```
%---------------------------------- - - - - - - - - - -  -  -  -
-   -

% SIST. REPR. CONHECIMENTO E RACIOCINIO - MiEI/3


%---------------------------------- - - - - - - - - - -  -  -  -
-   -

% TP1 - Programacao em logica e invariantes

%---------------------------------- - - - - - - - - - -  -  -  -
-   -

% SICStus PROLOG: Declaracoes iniciais


:- set_prolog_flag( discontiguous_warnings,off ).

:- set_prolog_flag( single_var_warnings,off ).

:- set_prolog_flag( unknown,fail ).



%---------------------------------- - - - - - - - - - -  -  -  -
-   -

% SICStus PROLOG: definicoes iniciais


:- op( 900,xfy,'::' ).

:- dynamic utente/4.

:- dynamic prestador/4.

:- dynamic cuidado/7.

:- dynamic '-'/1.


%---------------------------------- - - - - - - - - - -  -  -  -
-   -

% Extensao do predicado utente: IdUt, Nome, Idade, Morada ->
{V,F}


-utente(1,maria,20,morada('Rua do
Louro','Caldelas','Guimaraes')).


utente( 1,ana,20,morada( 'Rua do Louro','Caldelas','Guimaraes' )
).
```

```
excecao( utente( 2,bruno,A,morada( 'Rua do Louro' , 'Caldelas' ,
'Guimaraes' ) ) ) :- A >= 25,

                                                    A =< 35.


utente( 3,carlos,xpto497,xpto336 ).

nuloD( xpto497 ).

excecao( utente( A,C,_,_ ) ) :- utente( A,C,xpto497,xpto336 ).

nuloI( xpto336 ).


+utente( O,B,C,D ) :: ( solucoes( N,( utente( 3,N,X,T ),nao(
nuloI( T ) ) ),L ),
                          comprimento( L,Aux ),
                          Aux == 0
                          ).


+(-utente( O,B,C,D )) :: ( solucoes( N,( -utente( 3,N,X,T ),nao(
nuloI( T ) ) ),L ),
                          comprimento( L,Aux ),
                          Aux == 0
                          ).



utente( 4,duarte,35,morada( 'Rua dos Loiros' , 'Caldelas' ,
'Guimaraes' ) ).



utente( 5,elisabete,26,morada( 'Rua da Ajuda' , 'Vila Nova' ,
'Guimaraes' ) ).


utente( 6,filipa,xpto001,morada('Rua do
Emigrante','Azurem','Braga') ).

nuloD( xpto001 ).
```

```
excecao( utente(A,B,_,D) ) :- utente( A,B,xpto001,D ).



utente( 7,gisela,33,xpto002 ).

nuloD( xpto002 ).

excecao( utente(A,B,C,_) ) :- utente(A,B,C,xpto002).



excecao( utente( 8,helder,Idade,morada('Rua do
Azevinho','Braga','Braga') ) ) :-

      Idade >= 15,

      Idade =< 18.


excecao(utente(9,irene,30,morada('Rua dos
tolos','Briteiros','Guimaraes'))).

excecao(utente(9,irene,30,morada('Rua dos
tolos','S.Clemente','Guimaraes'))).

excecao(utente(9,irene,50,morada('Rua dos
tolos','Briteiros','Guimaraes'))).

excecao(utente(9,irene,50,morada('Rua dos
tolos','S.Clemente','Guimaraes'))).


utente(10,jacinto,48,morada('Rua da Rua','Braga','Braga')).


excecao(utente(11,marta,34,morada('Rua do Pinheiro','Sao
Lourenco','Braga'))).

excecao(utente(11,marta,34,morada('Rua do Pinheiro','Sao
Lourenco','Guimaraes'))).

excecao(utente(11,mara,34,morada('Rua do Pinheiro','Sao
Lourenco','Braga'))).

excecao(utente(11,mara,34,morada('Rua do Pinheiro','Sao
Lourenco','Guimaraes'))).


excecao( utente( 12,joaquim,I,morada( 'Rua do Limoeiro' ,
'Amais' , 'Viana do Castelo' ) ) ) :-

      I >= 60,
```

```
      I =< 80.


utente( 13,marcelo,45,xpto115 ).

excecao( utente( ID,N,I,L ) ) :- utente( ID,N,I,xpto115 ).

nuloI(xpto115).


+utente( ID,N,I,L ) :: ( solucoes( ID,( utente( 13,N,I,M ),nao(
nuloI(M) ) ),R ),
                                  comprimento( R,Aux ),
                                  Aux==0
                                  ).


+(-utente( ID,N,I,L )) :: ( solucoes( ID,( -utente( 13,N,I,M
),nao( nuloI(M) ) ),R ),
                                  comprimento( R,Aux ),
                                  Aux==0
                                  ).


