Ficha Prática 7

José Creissac Campos, António Nestor Ribeiro
{jose.campos, anr}@di.uminho.pt
PPIII (LESI)

2002/03

Objectivos

1. Praticar Cut e Fail (continuação).
2. Praticar a utilização de `see` e `tell`.
3. Praticar a utilização de `assert` e `retract`.

Cut e Fail

1. Considere o seguinte predicado:

```
% norep(L1,L2) :- L2 é a lista L1 sem elementos repetidos
norep([],[]).
norep([H|T1],L) :- member(H,T1), norep(T1,L).
norep([H|T1],[H|T2]) :- \+ member(H,T1), norep(T1,T2).
```

   O predicado funciona mas tem um problema:

```
?- norep([a,d,a,d,a],L).
L = [d, a] ;
L = [d, a] ;
No
```

   O predicado calcula o resultado certo, mas mais que uma vez!

   Utilizando o `trace` e árvores de prova, identifique onde está o problema e procure resolvê-lo utilizando *cuts*.

2. Escreva cada um dos seguintes predicados sobre listas, procurando utilizar *cuts* sempre que apropriado.

   (a) predicado `del/3`:

   ```
   % del(E,L1,L2) :- L2 é a lista L1 com todas as ocorrências
   %                 de E removidas
   ```

   (b) predicado `interseccao/3`:

   ```
   % interseccao(L1,L2,L3) :- L3 é a intersecção das listas
   %                           L1 e L2
   ```

   (c) predicado `quicksort/2`:

   ```
   % quicksort(L1,L2) :- L2 é uma versão ordenada de L1
   %                     (utilizando o algoritmo quicksort)
   ```

assert/retract (retractall e abolish)

Com base no que sabe sobre os predicados `assert`, `retract`, `retractall` e `abolish` procure prever o resultado das queries sublinhadas:

1.
```
?- [user]
impar(1). impar(2). impar(3).
^D
?- assert(par(2)).
?- assert(par(3)).
?- assert(par(4)).
?- listing(impar).
?- listing(par).
?- assert(impar(5)).
?- retract(par(3)).
?- retract(impar(2)).
```

2.
```
?-  [user]
:- dynamic m2/1, m3/1.
m2(2). m2(4). m2(6).
m3(3). m3(6). m3(8).
^D
?- retract(m3(8)).
?- assert(m2(8)).
?- assert(m3(9)).
```

3.
```
?-  [user]
:- dynamic m23/1.
m23(X) :- m2(X).
m23(X) :- m3(X).
nãoComum(X) :- m2(X), not(m3(X)).
nãoComum(X) :- m3(X), not(m2(X)).
^D
?- assert(comum(X):- m2(X), m3(X)).
?- retract(comum(_)).
?- retract(comum(_):- m2(X), m3(X)).
?- retractall(m23(_)).
?- retractall(nãoComum(_)).
?- abolish(nãoComum(_)).
```

see/tell (seen/told, seing/telling e read/write)

Com base no que sabe sobre os predicados `see` e `tell` procure prever o resultado das seguintes queries:

1.
```
?- write(´Escreva o seu nome:´), read(N),
                atom_concat(´Olá ´, N, S), write(S).
?-
?- write(´Escreva o seu nome:´), read(N), tell(´Teste.txt´),
                atom_concat(´Olá ´, N, S), write(S), told.
```

Veja o conteúdo do ficheiro Teste.txt
Crie o ficheiro Entrada.txt com uma linha contendo o seu nome sob a forma de átomo prolog.

```
?- see(´Entrada.txt´), read(N),
                atom_concat(´Olá ´, N, S), write(S), seen.
?- tell(t1), write(olá), tell(t2), write(olé),
             told, write(oli), told, write(olo).
?- tell(t1), write(olá), telling(F), tell(t2), write(olé),
             tell(F), write(oli), told, write(olo).
```

Exercícios[1]

1. Num ficheiro encontram-se por ordem aleatória predicados que representam definições de figuras geométricas sob a forma: `quadrado(p, tam)`, `triangulo(p1, p2, p3)`, `rectangulo(p1, p2)` e `circunf(p, r)`:

   (a) Escreva o predicado `lerFiguras/1` que carrega para a Base de Conhecimento todas as figuras presentes no ficheiro.

   (b) Escreva o predicado `lerQuadrados/1` que carrega para a Base de Conhecimento todos os quadrados presentes no ficheiro.

   (c) Escreva o predicado `maiorCircunf/2` que define qual a maior circunferência presente no ficheiro.

   (d) Escreva ainda o predicado `listaQuad/2` que define a lista dos quadrados presentes na Base de Conhecimento que possuem como lado um valor dado.

2. Termos da forma `telefonou(pessoa1, pessoa2)` são utilizados numa Base de Conhecimento para indicar que uma dada pessoa contactou outra. Escreva predicados que lhe permitam:

   (a) Carregar para a BC tais factos a partir de um ficheiro F.

   (b) Determinar qual a pessoa que mais vezes ligou para a pessoa X.

   (c) Determinar qual o conjunto de pessoas que a pessoa Y contactou.

   (d) Determinar a lista de pares `(pessoa, lista_de_pessoas_contactadas)`.

   (e) Sabendo que cada chamada custa P, gravar num ficheiro, para cada pessoa, o custo das suas chamadas.

[1] Exercícios baseados no Exame de 18 de Novembro de 1999.