5. PREDICATE DE INTERACȚIUNE

PROLOGUL ca limbaj de programare rafinat, oferă posibilitatea interacțiunii unui program cu echipamentul periferic al calculatorului, cu ecranul şi cu utilizatorul. Aceasta se realizează cu ajutorul unor predicate predefinite. Există trei predicate predefinite de ieşire: ***write***, ***writedevice*** şi ***writef***. Forma generală a predicatului write este:

$$write(arg_1, arg_2, \ldots, arg_n)$$

unde arg_1, arg_2,…, arg_n sunt obiecte elementare sau variabile. În cazul în care arg_k este o variabilă, ea trebuie să fie legată în momentul execuției predicatului write. Predicatul write reuşeşte întotdeauna şi are ca efect afişarea pe ecran a valorilor legate de argumentele sale. Pentru trecerea la "o linie nouă" se poate utiliza predicatul ***nl*** (new line), sau secvența de caractere ***"\n"***. Exemplificăm utilizarea acestor predicate cu ajutorul următorului program:

```
domains
        nume=string
        firma=symbol
predicates
        angajat(nume, firma)
        salariu(nume, real)
        afis
clauses
        angajat("Mihai Popescu",bcr).
        angajat("Petre Georgescu",bcr).
        angajat("Maria Ionescu",banc_post).
        salariu("Mihai Popescu",10000000).
        salariu("Petre Georgescu",5000000).
```

```
salariu("Maria Ionescu", 70000000).
afis :-angajat(X,Y), salariu(X,Z),write(X, " lucreaza la ", Y),
        write(" si are salariul ", Z, "\n"), fail.
afis.
```

Predicatul fail a fost utilizat în prima clauză afis pentru a forța backtrackingul. Aceasta este necesar deoarece predicatul write nu se resatisface. A doua clauză afis a fost adăugată pentru a evita răspunsul "no" (prima clauză afis esuează întotdeauna datorită prezenței predicatului fail). Redăm în continuare răspunsul sistemului la interogarea afis.

```
Goal : afis
Mihai Popescu lucreaza la bcr şi are salariul 10000000.
Petre Georgescu lucreaza la bcr şi are salariul 5000000.
Maria Ionescu lucreaza la banc_post şi are salariul 7000000.
Yes
```

Predicatul ***writef*** se utilizează în cazul transmiterii cu format de ieşire a datelor. Forma sa este:

$$writef(şir\_format, arg_1, arg_2, \ldots, arg_n)$$

unde şir_format este un şir de caractere care indică formatul utilizat pentru afişarea pe ecran a valorilor legate de $arg_1, arg_2, \ldots, arg_n$. În general şirul şir_format conține:

- caractere care se copiază ca atare pe ecran
- grupuri de specificatori de forma % -<dimens>.<prec>tip (pentru fiecare argument din listă trebuie să existe un specificator)

Semnificația caracterelor dintr-un specificator este următoarea:

- % marchează începutul specificatorului
- - determină alinierea la stânga şi completarea cu spații la dreapta. Prezența acestui caracter este opțională. Implicit: aliniere la dreapta şi completare la stânga cu spații sau cu 0
- dimens specifică dimensiunea minimă a câmpului afişat; nu se efectuează trunchierea valorii afişate.
- prec este opțional şi arată precizia numărulzi în virgulă mobilă sau lungimea maximă a numărului de caractere care se tipăresc în cazul obiectului de tip string.

- tip (opțional) specifică forma în care este afişat argumentul în virgulă mobilă; poate avea valorile f, e, g cu următoarele semnificații:
  - f indică virgula mobilă
  - e indică scriere exponențială
  - g indică scrierea cu format scurt

Considerăm programul:

```
predicates
scrie
clauses
        scrie:-writef("**************%7.7*************\n","exemplu"),
                writef("%9.8 %5.3f %-20","12.3456=",12.3456,"(virgula mobila)"),nl,
          writef("%9.8 %5.3e %-20","12.3456=",12.3456,"(exponentiala)"),nl,
          writef("%9.8 %5.3g %-20","12.3456=",12.3456,"(format scurt)"),nl,
          writef("**************=%7.7=*************\n","sfarsit").
```

Interogarea scrie are următorul efect:

```
Goal: scrie
**************exemplu*************
 12.3456= 12.346 (virgula mobila)
 12.3456= 1.235E+01 (exponentiala)
 12.3456= 12.350 (format scurt)
**************=sfarsit=*************
Yes.
```

Predicatul ***writedevice*** este utilizat pentru a redirecta afişarea rezultatelor către imprimantă sau către un fişier. De exemplu, writedevice(printer) determină afişarea la imprimantă a rezultatelor, iar writedevice(screen) determină afişarea rezultatelor pe ecran.

Există patru predicate predefinite de intrare: ***readln***, ***readchar***, ***readint***, ***readreal***.

Predicatul readln permite citirea unui obiect de tip symbol sau string, şi are forma

readln(X)

unde X este o variabilă, care în urma introducerii textului (urmat de ENTER) devine legată la valoarea obținută în urma citirii. Predicatul readln se execută cu ecou pe ecran.

Predicatul readchar permite citire unui singur caracter, şi are forma

readchar(X)

unde X este o variabilă care în urma execuției va fi legată la caracterul introdus. Acest predicat se execută fără ecou pe ecran.

Predicatul readint are forma

readint(X)

şi permite citirea unui număr întreg care va fi legat la variabila X.

Predicatul readreal are forma

readreal(X)

şi permite citirea unui număr real care va fi legat la variabila X.

