct dat.
9. $f_2(x) = 4x^2 + 2x - 3$
10. $f_3(x)$ care este $x - 1$ dacă $x > 0$ și 0 altfel.
11. $f_4(x) = |x| - 4$.
12. $f_5(x) = 2 * |x - 3|$.
13. $f_6(x)$ care este $x^2 + 2$ dacă $x > 10$ și $|x| - 4$ altfel.
14. $f_7(x)$ care este $2x^2 + 1$ dacă x este par și $x - 7$ altfel.
15. Stim ca studentii de mai jos au luat urmatoarele note la examen:

Ion	4
Marius	8
Adriana	3
Carmen	10

La o interogare asupra starii unui student, sistemul sa ne raspunda in functie de nota daca este promovat (> 4) sau nepromovat.