Programare Logică – Laboratorul 10

Eficienţă în Prolog; Probleme cu stringuri

Isabela Drămnesc

December 12, 2012

1 Noţiuni

- Gândire declarativă;
- Gândire procedurală;
- Recursivitate finală;
- Eficienţă;
- Predicate pentru stringuri;
- I/O operaţii cu fişiere;
- Indexare

2 Exemple–Recursivitate finală

Câteva exemple începând cu pagina 47 din curs

Alte exemple:

```
test1:-write(hello),nl,test1. % predicatul este final recursiv,
                              % DAR poate rula la infinit.

test2:-test2, write(hello),nl. % predicatul nu este final recursiv
                               % deoarece apelul recursiv nu este ultimul

test3:-write(hello),nl,test3.  % predicatul nu este final recursiv
test3:-write(goodbye).         % deoarece are o varianta care nu este incercata

test4:-a, write(hello),nl,test4.  % predicatul nu este final recursiv
                          % deoarece predicatul g are variante neincercate
a:-write(starting).
a:-write(beginning).
```

Concluzii:

Un predicat este final recursiv dacă:

- apelul recursiv este ultima condiţie din clauză;
- nu are clauze neîncercate;
- nu exista alte condiţii dinaintea apelului recursiv neîncercate.

În cazul în care avem multe apeluri recursive programele noastre consumă o memorie mare, deci programele noastre trebuie să fie eficiente. Pentru a fi eficiente trebuie să scriem predicatele în Prolog final recursive.

3 Probleme

1. Pentru exemplele din curs test3 şi test4 găsiţi şi o altă variantă final recursivă.

2. Daţi exemple de cel puţin 2 predicate care nu sunt eficiente în Prolog şi cel puţin 2 predicate care sunt eficiente în Prolog. Explicaţi!

3. Scrieţi un program în Prolog şi folosiţi indexarea.

4. Să ne reamintim puţin de stringuri:

Exemple de predicate predefinite în Prolog:

```
?- string_to_list(String, [99,101,118,97]).
String = "ceva".

% predicatul care realizeaza concatenarea a doua stringuri
?- string_concat(timi,X,timisoara).
X = "soara".

?- string_concat(Y,X,timisoara).
Y = "",
X = "timisoara" ;
Y = "t",
X = "imisoara" ;
Y = "ti",
X = "misoara" ;
Y = "tim",
X = "isoara" ;
Y = "timi",
X = "soara" ;
Y = "timis",
X = "oara" ;
Y = "timiso",
X = "ara" ;
Y = "timisoa",
X = "ra" ;
Y = "timisoar",
```

```
X = "a" ;
Y = "timisoara",
X = "".


% predicatul care afiseaza lungimea unui string
?- string_length('lungime string', M).
M = 14.

?- string_length('lala', M).
M = 4.


% predicatul care afiseaza subsirul unui sir de caractere
% de la o pozitie pana la alta pozitie indicata
?- sub_string('Azi e o zi tare frumoasa',0,10,CatePozitiiMaiSunt,Subsirul).
CatePozitiiMaiSunt = 14,
Subsirul = "Azi e o zi".

% predicatul care returneaza ora curenta si data curenta
?- get_time(Timp), convert_time(Timp, Data), nl, write('Data de azi este '), write(Data).
Data de azi este Mon Dec 07 15:01:55 2009
Timp = 1.26019e+009,
Data = "Mon Dec 07 15:01:55 2009".

?- get_time(Timp), convert_time(Timp, An, Luna, Zi, Ora, Minute, Secunde, Milisec).
Timp = 1.26019e+009,
An = 2009,
Luna = 12,
Zi = 7,
Ora = 15,
Minute = 4,
Secunde = 1,
Milisec = 385.
```

5. Citiţi de la tastatura N numere până la introducerea unui caracter.

6. Generaţi M numere aleatoare pe care le scrieţi într-un fişier. Realizaţi asupra acestora cel puţin 10 operaţii! (de ex: sortare, stergerea primului element, etc.), fiecare operaţie asupra numerelor din fişier va fi afişată într-un alt fişier.

Dacă faceţi 10 operaţii aveţi:

- Un fişier care conţine datele de intrare (numerele generate aleator)
- 10 fişiere: sort.txt, stergere.txt,....etc.

Creaţi programe cât mai eficiente!

Testaţi timpul de execuţie pentru fiecare predicat creat.

4 Tema:

Nu uitaţi că la ultimul laborator pe lângă proba practică trebuie să prezentaţi toate problemele de la temă (cele din fişierele de la temă şi cele din laboratoare)!!!

Studiu.