excecao( utente( 14,diana,I,morada( 'Rua dos Loiros' ,
'Caldelas' , 'Guimaraes' ) ) ) :-
      I >= 8,
      I =< 12.




-utente(ID,N,I,M) :-
      nao( utente(ID,N,I,M) ),
      nao( excecao( utente( ID,N,I,M ) ) ).



%--------------------------------- - - - - - - - - - -  -  -  -
-   -

% Extensao do predicado prestador: IdPrest, Nome, Especialidade,
Instituicao -> {V,F}
```

```
-prestador( 1,peixe,'medicina interna','Hospital de Guimaraes'
).



prestador( 1,antonio,urologia,'Hospital de Guimaraes' ).



prestador( 2,bernardo,ortopedia,'Hospital Privado de Guimaraes'
).



prestador( 3,carla,xpto789,'Hospital de Guimaraes' ).

nuloD( xpto789 ).

excecao( prestador( A,B,_,D ) ) :- prestador( A,B,xpto789,D ).



prestador( 4,dalila,neurologia,xpto123 ).

nuloI( xpto123 ).

excecao( prestador( A,B,C,_ ) ) :- prestador( A,B,C,xpto123 ).



+prestador( O,B,C,D ) :: ( solucoes( N,( prestador( 4,N,X,T
),nao( nuloI( T ) ) ),L ),
                           comprimento( L,Aux ),
                           Aux == 0
                           ).


+(-prestador( O,B,C,D )) :: ( solucoes( N,( -prestador( 4,N,X,T
),nao( nuloI( T ) ) ),L ),
                           comprimento( L,Aux ),
                           Aux == 0
                           ).
```

```
prestador( 5,ermelinda,enfermeira,'Hospital de Braga' ).



excecao( prestador( 6,fausto,neurologia,'Hospital Privado de
Braga' ) ).
excecao( prestador( 6,fausto,neurologia,'Hospital de Braga' ) ).



excecao( prestador( 7,gabriel,xpto456,'Hospital de Braga' ) ).
excecao( prestador( 7,gabriel,xpto456,'Hospital de Guimaraes' )
).
nuloI(xpto456).
prestador(7,gabriel,xpto456,'Hospital de Braga').
prestador(7,gabriel,xpto456,'Hospital de Guimaraes').
excecao( prestador( A,B,_,D ) ) :-   prestador( A,B,xpto456,D).


+prestador( O,B,C,D ) :: ( solucoes( N,( prestador( 7,N,X,T
),nao( nuloI( X ) ) ),L ),
                           comprimento( L,Aux ),
                           Aux == 0
                           ).


+(-prestador( O,B,C,D )) :: ( solucoes( N,( -prestador( 7,N,X,T
),nao( nuloI( X ) ) ),L ),
                           comprimento( L,Aux ),
                           Aux == 0
                           ).



prestador( 8,henriqueta,cardiologia,'Hospital de Braga' ).
```

```
excecao(prestador(9,iglesias,urologia,'Hospital Privado de
Braga')).

excecao(prestador(9,iglesias,patologia,'Hospital Privado de
Braga')).


prestador(10,josefina,patologia,xpto423).

excecao(prestador(A,B,C,_)) :- prestador(A,B,C,xpto423).


prestador(11,nuria,dermatologia,'Hospital do Porto').



prestador( 12,joao,xpto171,'Hospital do Porto' ).

excecao( prestador( ID,N,_,L ) ) :- prestador( ID,N,xpto171,L ).

nuloI(xpto171).


+prestador( ID,N,E,L ) :: ( solucoes( ID,( prestador( 12,N,I,L
),nao( nuloI(I) ) ),R ),
                                    comprimento( R,Aux ),
                                    Aux==0
                                ).



excecao( prestador( 13,julia,neurocirurgia,'Hospital do Porto' )
).

excecao( prestador( 13,julia,neurocirurgia,'Hospital de
Guimaraes' ) ).

excecao( prestador( 13,julia,neurologia,'Hospital do Porto' ) ).

excecao( prestador( 13,julia,neurologia,'Hospital de Guimaraes'
) ).



prestador( 14,renato,xpto145,xpto167 ).

excecao( prestador( ID,N,_,_ ) ) :- prestador(
ID,N,xpto145,xpto167 ).
```

```
-prestador(ID,N,E,I) :-
      nao( prestador(ID,N,E,I) ),
      nao( excecao( prestador( ID,N,E,I ) ) ).


%-------------------------------- - - - - - - - - - -  -  -  -
-   -

% Extensao do predicado cuidado: Data, IdUt, IdPrest, Descricao,
Custo, Instituicao -> {V,F}


-cuidado(1,data( 2,1,2018,20 ),1,5,'curativo',10,'Hospital de
Braga' ).


