Programação em Lógica

Prof. A. G. Silva

29 de setembro de 2016

Exercícios

Sugestões de resoluções dos exercícios 1 a 6, da aula passada, em cor púrpura. Exercícios 7 e 8 ainda não resolvidos; pense em suas implementações, tomando como base o código da questão 6.

1. Escreva um predicado `estrelas(N)` que imprime `N` caracteres "`*`" no dispositivo de saida.

```
estrelas(0) :- !.
estrelas(N) :- N > 0,
               M is N - 1,
               write('*'),
               estrelas(M).
```

Exercícios (cont...)

2. Escreva um predicado `guess(N)` que incita o usuário a adivinhar o número `N`. O predicado repetidamente lê um número, compara-o com `N`, e imprime "`Muito baixo!`", "`Acertou!`", "`Muito alto!`, conforme o caso, orientando o usuário na direção certa.

```
compara(N,X) :- N == X,
                write('Acertou! ').
compara(N,X) :- N < X,
                write('Muito alto! '),
                guess(N).
compara(N,X) :- N > X,
                write('Muito baixo! '),
                guess(N).
guess(N) :- write('Adivinhe N: '),
            read(X),
            compara(N,X).
```

Exercícios (cont...)

3. Escreva um predicado que lê uma linha e imprime a mesma linha trocando todos os caracteres 'a' por 'b'.

```
troca(R) :- write('Escreva algo: '),
            read(X),
            string_to_list(X,Y),
            troca(Y,R).
troca([],_) :- !.
troca([A|B],R) :- char_code('a',Code),
                  A=:=Code, write('b'),
                  troca(B,R).
troca([A|B],R) :- char_code(Char,A),
                  write(Char),
                  troca(B,R).
```

Exercícios (cont...)

4. Implemente os predicados `liga`, `desliga` e `lâmpada` para que eles funcionem conforme indicado pelos exemplos a seguir:

```
?- liga, lâmpada(X).
X = acessa
Yes
?- desliga, lâmpada(X).
X = apagada
Yes
```

```
:- dynamic lampada/1.

lampada(acesa).
liga :- retract(lampada(_)),
        assert(lampada(acesa)).
desliga :- retract(lampada(_)),
           assert(lampada(apagada)).
```

Exercícios (cont...)

5. O predicado `asserta` adiciona um fato à base de dados, incondicionalmente, mesmo que ele já esteja lá. Para impedir essa redundância, defina o predicado `memorize`, tal que ele seja semelhante a `asserta`, mas só adicione à base de dados fatos inéditos.

```
% (testar com um predicado dinamico)

memorize(A) :- retractall(A),
               asserta(A),
               !.
memorize(A) :- asserta(A).
```

Exercícios (cont...)

6. Suponha um robô capaz de andar até um certo local e pegar ou soltar objetos. Além disso, suponha que esse robô mantém numa base de dados sua posição corrente e as respectivas posições de uma série de objetos. Implemente os predicados `pos(Obj,Loc)`, `ande(Dest)`, `pegue(Obj)` e `solte(Obj)`, de modo que o comportamento desse robô possa ser simulado, conforme exemplificado a seguir:

```
?- pos(O,L).
O = robô
L = garagem ;
O = tv
L = sala ;
No

?- pegue(tv), ande(quarto), solte(tv), ande(cozinha).
anda de garagem até sala
pega tv
anda de sala até quarto
solta tv
anda de quarto até cozinha
Yes
```

Exercícios (cont...)

6.
```
:- dynamic pos/2.
:- dynamic posse/1.

pos(robo, garagem).
pos(tv, sala).
pos([], []). %objeto indefinido em local indefinido
posse([]).  %objeto que o robo esta segurando, inicialmente indefinido

pegue(O) :- pos(robo,L1), pos(O,L2),
            retract(posse(_)),
            asserta(posse(O)),
            retract(pos(robo,_)), asserta(pos(robo,L2)),
            format('anda de ~w ate ~w\npega ~w\n', [L1,L2,O]).
ande(L) :- pos(robo,L1),
           retract(pos(robo,_)), asserta(pos(robo,L1)),
           format('anda de ~w ate ~w\n', [L1,L]), retract(pos(robo,_)),
           asserta(pos(robo,L)),
           posse(O),
           retract(pos(O,_)), asserta(pos(O,L)).
solte(O) :- posse(O),
            retract(posse(_)),
            asserta(posse([])).
```

Exercícios (cont...)

7. *(não resolvido)* Modifique o programa desenvolvido no exercício anterior de modo que, quando for solicitado ao robô pegar um objeto cuja posição é desconhecida, ele pergunte ao usuário onde está esse objeto e atualize a sua base de dados com a nova informação. Veja um exemplo:

```
?- pos(O,L).
O = robô
L = cozinha ;
O = tv
L = quarto ;
No

?- pegue(lixo), ande(rua), solte(lixo), ande(garagem).
Onde está lixo? quintal
anda de cozinha até quintal
pega lixo
anda de quintal até rua
solta lixo
anda de rua até garagem
Yes

?- pos(O,L).
O = robô
L = garagem ;
O = lixo
L = rua ;
O = tv
L = quarto ;
No
```