Código fuente

Prolog

```
                    %Trabajo final de Taller de Nuevas Tecnologias - UNPSJB
%Integrantes
%   Cobo Medvedsky, Elias Daniel
%   Ligorria, Alexis Enrique
%   Suazo, Leonardo Ezequiel
%-------------------------------------------------------------------------------------------------%
:- dynamic
        kb/1.
:- dynamic
        ee/2.
ee(X,X):-
    !,
    fail.
%-------------------------------------------------------------------------------------------------%
                                %---------- LIMPIAR BC ----------%
% Limpiar la base de conocimiento
limpiarBC:-
    retractall(kb(_)),
    retractall(ee(_,_)).
%-------------------------------------------------------------------------------------------------%
                                %---------- MOSTRAR ----------%
% Mostrar las clausulas y sus relaciones de importancia
mostrarBC(KB, EE):-
    findall(X,kb(X),KB),
    findall([Y,Z], ee(Y,Z), EE).
%-------------------------------------------------------------------------------------------------%
                                %---------- EXPANDIR ----------%
% Inserta A en la BC y las relaciones de importancia asociadas
expandirBC(A,[B,C]):-
    expandirBC(A),
    insertarSuEE(B,A),
    insertarMaEE(C,A).
%-------------------------------------------------------------------------------------------------%
% Inserta A en la BC
expandirBC(A):-
    kb(A),!.
expandirBC(A):-
    asserta(kb(A)).
%-------------------------------------------------------------------------------------------------%
% Inserta las relaciones de importancia que contien las creencias a lo sumo tan importanes como A
insertarSuEE([],_):-!.
insertarSuEE([B|C],A):-
    insertarSuEE(C,A),
    insertarEE(A,B),!.
insertarSuEE(B,A):-
    insertarEE(A,B),!.
%-------------------------------------------------------------------------------------------------%
% Inserta las relaciones de importancia que contien las creencias mas importanes que A
insertarMaEE([],_):-!.
insertarMaEE([B|C],A):-
    insertarMaEE(C,A),
```

```
52      insertarEE(B,A),!.
53  insertarMaEE(B,A):-
54      insertarEE(B,A),!.
55  %-------------------------------------------------------------------------------------------------%
56  % Inserta una ralacion de importancia particular
57  insertarEE(A,B):-
58      ee(A,B),!.
59  insertarEE(A,B):-
60      asserta(ee(A,B)).
61  %-------------------------------------------------------------------------------------------------%
62                                      % ---------- CONTRAER ---------- %
63  % Presenta las diferentes opciones por las cuales se puede contraer
64  opcionesContraccion(A, EMImp, CMImp, EMCant):-
65      findall(K0,deduce(A,K0),K),
66      listaNoVacia(K),
67      menosImportantes(K, EMImp),
68      masImportantes(K, CMImp),
69      minimosDeCadaUno(K,EMCant).
70  %-------------------------------------------------------------------------------------------------%
71  contraccionPorMinimaCantidad(A):-
72      findall(K0,deduce(A,K0),K),
73      minimosDeCadaUno(K,Min),
74      eliminarDeBC(Min).
75  %-------------------------------------------------------------------------------------------------%
76  contraccionPorMenosImportantes(A):-
77      findall(K0,deduce(A,K0),K),
78      menosImportantes(K,MenImp),
79      eliminarDeBC(MenImp).
80  %-------------------------------------------------------------------------------------------------%
81  contraccionPorMantenerMasImportantes(A):-
82      findall(K0,deduce(A,K0),K),
83      masImportantes(K,MasImp),
84      eliminarDeBC(MasImp).
85  %-------------------------------------------------------------------------------------------------%
86  % verifica que la longitud de una determinada lista sea mayor a cero
87  listaNoVacia(K):-
88      longLista(K,0),
89      !,
90      fail.
91  listaNoVacia(_).
92  %-------------------------------------------------------------------------------------------------%
93                                      % ---------- CONSOLIDAR ---------- %
94  % Presenta las diferentes opciones por las cuales se puede consolidar
95  opcionesConsolidar(EMImp, ConsMasImp, ElimMinCant):-
96      opcionesContraccion(false, EMImp, ConsMasImp, ElimMinCant),
97      !.
98  opcionesConsolidar([], [], []).
99  %-------------------------------------------------------------------------------------------------%
100 consolidarPorMinimaCantidad:-
101     contraccionPorMinimaCantidad(false).
102 %-------------------------------------------------------------------------------------------------%
103 consolidarPorMenosImportantes:-
104     contraccionPorMenosImportantes(false).
105 %-------------------------------------------------------------------------------------------------%
106 consolidarPorMantenerMasImportantes:-
107     contraccionPorMantenerMasImportantes(false).
108 %-------------------------------------------------------------------------------------------------%
109                                     % ---------- REVISAR ---------- %
110 % Presenta las diferentes opciones por las cuales se puede revisar
111 opcionesRevision(A, EMImp, ConsMasImp, ElimMinCant):-
112     opcionesContraccion(no(A), EMImp, ConsMasImp, ElimMinCant),
113     !.
