Le langage Prolog
(Demo IV)

Atefeh Farzindar

Table des matières

Demo IV

- Entrées / Sorties
- Les prédicats du 2nd ordre
- setof / bagof/ findall/3

Entrées / Sorties

Le Prolog standard ne connaît que l'ASCII. Les entrées/sorties son primitives.

1. *Lire et écrire des termes*
2. *Lire et écrire des caractères*
3. *Ouvrir et fermer des fichiers*

1. Lire et écrire des termes

- read(X) : Lecture d'un terme

  Le prochain terme `T` est lu est unifié à X.
  - Si X et T peuvent être instanciés, alors read(X) est vrai.
  - Sinon read(X) est évalué à faux.
- write(X) : Écriture du terme X.
- tab(X) : Écriture de N espaces vides.
- nl : Insère une nouvelle ligne.

Exemple

```
?- read(X).                %Lecture d'un Terme
père(françois, isa).

X = père(françois, isa).
```

read renvoie end_of_file en fin de fichier.

```
?- T = père(françois, isa), write(T).
père(françois, isa).
```

Exemple 2:

```
carre :- read(X),  manipule(X).
manipule(stop):- !.
manipule(N):- C is N*N,
write(C),carre.
```

Exécution:

```
| ?- carre.
5. 
25
stop. 
yes
```

2. Lire et écrire des caractères

- get0(X) : Lire un code ASCII et unifier X à ce code.

- get(X) : Même que get0(X) mais saute les caractères non-imprimable blanc (c'est-à-dire un caractère de code ASCII < 32) lu sur le flux d'entrée.

- put(a) : Écrire le caractère associé au code ASCII

Exemple

```
?- get0(X).
:|a □

X = 65.

?- put(65).
a.

?- get0(X).
:|  □          %lire un « blanc »

X = 32.
```

Example : Effacer des espaces vides d'une phrase

```
comprimer:- get0(C),put(C),rest(C).
rest(46):- !.    %46 ='.' point
rest(32):-!,     %32=' '  espace vide
get(C),put(C),rest(C).
rest(Lettre):-comprimer.

| ?- comprimer.
|: Bonjour                !      nous sommes
   étudiants.
Bonjour ! nous sommes étudiants .
```

3. Ouvrir des fichiers

`see('nom1').`

Ouverture en lecture du fichier «**`nom1`**»

Le fichier devient le flux d'entrée courant.

`seeing(F).`

`F` s'unifie au fichier en cours.

`see(user).`

Le flux courant redevient l'utilisateur – le clavier.

`seen.`

Fermeture du fichier ouvert en lecture. L'utilisateur devient le flux courant.

Exemple:calcule le carré d'un chiffre de fichier f1.pl

```
main :-
   see('f1'),       %Entrée: lecture du fichier f1 qui
                      %contient un chiffre N et un point.
   carre,           %calcule le carré de N
   seen.            %fermeture du fichier f1
                    %l'usager devient le flux courant.
carre :- read(N),
         C is N*N, write(C), nl,
         write('le flux courant est le fichier:'),
         seeing(F),
         write(F).
```

4. Ècrire dans les fichiers

`tell(nom2).`

Ouverture en écriture du fichier «**`nom2`**».

Le fichier devient le flux de sortie courant.

`telling(F)`

`F` s'unifie au fichier en cours.

`tell(user).`

Le flux courant de sortie redevient l'utilisateur.

`told.`

Fermeture du fichier ouvert en écriture.

Exemple

```
tell(f1),
write('Ecrire dans f1'),
told,
write('Ecrire hors de f1'),nl.
```

Les prédicats du 2nd ordre

- Les prédicats du second ordre donnent toutes les solutions à une question. Ce sont **`findall/3`**, **`bagof/3`**, **`setof/3`**.

- Pour trouver un ensemble d'objets satisfaisant un prédicat, on peut compter sur le *Backtraking*, mais on perd la solution précédente.

Les prédicats `bagof/3`

```
élève(françois, info, 2).
élève(isa, info, 2).
élève(françois, info, 3).
élève(paul, math, 3).
masculin(françois).
masculin(paul).
féminin(isa).

?- bagof(X, élève(X, info, 2), B).
B = [françois, isa].

?- bagof(X, élève(X, info, Y), B).
B =[françois, isa], Y = 2 ;
B = [françois], Y = 3 ;
No.
```

Utilisation d'un quantificateur Existentielles

On peut utiliser le quantificateur quelque soit avec le symbole ^□

```
?- bagof(X, Y^élève(X, info, Y), B).
B = [françois, isa, françois].

?- bagof(X, Z^Y^élève(X, Y, Z), B).
B = [françois, isa, françois, paul].

?- bagof(X, Y^Z^(élève(X, Y, Z),
  masculin(X)), B).
B = [françois, françois, paul].
```

Les prédicats `setof/3`

`setof/3 = bagof/3` sauf que

- mettre les éléments en ordre du plus petit au plus grand,

- Enlever les doubles.

Exemple

B.C.

etudiant(jean,b).

etudiant(marie,a).

etudiant(paul,c).

etudiant(juli,b).

-? setof(**N, E^etudiant(E,N),L).**

L=[a,b,c]

-? **bagof(N, E^etudiant(E,N),L).**

L=[b,a,c,b]

Le prédicat `findall/3`

```
élève(françois, 2, info).
élève(isa, 2, info).
élève(françois, 3, instru).

?- findall(X, élève(X, 2, info), B).
B = [françois, isa].

?- findall(X, élève(X, Y, Z), B).
B = [françois, isa, françois].
```

`Findall/3 = bagof/3` sauf que
toutes les variables sont existentielles.

Exemple:

```
lettre(a,1,voy).
lettre(b,2,cons).
lettre(e,5,voy).

-?findall(L,lettre(L,N,T),Alpha).
Alpha = [a,b,e]

-?bagof(L,N^T^lettre(L,N,T),Alpha).
Alpha = [a,b,e]
```