Prolog CheatSheet

Laborator 10

Aflarea tuturor soluțiilor pentru satisfacerea unui scop

findall/3

```
findall(+Template, +Goal, -Bag)
```

Predicatul findall creează o listă de instanțieri ale lui Template care satisfac Goal și apoi unifică rezultatul cu Bag

```
higherThan(Numbers, Element, Result):-
    findall(X, (member(X, Numbers), X > Element), Result).
?- higherThan([1, 2, 7, 9, 11], 5, X).
X = [7, 9, 11]

?- findall([X, SqX], (member(X, [1,2,7,9,15]), X > 5, SqX is X **
     2), Result). % în argumentul Template putem construi
    structuri mai complexe
Result = [[7, 49], [9, 81], [15, 225]].
```

Aflarea tuturor soluțiilor pentru satisfacerea unui scop

bagof/3

```
bagof(+Template, +Goal, -Bag)
```

Predicatul bagof este asemănător cu predicatul findall, cu excepția faptului că predicatul bagof construiește câte o listă separată pentru fiecare instanțiere diferită a variabilelor din Goal (fie că ele sunt numite sau sunt înlocuite cu underscore.

```
are(andrei,laptop,1). are(andrei,pix,5). are(andrei,ghiozdan,2).
are(radu,papagal,1). are(radu,ghiozdan,1). are(radu,laptop,2).
are(ana, telefon, 3). are(ana, masina, 1).

?- findall(X, are(_, X, _), Bag).
Bag = [laptop, pix, ghiozdan, papagal, ghiozdan, laptop, telefon,
     masina]. % laptop și ghiozdan apar de două ori pentru că
    sunt două posibile legări pentru persoană și pentru cantitate

?- bagof(X, are(andrei, X, _), Bag).
Bag = [laptop] ;
Bag = [ghiozdan] ;
Bag = [pix].
% bagof creează câte o soluție pentru fiecare posibilă legare
    pentru cantitate.  Putem aici folosi operatorul existențial ^
?- bagof(X, C^are(andrei, X, C), Bag).
Bag = [laptop, pix, ghiozdan]. % am cerut lui bagof să pună toate
    soluțiile indiferent de legarea lui C în același grup

?- bagof(X, C^are(P, X, C), Bag).
P = ana, Bag = [telefon, masina] ;
P = andrei, Bag = [laptop, pix, ghiozdan] ;
P = radu, Bag = [papagal, ghiozdan, laptop].
```

Dacă aplicăm operatorul existențial pe toate variabilele libere din scop, rezultatul este identic cu cel al lui `findall`.

```
?- bagof(X, X^P^C^are(P, X, C), Bag).
Bag = [laptop, pix, ghiozdan, papagal, ghiozdan, laptop, telefon,
     masina].
```

Aflarea tuturor soluțiilor pentru satisfacerea unui scop

setof/3

```
setof(+Template, +Goal, -Bag)
```

Predicatul setof este asemănător cu bagof, dar sortează rezultatul (și elimină duplicatele) folosind sort/2.

```
?- setof(X, C^are(P, X, C), Bag).
P = ana, Bag = [masina, telefon] ; %se observă sortarea
P = andrei, Bag = [ghiozdan, laptop, pix] ;
P = radu, Bag = [ghiozdan, laptop, papagal].

?- setof(X, P^C^are(P, X, C), Bag).% setof elimină duplicatele
Bag = [ghiozdan, laptop, masina, papagal, pix, telefon].
```

Aflarea tuturor soluțiilor pentru satisfacerea unui scop

forall/2

```
forall(+Cond, +Action)
```

Predicatul forall verifică dacă pentru orice legare din Cond, care reprezintă un domeniu ce conține legări de variabile, se pot îndeplini condițiile din Action.

```
?- forall(member(X,[2, 4, 6]), X mod 2 =:= 0).
true.

?- forall(member(X,[2, 4, 3, 6]), X mod 2 =:= 0).
false.

?- forall(member(X, [6, 12, 18]), (X mod 2 =:= 0, X mod 3 =:= 0))
    .
true.
```