114 opcionesRevision(_,[], [], []).
115 %-------------------------------------------------------------------------------------------------%
116 revisarPorMenosImportantes(A):-
```

```
117     contraccionPorMenosImportantes(no(A)),
118     expandirBC(A).
119 %-------------------------------------------------------------------------------------------------%
120 revisarPorMantenerMasImportantes(A):-
121     contraccionPorMantenerMasImportantes(no(A)),
122     expandirBC(A).
123 %-------------------------------------------------------------------------------------------------%
124 revisarPorMinimaCantidad(A):-
125     contraccionPorMinimaCantidad(no(A)),
126     expandirBC(A).
127 %-------------------------------------------------------------------------------------------------%
128                                     % ---------- KERNELS  ---------- %
129 %% subconjuntos minimales de K  que deriban X
130 deduce(X,K):-
131     posiblesSubconjuntos(L),
132     unificarSubconjunto(L,K),
133     deduceEn(X,K),
134     subconjuntoMinimo(X,K).
135 %-------------------------------------------------------------------------------------------------%
136 % genera todos los posibles subconjuntos minimmales de la BC
137 posiblesSubconjuntos(L):-
138     findall(X,kb(X),KB),
139     findall(Y, posiblesSubconjuntosL(KB,Y),L0),
140     eliminarRepetidosLista(L0,L).
141 %-------------------------------------------------------------------------------------------------%
142 posiblesSubconjuntosL(X,X).
143 posiblesSubconjuntosL(K,L):-
144     eliminarUno(K,K1),
145     posiblesSubconjuntosL(K1,L).
146 %-------------------------------------------------------------------------------------------------%
147 eliminarUno([_|Xs],Xs).
148 eliminarUno([X|Xs],[X|Y]):-
149     eliminarUno(Xs,Y).
150 %-------------------------------------------------------------------------------------------------%
151 %elimina los repetidos de una lista
152 eliminarRepetidosLista([X|Y],[X|Z]):-
153     tieneRepetidoLista(X,Y),
154     noElemEnLista(X,Y,L1),
155     eliminarRepetidosLista(L1,Z),!.
156 eliminarRepetidosLista([X|Y],[X|Z]):-
157     eliminarRepetidosLista(Y,Z),!.
158 eliminarRepetidosLista(X,X).
159 %-------------------------------------------------------------------------------------------------%
160 %verifica si una lista tiene elementos repetidos
161 tieneRepetidoLista(X, X):-
162     !.
163 tieneRepetidoLista(X, [X|_]):-
164     !.
165 tieneRepetidoLista(X, [_|Y]):-
166     tieneRepetidoLista(X,Y).
167 %-------------------------------------------------------------------------------------------------%
168 %verifica si una lista no contiene a un elemento en particular
169 noElemEnLista(X, [Y|X], Y):-
170     !.
171 noElemEnLista(X, [X|Y], Z):-
172     noElemEnLista(X, Y, Z),
173     !.
174 noElemEnLista(X, [W|Y], [W|Z]):-
175     noElemEnLista(X, Y, Z),
176     !.
177 noElemEnLista(_, Z, Z).
178 %-------------------------------------------------------------------------------------------------%
179 % Se unifica la BC con alguno de los subconjuntos de la lista
180 unificarSubconjunto([Y|_], Y).
181 unificarSubconjunto([_|Ys], Z):-
```

```
182     unificarSubconjunto(Ys,Z).
183 %-------------------------------------------------------------------------------------------%
184 deduceSubc([X|[]], X1):-
185     fnc(X,X1),!.
186 deduceSubc([X|Xs], Z):-
187     deduceSubc(Xs,Y1),fnc(X,X1),
188     unirConj(X1,Y1, Z),!.
189 deduceSubc(X, X1):-
190     fnc(X,X1).
191 %-------------------------------------------------------------------------------------------%
192 % Verifica si el subconjunto deduce Alpha(X)
193 deduceEn(X,Y):-
194     fnc(no(X),X1),
195     deduceSubc(Y,Y0),
196     resolverYSimplificar(Y0 -y- X1,false).
197 %-------------------------------------------------------------------------------------------%
198 % verifica si el subconjunto es minimo
199 subconjuntoMinimo(X,Y):-
200     findall(Z0,unaSubListaDeduc(X,Y,Z0),Z),
201     longLista(Z,0).
202 %-------------------------------------------------------------------------------------------%
203 unaSubListaDeduc(X,Y,Ys):-
204     eliminarUno(Y,Ys),
205     deduceEn(X,Ys).
206 %-------------------------------------------------------------------------------------------%
207 resolverYSimplificar(X,Z):-
208     resolver(X,X1),
209     simplificar(X1, X2),
210     simplificar(X2,X3),
211     simplificarConjunciones(X3,Z).