excecao( cuidado( 1,D,1,5,'curativo',10,'Hospital de Braga' ) )
:-

      comparaData( >=,D,data( 1,1,2018,20 ) ),

      comparaData( <=,D,data( 5,1,2018,20 ) ).



cuidado( 2,data( 2,1,2018,20
),2,6,'investigacao',xpto444,'Hospital Privado de Braga' ).

cuidado( 2,data( 2,1,2018,20
),2,6,'investigacao',xpto444,'Hospital de Braga' ).

nuloI( xpto444 ).

excecao( cuidado( 2,data( 2,1,2018,20
),2,6,'investigacao',xpto444,'Hospital Privado de Braga' ) ).

excecao( cuidado( 2,data( 2,1,2018,20
),2,6,'investigacao',xpto444,'Hospital de Braga') ).

excecao( cuidado( A,B,C,D,E,_,G ) ) :- cuidado(
A,B,C,D,E,xpto444,G ).


%-------- Invariante de nulo interdito ---------------

+cuidado( O,A,B,C,D,E,F ) :: ( solucoes( N,( cuidado(
2,N,_,T,_,X,_ ),nao( nuloI( X ) ) ),L ),
                              comprimento( L,Aux ),
```

```
                            Aux == 0

                            ).


%-------- Invariante de nulo interdito ---------------

+(-cuidado( O,A,B,C,D,E,F )) :: ( solucoes( N,( -cuidado(
2,N,_,T,_,X,_ ),nao( nuloI( X ) ) ),L ),

                            comprimento( L,Aux ),

                            Aux == 0

                            ).



cuidado( 3,data( 1,2,2018,20 ),3,7,xpto908,50,'Hospital de
Braga' ).

cuidado( 3,data( 1,2,2018,20 ),3,7,xpto908,50,'Hospital de
Guimaraes' ).

nuloD( xpto908 ).

excecao( cuidado( A,L,C,D,E,F,G ) ) :- cuidado(
A,L,C,D,xpto908,F,G ).

excecao( cuidado( 3,data( 1,2,2018,20 ),3,7,xpto908,50,'Hospital
de Braga' ) ).

excecao( cuidado( 3,data( 1,2,2018,20 ),3,7,xpto908,50,'Hospital
de Guimaraes') ).



cuidado( 4,data( 2,2,2018,20 ),4,8,xpto007,15489,'Hospital de
Braga' ).

nuloI( xpto007 ).

excecao( cuidado( A,B,C,D,_,F,G ) ) :- cuidado(
A,B,C,D,xpto007,F,G ).


%-------- Invariante de nulo interdito ---------------

+cuidado( O,A,B,C,D,E,F ) :: ( solucoes( N,( cuidado(
4,N,_,T,X,_,_ ),nao( nuloI( X ) ) ),L ),

                            comprimento( L,Aux ),

                            Aux == 0

                            ).
```

```
+(-cuidado( O,A,B,C,D,E,F )) :: ( solucoes( N,( -cuidado(
4,N,_,T,X,_,_ ),nao( nuloI( X ) ) ),L ),
                                comprimento( L,Aux ),
                                Aux == 0
                                ).



excecao( cuidado( 5,data( 3,3,2018,20
),5,9,'rotina',25,'Hospital Privado de Braga' ) ).

excecao( cuidado( 5,data( 3,3,2018,20 ),5,9,'exame',25,'Hospital
Privado de Braga' ) ).


cuidado( 6,data(3,4,2018,20),6,10,'medicao',70,xpto424 ).

nuloD( xpto424 ).

excecao( cuidado( A,B,C,D,E,F,_ ) ) :- cuidado(
A,B,C,D,E,F,xpto424 ).



excecao( cuidado( 7,data(4,4,2018,20),7,1,'exame',69,'Hospital
de Guimaraes' ) ).

excecao( cuidado( 7,data(5,4,2018,20),7,1,'exame',69,'Hospital
de Guimaraes' ) ).



excecao( cuidado(
8,data(1,1,2018,20),8,2,'cirurgia',Custo,'Hospital Privado de
Guimaraes' ) ) :-
     Custo >= 100,
     Custo =< 500.



nuloD(xpto111).

nuloD(xpto222).

cuidado(9,data(28,5,2018,20),0,3,xpto111,xpto222,'Hospital de
Guimares').
```

```
excecao(cuidado(A,B,C,D,_,_,G)) :-
cuidado(A,B,C,D,xpto111,xpto222,G).



nuloI(xpto400).

nuloI(xpto123).

cuidado(10,xpto400,10,4,rotina,333,xpto123).

excecao(cuidado(A,_,C,D,E,F,_)) :-
cuidado(A,xpto400,C,D,E,F,xpto123).