Să considerăm, pentru exemplificare următorul program:

```
predicates
        cont(integer, symbol)
        afis_simbol
clauses
        cont(506, obligatiuni).
        cont(512, conturi_curente_la_banci).
        cont(531, casa).
        afis_simbol:-write("Denumirea contului= "), readln(X), cont(Y,X),
                        write("Simbolul acestui cont este: ",Y),nl.
```

Iată şi o interogare:

```
Goal: afis_simbol
Denumirea contului = conturi_curente_la_banci

Simbolul acestui cont este: 512
Yes
```

Prezentăm în continuare predicatele predefinite utilizate pentru definirea ferestrelor pe ecran şi dirijarea fluxului de informaţii între acest ferestre. Pentru a defini o fereastră se foloseşte se foloseşte predicatul ***makewindow*** care are următoarea formă:

makewindow(Nr_f, Atr_f, Atr_m, Header, L, C, H, W)

cu următoarea semnificaţie a argumentelor:

- Nr_f este numărul ataşat ferestrei. Este un număr întreg cu ajutorul căruia se identifică fereastra.
- Atr_f codifică atibutele video ale ferestrei
- Atr_m codifică atributele video pentru marginile ferestrei.
- Header defineşte headerul (titlul) ferestrei, i.e un text care se afişeză pe margine superioară a ferestrei.
- L, C definesc coordonatele colţului din stânga sus al ferestrei (L = numarul liniei, C = numărul coloanei)
- H, W definesc dimensiunile ferestrei (H = inalţimea, W = lungimea).

Valorile L, C, H, W trebuie să se încadreze în domeniul oferit de modul text curent ($L + H \leq 25$, $C + W \leq 40$ sau $C + W \leq 80$). Dimensiunea minimă a ferestrei este de o coloană şi o linie. Ferestrele pot fi plasate oriunde pe ecran şi se pot suprapune. La fiecare scriere în fereastră cursorul este avansat conform scrierii: calculatorul memorează de fiecare dată ultima poziţie pe care s-a scris, astfel ă la fiecare revenire în fereastră se continuă scrierea din locul în care a rămas cursorul. Atr_f controlează atributul unei celule din fereastră (toate celulele unei ferestre au acelaşi atribut), i.e controlează

- culoarea cu care va apărea caracterul din celulă
- culoare fondului celulei
- clipirea caracterului

Atr_m controlează atributele celulelor de pe marginea ferestrei. Pentru a obţine atributul unei celule se adună la numărul asociat culorii cu care se scrie caracterul numărul asociat culorii fondului înmulţit cu 16. Dacă se doreşte pulsarea imaginii se adaugă 128 la acest atribut. Tabelul de mai jos prezintă numerele asociate culorilor.

Culoare	Număr	Pentru text sau pentru fond
negru	0	ambele

albastru	1	ambele
verde	2	ambele
cyan	3	ambele
roşu	4	ambele
magenta	5	ambele
maron	6	ambele
alb	7	ambele
gri	8	numai pentru text
albastru deschis	9	numai pentru text
verde deschis	10	numai pentru text
cyan deschis	11	numai pentru text
roşu deschis	12	numai pentru text
magenta deschis	13	numai pentru text
galben	14	numai pentru text
alb intens	15	numai pentru text
clipire	128	numai pentru text

Deplasare între ferestre se realizează cu predicatul:

shiftwindow(Nr_f)

în care Nr_f este numărul ferestrei în care urmează să şe facă deplasarea. Un efect asemănător are predicatul:

gotowindow(Nr_f)

dar nu poate fi utilizat pentru ferestre care se suprapun.

Alte predicate predefinite pentru controlul ferestrelor sunt:

removewindow : scoate din uz fereastra activă.

clearwindow : şterge fereastra activă şi mută cursorul în colţul din stânga sus.

cursor(L, C) : poziţionează cursorul în punctul de coordonate C (coloana) şi L (linia) în fereastra activă. Dacă L şi C sunt două variabile libere, atunci ele sunt legate la coordonatele cursorului.

Programul următor reprezintă o demonstrație de utilizare a ferestrelor. Se creează două ferestre cu titlurile “Fereastra 1”, respectiv “Fereastra 2”. Fereastra 2 este fereastră activă. În această fereastră este afişat textul

“Sunt în fereastra 2.

Apăsați o tastă pentru a trece în fereastra 1”

În urma apăsării unei taste se şterge conținutul ferestrei şi se trece în fereastra 1, care devine fereastră activă. În această fereastră este afişat mesajul

“Sunt în fereastra 1.

Continuați? [y|n] ”

Dacă se răspunde cu y se şterge textul din fereastra 1 şi se trece în fereastra 2, după care se reiau paşi de mai sus. Dacă se răspunde cu orice alt caracter execuția programului se încheie.

```
predicates
        ferestre
        comutare21
        comutare12
        comutare
clauses
        ferestre:-makewindow(1,20,4,"Fereastra 1",10,5,10,70),
                  makewindow(2,48,32,"Fereastra 2",15,5,5,70).
    comutare21:-
     write("Sunt in ferestra 2.","\n", "Apasati o tasta pentru a trece la fereastra 1!"),
                  readchar(_),clearwindow,comutare12.

     comutare12:-shiftwindow(1), write("Sunt in fereastra 1","\n"),
                    write("Continuati ? [y|n]"),readchar(X),nl,X='y',clearwindow,
                    shiftwindow(2),comutare21.
     comutare12.
    comutare:-ferestre,comutare21.

Goal: comutare
```