212 %-------------------------------------------------------------------------------------------%
213 resolver(X-y-Y,Z):-
214     resolver(X,X1),
215     resolver(Y,Y1),
216     resolverDos([X1]-y-[Y1],Z0),
217     aplanarSimple(Z0,Z),
218     !.
219 resolver([X],Z):-
220     resolver(X,Z0),
221     aplanarSimple(Z0,Z),
222     !.
223 resolver(X,X).
224 %-------------------------------------------------------------------------------------------%
225 resolverDos([X],[Z]):-
226     resolverDos(X,Z),
227     !.
228 resolverDos(X -y- Y,Z):-
229     resolverDos(X,X1),
230     resolverConj(X1,Y,Z0),
231     aplanarUnirDisj(Z0,Z),
232     !.
233 resolverDos(X, Z):-
234     aplanarUnirDisj(X,Z),!.
235 %-------------------------------------------------------------------------------------------%
236 resolverConj([Y],[X1 -o- X2], Z):-
237     resolverConj([X1],[Y], Z0),
238     resolverDos(Z0,Z1),
239     unirConj([X2],[Y], Z2),
240     resolverDos(Z2,Z3),
241     unirDisj(Z1,Z3,Z4),
242     resolverDos(Z4,Z),
243     !.
244 resolverConj([X1 -o- X2],[Y], Z):-
245     resolverConj([X1],[Y], Z0),
246     resolverDos(Z0,Z1),
```

```
247     unirConj([X2],[Y], Z2),
248     resolverDos(Z2,Z3),
249     unirDisj(Z1,Z3,Z4),
250     resolverDos(Z4,Z),
251     !.
252 resolverConj(X,Y, Z):-
253     unirConj(X,Y, Z).
254 %-------------------------------------------------------------------------------------------------%
255 aplanarUnirDisj(X,Z):-
256     aplanarLastDisj(X, X1, Z0),
257     aplanarUnirDisj(X1,Z2),
258     unirDisj(Z2,[Z0],Z),
259     !.
260 aplanarUnirDisj(X,X).
261 %-------------------------------------------------------------------------------------------------%
262 aplanarLastDisj(X -y- [Y], X0, Z):-
263     aplanarLastDisj(X, X0, X1),
264     unirConj(X1,Y,Z),!.
265 aplanarLastDisj(X -o- [Y], X, Y):-
266     !.
267 aplanarLastDisj([X],[],X).
268 %-------------------------------------------------------------------------------------------------%
269 aplanarSimple(X,Z):-
270     aplanarConjSimple(X,Z0),
271     aplanarDisjSimple(Z0,Z).
272 %-------------------------------------------------------------------------------------------------%
273 aplanarConjSimple([X],Z):-
274     aplanarConjSimple(X,Z),!.
275 aplanarConjSimple([X -o- Y],Z):-
276     aplanarConjSimple(X-o-Y,Z),!.
277 aplanarConjSimple(X -o- Y,Z):-
278     aplanarConjSimple(X,X1),
279     aplanarConjSimple(Y,Y1),
280     unirDisj([X1],[Y1],Z),
281     !.
282 aplanarConjSimple(X -y- Y,Z):-
283     aplanarConjSimple(X,X1),
284     aplanarConjSimple(Y,Y1),
285     unirConj(X1,Y1,Z),
286     !.
287 aplanarConjSimple(X,X).
288 %-------------------------------------------------------------------------------------------------%
289 aplanarDisjSimple([X -o- Y],Z):-
290     aplanarDisjSimple(X,X1),
291     aplanarTotalSimple(Y,Y1),
292     unirDisj(X1,Y1,Z0),
293     aplanarTotalSimple(Z0,Z),
294     !.
295 aplanarDisjSimple(X -o- Y,Z):-
296     aplanarDisjSimple(X,X1),
297     aplanarTotalSimple(Y,Y1),
298     unirDisj(X1,Y1,Z0),
299     aplanarTotalSimple(Z0,Z),!.
300 aplanarDisjSimple(X,Z):-
301     aplanarTotalSimple(X,Z).
302 %-------------------------------------------------------------------------------------------------%
303 aplanarTotalSimple([X -o- Y],Z):-
304     aplanarTotalSimple(X-o-Y,Z),
305     !.
306 aplanarTotalSimple(X,X).
307 %-------------------------------------------------------------------------------------------------%
308 % elimina las sentencias falsas de la disjuncion
309 simplificar(false -o- X, X1):-
310     simplificar(X,X1),
311     !.
```

```
312 simplificar(X -o- false, X1):-
313     simplificar(X,X1),
314     !.
315 simplificar(X -o- Y, X1):-
316     esContradiccion(Y),
317     simplificar(X,X1),
318     !.
319 simplificar(X -o- Y, X1 -o- Y):-
320     simplificar(X,X1),
321     !.
322 simplificar(X, false):-
323     esContradiccion(X),
324     !.
325 simplificar(X, X).
326 %-------------------------------------------------------------------------------------------------%
327 %Verifica si el los valores ingresado son contradictorios
328 esContradiccion([X]):-
329     esContradiccion(X),
330     !.
