Práctica 2: Programación en Prolog

Programación Declarativa: Lógica y Restricciones

DIA

Vidal Peña, Arturo
W140307

31 de mayo de 2019

Índice de contenidos

1. Código empleado y las explicaciones

1.1. Predicado menor/4

```
menor(A, B, Comp, M) :-
 functor(X, Comp, 2),
 arg(1,X,A),
 arg(2,X,B),
 call(X)->
 M = A;
 M = B.
```

En este predicado, preparo con *functor/3* una estructura en *X* de tres términos, y añado *Comp, A* y *B* con los predicados arg/3.
Luego, con *call/1* llamo a esa estructura y guardo el resultado en *M*.

1.2. Predicado menor_o_igual/4

```
menor_o_igual(X, Y) :-
 menor_o_igual_libre(X, Y);
 (nonvar(X), nonvar(Y), functor(X, Term1, Aridad1), functor(Y, Term2, Aridad2),
  (menor_nombre(Term1, Term2);
     menor_aridad(Term1,Term2,Aridad1,Aridad2);
     menor_o_igual_argumento(X, Y, Term1, Term2, Aridad1, Aridad2))).

menor_o_igual_libre(X, Y) :-
 var(X) = Y;
 var(Y) = X.

menor_nombre(Term1, Term2) :-
 Term1 @< Term2.

menor_aridad(Term1, Term2, Aridad1, Aridad2) :-
 Term1 == Term2,
 Aridad1 < Aridad2.

menor_o_igual_argumento(X, Y, Term1, Term2, Aridad1, Aridad2) :-
 Term1 == Term2,
 Aridad1 == Aridad2,
 X =.. [Term1|ListaX],
 Y =.. [Term2|ListaY],
 menor_igual_argumento_rec(ListaX, ListaY).

menor_igual_argumento_rec([], []).

menor_igual_argumento_rec([Arg1|T1], [Brg1|T2]) :-
 (number(Arg1), number(Brg1), Arg1 =< Brg1 ;
    Arg1 @=< Brg1),
 menor_igual_argumento_rec(T1, T2).
```

1.3. Predicado lista_hojas/2

```
lista_hojas([], []).

lista_hojas([H|T], [tree(H, void, void)|Hojas]) :-
 lista_hojas(T, Hojas).
```

1.4. Predicado hojas_arbol/3

```
hojas_arbol([Hoja|Hojas],Comp,Arbol) :-
  hojasArbolRec(Hojas,Comp,Hoja,Arbol).

hojasArbolRec([],_,Arbol,Arbol).

hojasArbolRec([H|T],Comp,Arbol,Tree) :-
  arg(1,H,RaizIzq),
  arg(1,Arbol,Raiz),
  menor(Raiz,RaizIzq,Comp,M),
  hojasArbolRec(T,Comp,tree(M,Arbol,H),Tree).
```

1.5. Predicado ordenacion/3

```
ordenacion(Arbol,Comp,Orden) :-
  ordenacion_aux(Arbol,Comp,[],Orden).

ordenacion_aux(tree(Element,void,void),_,Orden,OrdenF) :-
  append(Orden,[Element],OrdenF).

ordenacion_aux(tree(Element,Left,Right),Comp,Orden,OrdenF) :-
  append(Orden,[Element],OrdenS),
  makeList(tree(Element,Left,Right),[],Lista),
  borraElemento(Element,Lista,Salida),
  lista_hojas(Salida,Hojas),
  hojas_arbol(Hojas,Comp,Arbol),
  ordenacion_aux(Arbol,Comp,OrdenS,OrdenF).

makeList(tree(Element,void,void), Lista, [Element|Lista]).

makeList(tree(_,Left,Right), Lista, ListaSalida) :-
  Right = tree(ElementRight,_,_),
  append(Lista,[ElementRight],Lista2),
  makeList(Left,Lista2,ListaSalida).

borraElemento(_,[],[]).

borraElemento(Elemento,[Elemento|T],T).

borraElemento(Elemento,[H|T1],[H|T2]) :-
  borraElemento(Elemento,T1,T2).
```

1.6. Predicado ordenar/3

```
ordenar(Lista,Comp,Orden) :-
  lista_hojas(Lista,Hojas),
  hojas_arbol(Hojas,Comp,Arbol),
  ordenacion(Arbol,Comp,Orden).
```

2. Pruebas realizadas

Todas las pruebas a continuación han dado los valores esperados.

1. nat/1:
   - nat(0),
   - nat(s(0)),
   - nat(s(s(s(s(0))))),
   - nat(t)
2. menor_igual/2:
   - menor_igual(s(0),s(0)),
   - menor_igual(s(s(0)),s(s(0))),
   - menor_igual(s(0),s(s(0)))
3. resta/2:
   - resta(s(0),0,s(0)),
   - resta(s(0),s(0),0),
   - resta(s(s(s(s(0)))),s(s(s(s(0)))),0)
4. par/1:
   - par(0)
   - par(s(s(0))),
   - par(s(s(s(s(0)))))
5. iguales/2:
   - iguales(s(0),s(0)),
   - iguales(s(0),s(0)),
   - iguales(s(s(s(s(0)))),s(s(s(s(0))))),
6. color/1:
   - color(r),
   - color(v),
   - color(am),
   - color(a),
   - color(b)
7. esTorre/1:
   - esTorre([pieza(s(0),s(s(0)),s(0),r)]),
   - esTorre([pieza(s(0),s(s(0)),s(0),r),pieza(s(0),s(0),s(0),a)]),
   - esTorre([pieza(s(s(s(0))),s(s(s(0))),s(s(s(0))),am),pieza(s(s(s(0))),s(s(s(0))),s(s(s(0))),v)])

8. alturaTorre/2:
    - alturaTorre([pieza(s(0),s(0),s(0),r)],s(0)),
    - alturaTorre([pieza(s(0),s(0),s(0),r),pieza(s(s(0)),s(s(0)),s(0),a)],s(s(s(0)))),
    - alturaTorre([pieza(s(0),s(0),s(0),r),pieza(s(0),s(0),s(0),r),pieza(s(0),s(s(0)),s(0),r)],s(s(s(s(0)))))
9. coloresTorre/2:
    - coloresTorre([pieza(s(0),s(0),s(0),r)],[r]),
    - coloresTorre([pieza(s(0),s(0),s(0),r),pieza(s(0),s(s(0)),s(0),a)],[r,a]),
    - coloresTorre([pieza(s(0),s(s(0)),s(0),am),pieza(s(0),s(0),s(0),r),pieza(s(0),s(s(0)),s(0),a)],[am,r,a])
10. coloresIncluidos/2:
    - coloresIncluidos([pieza(s(0),s(0),s(0),r)],[pieza(s(0),s(0),s(0),r)]),
    - coloresIncluidos([pieza(s(0),s(0),s(0),a),pieza(s(0),s(0),s(0),a)],[pieza(s(s(0)),s(0),s(s(0)),a)]),
    - coloresIncluidos([pieza(s(0),s(0),s(0),r),pieza(s(0),s(0),s(0),v),pieza(s(0),s(0),s(0),a)])
11. esEdificioPar/2:
    - esEdificioPar([[a,a],[v,v],[a,a],[v,v]]),
    - esEdificioPar([[a,a,r,v,am,r],[v,r,v,v,am,r],[v,r,v,r,a,r],[v,r,a,r,v,r],[v,r,a,r,v,am]]),
    - esEdificioPar([[a,a,am,r],[v,v,am,r],[v,r,a,r],[v,r,a,r],[v,r,a,am]])

3. Comentarios adicionales