1. DEOSEBIRI ÎNTRE PROGRAMAREA LOGICĂ ŞI PROGRAMAREA CLASICĂ

Un program este alcătuit din două elemente, ***logica*** şi ***controlul*** ([9], [10]). Prin termenul "logică" se desemnează toate noţiunile care stabilesc **CE** face un program, în timp ce termenul "control" semnifică toate noţiunile sintactice care stabilesc **CUM** o face (de exemplu algoritmul care rezolvă o problemă). Un program scris în PASCAL sau vreun alt limbaj de programare tradiţional constă din ***instrucţiuni*** care descriu acţiunile ce trebuie executate pas cu pas de către calculator pentru ca programul să producă rezultatul dorit. Dimpotrivă un program în PROLOG este o bază de natură logică, în care programatorul defineşte ***obiecte*** şi ***relaţii*** între aceste obiecte (fie relaţii care există direct între obiecte, fie regulile după care se pot deduce alte relaţii sau proprietăţi ale acestora).

Obiectele sunt reprezentate în PROLOG prin ***nume simbolice***. Relaţiile existente între obiecte se definesc cu ajutorul clauzelor. Există două feluri de ***clauze***: ***fapte*** şi ***reguli***. Structura sintactică a unui fapt este următoarea:

$$nume(arg_1, arg_2, \ldots, arg_n).$$

unde:

- ***nume*** este un nume de predicat, adică o succesiune de caractere (alfabetice, cifre sau liniuţa de subliniere) având proprietatea că primul caracter este o literă mică a alfabetului
- $arg_1, arg_2, \ldots, arg_n$ se numesc argumentele predicatului, iar din punct de vedere sintactic pot fi nume de obiecte sau nume de variabilă (un nume de variabilă este o succesiune de caractere astfel încât primul caracter este o literă mare din alfabet).

O regulă este o construcţie sintactică de forma:

$$c_{n+1} :- c_1, c_2, \ldots, c_n.$$

unde:

- entităţile $c_1, c_2, \ldots, c_n, c_{n+1}$ sunt de forma $nume(arg_1, arg_2, \ldots, arg_k)$, **nume**, $arg_1, arg_2, \ldots, arg_k$ având aceeaşi semnificaţie ca în cazul unei fapte.

- c_{n+1} se numeşte ***capul reguli***.
- $c_1, c_2, \ldots, c_n$ formează ***corpul reguli***.

Clauza codifică următorul enunț:

"Dacă c_1 şi c_2 şi … c_n sunt adevărate, atunci c_{n+1} este adevărată."

Observăm că orice faptă şi orice regulă se încheie cu caracterul ".", iar "," prezentă în corpul regulii suplineşte operatorul "şi". Dacă în loc de "," utilizăm ";" atunci în locul operatorului "şi" se consideră operatorul "sau". Prin urmare clauza ***a :- b; c.*** codifică enunțul "dacă b sau c, atunci a".

În general, într-un limbaj de programare tradițional, un program exprimă o funcție de la intrarea la ieşirea programului, în timp ce un program într-un limbaj de programare logică exprimă o ***relație între date***. Întrucât relațiile sunt mai generale decât funcțiile, programarea logică are posibilități mai mari decât are programarea tradițională.

Să luăm drept exemplu un program care citeşte două numere reale şi îl afişează pe cel mai mare dintre ele. Pentru a face diferența dintre programarea tradițională şi programarea logică mai explicită, vom da mai întâi programul în PASCAL, iar apoi acelaşi program în PROLOG:

Program în PASCAL:

```
var x,y: real;
begin
        write('numar1 = '); readln(x);
      write('numar2 = '); readln(y);
      if (x < y) then writeln(x)
            else writeln(y);
end.
```

Program în PROLOG:

```
predicates
        program
        mai_mare(real,real,real)
clauses
```

```
        program :- write("n1="), readreal(X), write("n2="), readreal(Y),
mai_mare(X,Y,Z),
                write(Z), nl.
        mai_mare(X,X,X).
        mai_mare(X,Y,Y):-X<Y.
        mai_mare(X,Y,X):-Y<X.
goal
        program
```

Secțiunea var (din programul PASCAL) şi secțiunea predicates (din programul PROLOG) au un caracter declarativ: în secțiunea var se declară variabilele reale x şi y, iar în secțiunea predicates se declară predicatele program (fără argumente) şi mai_mare (cu trei argumente reale). Programul în PASCAL este doar un şir de instrucțiuni. Aceste instrucțiuni, care sunt executate în ordinea indicată de program, constituie "controlul". "Elementul logic" în programul din PASCAL se află în relația ">". Dimpotrivă, programul în PROLOG este o colecție de clauze care descriu complet relația de ordine totală a două numere reale, în speță predicatul "mai_mare". Această colecție de clauze exprimă logica programului, care are şi rolul dominant într-un program PROLOG, în timp ce controlul se află în ordinea în care aranjăm şi definim predicatele. De asemenea lansarea în execuție a unui program PROLOG, nu se face definind un punct de început după care instrucțiunile să fie preluate secvențial, aşa cum se întâmplă în limbajele tradiționale de programare. Execuția unui program cere definirea unui ***scop*** (în secțiunea goal), care trebuie verificat pe baza elementelor din program (în exemplul nostru scopul este dat de predicatul program).

O altă diferență între programarea logică şi cea clasică constă în semnificația ***variabilelor***. Programarea în PROLOG nu utilizează conceptul de ***atribuire***. ***Actualizarea*** variabilelor prin obiecte PROLOG date se realizează prin ***unificare***. În general, cu unificarea se analizează dacă două predicate pot fi identice. Dacă ele nu pot fi identice, procedura de unificare eşuează. Dacă ele pot fi identice, unificarea reuşeşte, iar rezultatul unificării este actualizarea variabilelor celor două predicate cu anumite de valori astfel încât cele două predicate să coincidă ("match"). Pentru actualizare se foloseşte adesea termenul ***legare*** ("binding") sau ***instanțire***. O variabilă prezentă într-o clauză are efect local clauzei respective. Variabilei i se leagă o valoare, iar ieşirea din

clauză determină dezlegarea ei de valoare. Există o variabilă care joacă un rol special în PROLOG: ea se numeşte ***variabilă anonimă*** şi se reprezintă prin "_". Prezența variabilei anonime pe locul unui argument precizează că nu interesează valoarea legată de argument ci numai existența unei asemenea valori.