331 esContradiccion(X -y- X1):-
332     complementarios(X,X1),
333     !.
334 esContradiccion(Y -y- X):-
335     tieneComplementarioConj(X,Y),
336     !.
337 esContradiccion(Y -y- _):-
338     esContradiccion(Y),
339     !.
340 %-------------------------------------------------------------------------------------------------%
341 tieneComplementarioConj(X, X1):-
342     complementarios(X,X1),!.
343 tieneComplementarioConj(X, _ -y- X1):-
344     complementarios(X,X1),!.
345 tieneComplementarioConj(X, Y -y- _):-
346     tieneComplementarioConj(X,Y).
347 %-------------------------------------------------------------------------------------------------%
348 % elimina las sentencias repetidas en las conjunciones
349 simplificarConjunciones(X -o- Y, X1 -o- Y1):-
350     eliminarConjRepetidas(X,X1),
351     simplificarConjunciones(Y, Y1),
352     !.
353 simplificarConjunciones(X , X1):-
354     eliminarConjRepetidas(X,X1).
355 %-------------------------------------------------------------------------------------------------%
356 eliminarConjRepetidas(X -y- X,X):-
357     !.
358 eliminarConjRepetidas(Y -y- X,Z -y- X):-
359     tieneConjRepetido(X,Y),
360     sinConjX(X,Y,L1),
361     eliminarConjRepetidas(L1,Z),
362     !.
363 eliminarConjRepetidas(Y -y- X,Z -y- X):-
364     eliminarConjRepetidas(Y,Z),
365     !.
366 eliminarConjRepetidas([X] , [Z]):-
367     eliminarConjRepetidas(X,Z),
368     !.
369 eliminarConjRepetidas(X , X).
370 %-------------------------------------------------------------------------------------------------%
371 %Verifica si tiene 2 elementos repetidos
372 tieneConjRepetido(X, X):-
373     !.
374 tieneConjRepetido(X, _ -y- X):-
375     !.
376 tieneConjRepetido(X, Y -y- _):-
```

```
377     tieneConjRepetido(X,Y).
378 %-------------------------------------------------------------------------------------------------%
379 sinConjX(X, X -y- Y, Y):-
380     !.
381 sinConjX(X, Y -y- X, Z):-
382     sinConjX(X, Y, Z),
383     !.
384 sinConjX(X, Y -y- W, Z -y- W):-
385     sinConjX(X, Y, Z),
386     !.
387 sinConjX(_, Z, Z).
388 %-------------------------------------------------------------------------------------------------%
389                                 % ---------- Que eliminar---------- %
390 % minima cantidad a descartar
391 minimosDeCadaUno(X,Y):-
392     findall(L,unoDeCadaUnoSinRep(X,L), L2),
393     listaNoVacia(L2),
394     minimaLista(L2,Y),
395     !.
396 minimosDeCadaUno(_,[]).
397 %-------------------------------------------------------------------------------------------------%
398 unoDeCadaUnoSinRep(X,Y):-
399     unoDeCadaUno(X, W),
400     eliminarRepetidosLista(W,Y).
401 %-------------------------------------------------------------------------------------------------%
402 unoDeCadaUno([X|Xs],[X1|L]):-
403     unificarSubconjunto(X,X1),
404     unoDeCadaUno(Xs,L).
405 unoDeCadaUno([X],[X1]):-
406     unificarSubconjunto(X,X1).
407 %-------------------------------------------------------------------------------------------------%
408 minimaLista(L,X):-
409     minimaListaCantidad(L,0,X),
410     !.
411 %-------------------------------------------------------------------------------------------------%
412 minimaListaCantidad(L,C,Lz):-
413     minimaListaCantidadC(L,C,Lz),
414     !.
415 minimaListaCantidad(L,C,Lz):-
416     succ(C, C2),
417     minimaListaCantidad(L,C2,Lz),
418     !.
419 minimaListaCantidadC([L|_], C, L):-
420     longLista(L,C),
421     !.
422 minimaListaCantidadC([_|Ls], C, Lz):-
423     minimaListaCantidadC(Ls,C,Lz),
424     !.
425 %-------------------------------------------------------------------------------------------------%
426 % Longitd de una lista
427 longLista([],0):-
428     !.
429 longLista([_|Xs],C2):-
430     longLista(Xs,C),
431     succ(C,C2),
432     !.
433 %-------------------------------------------------------------------------------------------------%
434 % menos importantes a descartar
435 menosImportantes(La,Lz):-
436     findall(X,menosImportantes0(La,X), Ln),
437     listaNoVacia(Ln),
438     eliminarRepetidosLista(Ln,Lz),
439     !.
440 %-------------------------------------------------------------------------------------------------%
441 menosImportantes(La,Lp):-
```

```
442     aplanarTodosLista(La, Lp).
443
444 menosImportantes0(La,Lz):-
445     unificarSubconjunto(La,Ln),
446     menosImportante(Ln, Lz).
447 %-----------------------------------------------------------------------------------------------------%
448 menosImportante(L,X):-
449     unificarSubconjunto(L,X),
450     noHayAlgunoMenosImportante(X,L).
451 %-----------------------------------------------------------------------------------------------------%
452 noHayAlgunoMenosImportante(X,L):-
453     hayAlgunoMenosImportante(X,L),
454     !,
455     fail.
456 noHayAlgunoMenosImportante(_,_).
457 %-----------------------------------------------------------------------------------------------------%
458 hayAlgunoMenosImportante(Y,[Y|Xs]):-
459     hayAlgunoMenosImportante(Y,Xs),
460     !.
461 hayAlgunoMenosImportante(Y,[X|_]):-
462     ee(X,Y),
463     !.
464 hayAlgunoMenosImportante(Y,[_|Xs]):-
465     hayAlgunoMenosImportante(Y,Xs),
466     !.
467 hayAlgunoMenosImportante(Y,X):-
468     ee(X,Y),
469     !.
470 %-----------------------------------------------------------------------------------------------------%
471 % más importantes a preservar
472 masImportantes(La,Lz):-
473     findall(X,masImportantes0(La,X), Ln),
474     aplanarTodosLista(Ln, Lp),
475     listaNoVacia(Lp),
476     eliminarRepetidosLista(Lp,Lz),
477     !.
478 masImportantes(La,Lp):-
479     aplanarTodosLista(La, Lp).
480 %-----------------------------------------------------------------------------------------------------%
481 masImportantes0(La,Lz):-
482     unificarSubconjunto(La,Ln),
483     masImportante(Ln, Lp),
484     borrarTodos(Lp,Ln,Lz).
485 %-----------------------------------------------------------------------------------------------------%
486 masImportante(L,X):-
487     unificarSubconjunto(L,X),
488     noHayAlgunoMasImportante(X,L).
489 %-----------------------------------------------------------------------------------------------------%
490 noHayAlgunoMasImportante(X,L):-
491     hayAlgunoMasImportante(X,L),
492     !,
493     fail.
494 noHayAlgunoMasImportante(_,_).
495 %-----------------------------------------------------------------------------------------------------%
496 hayAlgunoMasImportante(Y,[Y|Xs]):-
497     hayAlgunoMasImportante(Y,Xs),
498     !.
499 hayAlgunoMasImportante(Y,[Y|_]):-
500     ee(Y,Y),
501     !,
502     fail.
503 hayAlgunoMasImportante(Y,[X|_]):-
504     ee(Y,X),
505     !.
506 hayAlgunoMasImportante(Y,[_|Xs]):-
```

```
507     hayAlgunoMasImportante(Y,Xs),
508     !.
509 hayAlgunoMasImportante(Y,X):-
510     ee(Y,X),
511     !.
512 %-------------------------------------------------------------------------------------------------------%
513                             % ---------- ELIMINAR CONOCIMIENTO DE LA BC ---------- %
514 eliminarDeBC([X|Xs]):-
515     kb(X),
516     retract(kb(X)),
517     retractall(ee(_,X)),
518     retractall(ee(X,_)),
519     eliminarDeBC(Xs),
520     !.
521 eliminarDeBC(X):-
522     kb(X),
523     retract(kb(X)),
524     retractall(ee(_,X)),
525     retractall(ee(X,_)),
526     !.
527 eliminarDeBC(_).
528 %-------------------------------------------------------------------------------------------------------%
529                                     % ---------- FNC ---------- %
530 fnc(X,Z):-
531     normalizarDisjuncion(X,X1),
532     agregarParentesis(X1,X2),
533     formaConj(X2,X3),
534     aplanarConj(X3,X4),
535     eliminarRedundancias(X4,X5),
536     eliminarRedundancias(X5,X6),
537     eliminarTautologias(X6,Z).
538 %-------------------------------------------------------------------------------------------------------%
539 %Transforma una sentencia utilizando reglas de equivalencias
540 normalizarDisjuncion(X -> Y,Z):-
541     normalizarDisjuncion(X, X1),
542     normalizarDisjuncion(Y,Y1),
543     normalizarDisjuncion(no(X1) -o- Y1,Z),
544     !.
545 normalizarDisjuncion(no(no(X)),Z):-
546     normalizarDisjuncion(X,Z),
547     !.
548 normalizarDisjuncion(no(X -o- Y),Z):-
549     normalizarDisjuncion(X, X1),
550     normalizarDisjuncion(Y,Y1),
551     normalizarDisjuncion(no(X1) -y- no(Y1),Z),
552     !.
553 normalizarDisjuncion(no(X -y- Y),Z):-
554     normalizarDisjuncion(X, X1),
555     normalizarDisjuncion(Y,Y1),
556     normalizarDisjuncion(no(X1) -o- no(Y1),Z),
557     !.
558 normalizarDisjuncion(no(X -> Y),Z):-
559     normalizarDisjuncion(X, X1),
560     normalizarDisjuncion(Y,Y1),
561     normalizarDisjuncion(X1 -y- no(Y1),Z),
562     !.
563 normalizarDisjuncion(no(no(X)),Z):-
564     normalizarDisjuncion(X,Z),
565     !.
566 normalizarDisjuncion(X -o- (Y),Z):-
567     normalizarDisjuncion(X, X1),
568     normalizarDisjuncion(Y,Y1),
569     unirDisj(X1,Y1, Z),
570     !.
571 normalizarDisjuncion(X -o- Y,Z):-
```

```
572     normalizarDisjuncion(X, X1),
573     normalizarDisjuncion(Y,Y1),
574     unirDisj(X1,Y1, Z),
575     !.
576 normalizarDisjuncion(X -y- Y,Z):-
577     normalizarDisjuncion(X, X1),
578     normalizarDisjuncion(Y,Y1),
579     unirConj(X1,Y1, Z),
580     !.
581 normalizarDisjuncion(X,X).
582 %-------------------------------------------------------------------------------------------------%
583 %Utilizado para mantener la forma conjuntiva en la expresion
584 formaConj([X] -y- [Y],Z):-
585     formaConj(X ,X1) ,
586     formaConj(Y ,Y1) ,
587     unirConj(X1,Y1, Z),
588     !.
589 formaConj(true -o- Z, Z1):-
590     formaConj(Z ,Z1) ,
591     !.
592 formaConj(Y -o- X, W1):-
593     formaConj(X ,X1),
594     formaConj(Y ,Y1),
595     normalizarConj2(Y1, X1, W),
596     formaConj(W ,W1) ,
597     !.
598 formaConj(X,[X]):-
599     atom(X),
600     !.
601 formaConj(X,X).
602 %-------------------------------------------------------------------------------------------------%
603 normalizarConj2(Y1 -y- Y2 , X, Z1 -y- Z2):-
604     normalizarConj2(Y1,X,Z1),
605     normalizarConj2(Y2,X,Z2),
606     !.
607 normalizarConj2(X, Y1 -y- Y2 , Z1 -y- Z2):-
608     normalizarConj2(Y1,X,Z1),
609     normalizarConj2(Y2,X,Z2),
610     !.
611 normalizarConj2(Y, X, [X -o- Y]).
612 %-------------------------------------------------------------------------------------------------%
613 esConj(_ -y- _).
614 %-------------------------------------------------------------------------------------------------%
615 %Elimina las expresiones redundantes
616 eliminarRedundancias(X -y- [Y],Z):-
617     eliminarRedundancias(X,X1),
618     eliminarRedundancias(Y,Y1),
619     unirConj(X1,[Y1], Z),
620     !.
621 eliminarRedundancias(X -o- Y,Z):-
622     eliminarRepetidos(X -o- Y,L1),
623     eliminarComplementarios(L1,Z),
624     !.
625 eliminarRedundancias([X],[X1]):-
626     eliminarRedundancias(X,X1),
627     !.
628 eliminarRedundancias(X, X).
629 %-------------------------------------------------------------------------------------------------%
630 %Elimina las expresiones repetidas
631 eliminarRepetidos(X -o- X,X):-
632     !.
633 eliminarRepetidos(Y -o- X,Z -o- X):-
634     tieneRepetido(X,Y),
635     sinX(X,Y,L1),
636     eliminarRepetidos(L1,Z),
```

```
637     !.
638 eliminarRepetidos(Y -o- X,Z -o- X):-
639     eliminarRepetidos(Y,Z),
640     !.
641 eliminarRepetidos(X , X).
642 %-----------------------------------------------------------------------------------------------------%
643 tieneRepetido(X, X):-
644     !.
645 tieneRepetido(X, _ -o- X):-
646     !.
647 tieneRepetido(X, Y -o- _):-
648     tieneRepetido(X,Y).
649 %-----------------------------------------------------------------------------------------------------%
650 sinX(X, X -o- Y, Y):-
651     !.
652 sinX(X, Y -o- X, Z):-
653     sinX(X, Y, Z),
654     !.
655 sinX(X, Y -o- W, Z -o- W):-
656     sinX(X, Y, Z),
657     !.
658 sinX(_, Z, Z).
659 %-----------------------------------------------------------------------------------------------------%
660 eliminarComplementarios(X -o- X1, true):-
661     complementarios(X,X1),!.
662 eliminarComplementarios(Y -o- X,Z-o- true):-
663     tieneComplementario(X,Y),
664     sinComplementario(X,Y,L1),
665     eliminarComplementarios(L1,Z ),
666     !.
667 eliminarComplementarios(Y -o- X,Z -o- X):-
668     eliminarComplementarios(Y,Z),
669     !.
670 eliminarComplementarios(true -o- true, true):-
671     !.
672 eliminarComplementarios(X , X).
673 %-----------------------------------------------------------------------------------------------------%
674 complementarios(no(X),X).
675 complementarios(X,no(X)).
676 %-----------------------------------------------------------------------------------------------------%
677 tieneComplementario(X, X1):-
678     complementarios(X,X1),
679     !.
680 tieneComplementario(X, _ -o- X1):-
681     complementarios(X,X1),
682     !.
683 tieneComplementario(X, Y -o- _):-
684     tieneComplementario(X,Y).
685 %-----------------------------------------------------------------------------------------------------%
686 sinComplementario(X, X1 -o- Y, Y):-
687     complementarios(X,X1),
688     !.
689 sinComplementario(X, Y -o- X1, Z):-
690     complementarios(X,X1),
691     sinComplementario(X, Y, Z),
692     !.
693 sinComplementario(X, Y -o- W, Z -o- W):-
694     sinComplementario(X, Y, Z),!.
695 sinComplementario(_, Z, Z).
696 %-----------------------------------------------------------------------------------------------------%
697 %Elimina las tautologias
698 eliminarTautologias(X -y- [Y],X1):-
699     esTautologia(Y),
700     eliminarTautologias(X,X1),
701     !.
```

```
eliminarTautologias(X -y- [Y],X1 -y- [Y]):-
    eliminarTautologias(X,X1),!.
eliminarTautologias(X,X).
%-------------------------------------------------------------------------------------------------%
%Verifica si una construccion logica es siempre verdadera
esTautologia(_ -o- X):-
    esTautologia(X),
    !.
esTautologia(X -o- _):-
    esTautologia(X),
    !.
esTautologia(true).
%-------------------------------------------------------------------------------------------------%
agregarParentesis(true, true):-
    !.
agregarParentesis(X, C2 -o- (C1)):-
    ultimoConj(X,B1,C1),
    agregarParentesis(B1,C2),
    !.
agregarParentesis(X,(X)):-
    ultimoConj(X,true,X),
    !.
%-------------------------------------------------------------------------------------------------%
ultimoConj(X -o- Y, X, Y):-
    !.
ultimoConj(X -y- Y, X1, Y1 -y- Y):-
    ultimoConj(X, X1, Y1),
    !.
ultimoConj(X ,true, X).
%-------------------------------------------------------------------------------------------------%
                                % ---------- PREDICADOS PARA APLANAR ---------- %
aplanar([[X]], [X1]):-
    aplanar(X,X1),
    !.
aplanar([X], [X1]):-
    aplanar(X,X1),
    !.
aplanar(X -y- Y, Z):-
    aplanar(X,X1),
    aplanar(Y,Y1),
    unirConj(X1,Y1, Z),
    !.
aplanar(X -o- Y,Z):-
    aplanar(X,X1),
    aplanar(Y,Y1),
    unirDisj(X1,Y1, Z),
    !.
aplanar(X, X).
%-------------------------------------------------------------------------------------------------%
aplanarConj(X -y- Y, Z):-
    aplanarDisj(X,X1),
    aplanarDisj(Y,Y1),
    aplanarConj(X1,X2),
    aplanarConj(Y1,Y2),
    unirConj(X2,Y2, Z),
    !.
aplanarConj(X, X1):-
    aplanarDisj(X,X1).
%-------------------------------------------------------------------------------------------------%
aplanarDisj([[X] -o- [Y]], [Z]):-
    aplanarDisj(X,X1),
    aplanarDisj(Y,Y1),
    unirDisj(X1,Y1, Z),
    !.
aplanarDisj([X -o- [Y]], [Z]):-
```

```
767     aplanarDisj(X,X1),
768     aplanarDisj(Y,Y1),
769     unirDisj(X1,Y1, Z),
770     !.
771 aplanarDisj([[X] -o- Y], [Z]):-
772     aplanarDisj(X,X1),
773     aplanarDisj(Y,Y1),
774     unirDisj(X1,Y1, Z),
775     !.
776 aplanarDisj([X -o- Y], [Z]):-
777     aplanarDisj(X,X1),
778     aplanarDisj(Y,Y1),
779     unirDisj(X1,Y1, Z),
780     !.
781 aplanarDisj(X -o- [Y], Z):-
782     aplanarDisj(X,X1),
783     aplanarDisj(Y,Y1),
784     unirDisj(X1,Y1, Z),
785     !.
786 aplanarDisj([X] -o- Y, Z):-
787     aplanarDisj(X,X1),
788     aplanarDisj(Y,Y1),
789     unirDisj(X1,Y1, Z),
790     !.
791 aplanarDisj(X, X).
792 %-------------------------------------------------------------------------------------------%
793 unirConj(X, Y1 -y- Y2,  Z -y- Y2):-
794     unirConj(X, Y1, Z),
795     !.
796 unirConj(X, Y1 -o- Y2,  Z):-
797     unirConj(X, Y1, Z1),
798     unirDisj(Z1,Y2, Z),
799     !.
800 unirConj([],X,X):-
801     !.
802 unirConj(X,[],X):-
803     !.
804 unirConj(X, Y,  X -y- Y):-
805     !.
806 %-------------------------------------------------------------------------------------------%
807 unirDisj(X, Y1 -o- Y2,  Z -o- Y2):-
808     unirDisj(X, Y1, Z),
809     !.
810 unirDisj(X, Y1 -y- Y2,  Z):-
811     unirDisj(X, Y1, Z1), unirConj(Z1,Y2, Z),
812     !.
813 unirDisj([],X,X):-
814     !.
815 unirDisj(X,[],X):-
816     !.
817 unirDisj(X, Y,  X -o- Y):-
818     !.
819 %-------------------------------------------------------------------------------------------%
820 %Borra la primer ocurrencia de un elemento dentro de una lista
821 borrarUn(_,[],[]):-
822     !.
823 borrarUn(X,[X|Xs],Xs):-
824     !.
825 borrarUn(X,[Y|Xs],[Y|Z]):-
826     borrarUn(X,Xs,Z).
827
828 borrarTodos([],Y,Y):-!.
829 borrarTodos([X|Xs],Y,Z):-
830     borrarUn(X,Y,Zo),
831     borrarTodos(Xs,Zo,Z),
```

```
832     !.
833 borrarTodos(X,Y,Z):-
834     borrarUn(X,Y,Z).
835 %-------------------------------------------------------------------------------------------------%
836 aplanarTodosLista([],[]):-
837     !.
838 aplanarTodosLista([X|Xs],Y):-
839     aplanarTodosLista(Xs,Ys),
840     concatenarListas(X,Ys,Y),
841     !.
842 aplanarTodosLista(X,X).
843 %-------------------------------------------------------------------------------------------------%
844 % concatena 2 listas
845 concatenarListas([],L,L):-
846     !.
847 concatenarListas([X|Xs],Ys,[X|Zs]):-
848     concatenarListas(Xs,Ys,Zs).
849 %-------------------------------------------------------------------------------------------------%
850                                 % ---------- PHP  ---------- %
851 imprimirPorPHP :- mostrarBC(KB,EE), write('KB= '),write(KB),write('
852 EE= '),write(EE), write('
853
854 ').
855
856 expandirPorPHP(Conocimiento,'','') :-
857     expandirBC(Conocimiento,[[],[]]).
858
859 expandirPorPHP(Conocimiento,'',MenImpQue) :-
860     expandirBC(Conocimiento,[[],MenImpQue]).
861
862 expandirPorPHP(Conocimiento,TanImpComo,'') :-
863     expandirBC(Conocimiento,[TanImpComo,[]]).
864
865 expandirPorPHP(Conocimiento,TanImpComo,MenImpQue) :-
866     expandirBC(Conocimiento,[TanImpComo,MenImpQue]).
867
868 opcionesContraccionPHP(A) :-
869     opcionesContraccion(A, Emi, Cmi, Emc).
870 %   write('Eliminar por menos importante= '),
871 %   write(Emi),
872 %   write('
873
874 '),
875 %   write('Eliminar conservando mas importantes= '),
876 %   write(Cmi),
877 %   write('
878
879 '),
880 %   write('Eliminar por minima cantidad= '),
881 %   write(Emc),
882 %   write('
883
884 ').
885
886 contraerPorMenosImportantePHP(A):-
887     contraccionPorMenosImportantes(A).
888
889 contraerPorMantenerMasImportantesPHP(A):-
890     contraccionPorMantenerMasImportantes(A).
891
892 contraerPorMinimaCantidadPHP(A):-
893     contraccionPorMinimaCantidad(A).
894
895 opcionesRevisarPHP(A) :-
896     opcionesRevision(A, Rmi, Rmai, Rmc).
```

```
%   write('Revisar por menos importante= '),
%   write(Rmi),
%   write('

'),
%   write('Revisar por mas importantes= '),
%   write(Rmai),
%   write('

'),
%   write('Revisar por minima cantidad= '),
%   write(Rmc),
%   write('

').

revisarPorMenosImportantePHP(A):-
    revisarPorMenosImportantes(A).

revisarPorMantenerMasImportantesPHP(A):-
    revisarPorMantenerMasImportantes(A).

revisarPorMinimaCantidadPHP(A):-
    revisarPorMinimaCantidad(A).

opcionesConsolidarPHP :-
    opcionesConsolidar(ElimMenImp, ConsMasImp, ElimMinCant).
%   write('Consolidar por menos importante= '),
%   write(ElimMenImp),
%   write('

'),
%   write('Consolidar conservando mas importantes= '),
%   write(ConsMasImp),
%   write('

'),
%   write('Consolidar por minima cantidad= '),
%   write(ElimMinCant),
%   write('

').

consolidarPorMenosImportantePHP:-
    consolidarPorMenosImportantes.

consolidarPorMantenerMasImportantesPHP:-
    consolidarPorMantenerMasImportantes.

consolidarPorMinimaCantidadPHP:-
    consolidarPorMinimaCantidad.
```