+cuidado( O,A,B,C,D,E,F ) :: ( solucoes( N,( cuidado(
10,X,_,_,_,_,Y ),nao( nuloI( X ) ),nao(nuloI(Y) ) ),L ),
                            comprimento(L,Aux),
                            Aux == 0
                            ).


+(-cuidado( O,A,B,C,D,E,F )) :: ( solucoes( N,( -cuidado(
10,X,_,_,_,_,Y ),nao( nuloI( X ) ),nao(nuloI(Y) ) ),L ),
                            comprimento(L,Aux),
                            Aux == 0
                            ).


nuloD(xpto113).

cuidado(11,data(20,5,2018,20),11,12,cirurgia,100,xpto113).

exececao(cuidado(A,B,C,D,E,F,_)) :-
cuidado(A,B,C,D,E,F,xpto113).



cuidado( 12,data( 1,6,2018,20 ),12,14,'exame',50,xpto168 ).

excecao( cuidado( ID,D,U,P,Desc,C,_ ) ) :- cuidado(
ID,D,U,P,Desc,C,xpto168 ).

nuloD(xpto168).
```

```
cuidado( 13,data( 1,7,2018,20 ),13,12,'rotina',10,'Hospital do
Porto' ).


cuidado( 14,data( 4,7,2018,20 ),14,11,xpto112,xpto999,'Hospital
do Porto' ).

excecao( cuidado( ID,D,U,P,_,_,I ) ) :- cuidado(
ID,D,U,P,xpto112,xpto999,I ).

nuloI(xpto112).

nuloD(xpto999).


+cuidado( ID,D,U,P,Desc,C,I ) :: ( solucoes( ID,( cuidado(
14,D,U,P,Des,C,I ),nao( nuloI(Des) ) ),R ),
                                                comprimento( R,N
),
                                                N==0
                                            ).



-cuidado(ID,D,U,P,Desc,C,I) :-
      nao( cuidado(ID,D,U,P,Desc,C,I) ),
      nao( excecao( cuidado(ID,D,U,P,Desc,C,I) ) ).



%----------------------------------Data do cuidado é válida-----
----------------------------------


+cuidado( Data,U,P,Descricao,C,I ) :: (dataValida(Data)).



%----------------- ----- --- -- -
% Extensao do predicado dataValida: Data -> {V,F}
dataValida( data(A,M,D,H) ) :-  natural(A),
                                        natural(M),
                                        natural(D),
                                        natural(H),
```

```
                                        H < 25,
                                        M < 13,
                                        diasValidos( A,M,D ).

diasValidos( A,M,D ) :- R is M mod 2,
                                        R \= 0,
                                        D < 32,
                                        D > 0.
diasValidos( A,M,D ) :- R is M mod 2,
                                        R == 0,
                                        M \= 2,
                                        D < 30,
                                        D > 0.
diasValidos( A,2,D ) :- R is A mod 4,
                                        R \= 0,
                                        D < 29,
                                        D > 0.
diasValidos( A,2,D ) :- R is A mod 4,
                                        R == 0,
                                        D < 30,
                                        D > 0.

%----------------------------------Nao pode haver mais do que 3
cuidados à mesma hora tanto para o utente como o profissional---
--------------

+cuidado(Id,D,U,P,Des,C,I) ::
(solucoes((D,P),cuidado(Id,D,Ut,P,Descr,Cus,Ins),L),
                                        comprimento(L,X),
                                        X=<3
                                        ).

+cuidado(Id,D,U,P,Des,C,I) ::
(solucoes((D,U),cuidado(Id,D,U,Pr,Descr,Cus,Ins),L),
```

```
                                        comprimento(L,X),
                                        X=<3
                                        ).


%-------- Nao podem haver ids repetidos ------------


+utente( IdU,_,_,_ ) :: (solucoes( IdU,( utente( IdU,A,B,C
),nao( nuloD( A ) ),nao( nuloD( B ) ),nao( nuloD( C ) ) ),L ),
                                        comprimento( L,X ),
                                        X =< 1).


+prestador( IdP,_,_,_ ) :: (solucoes( IdP,( prestador( IdP,A,B,C
),nao( nuloD( A ) ),nao( nuloD( B ) ),nao( nuloD( C ) ) ),L ),
                                        comprimento( L,X ),
                                        X =< 1).



+cuidado( IdC,_,_,_,_,_,_ ) :: (solucoes( IdP,( cuidado(
IdC,A,B,C,D,E,F ),nao( nuloD( A ) ),nao( nuloD( B ) ),nao(
nuloD( C ) ),nao( nuloD( D ) ),nao( nuloD( E ) ),nao( nuloD( F )
) ),L ),
                                        comprimento( L,X ),
                                        X =< 1).



%--------- Nao pode ter uma idade inválida ----------
+utente( IdU,_,I,_ ) :: ( validaIdade( I ) ).


%--------------- --- -- -
% Extensao do predicado validaIdade: Idade -> {V,F}
validaIdade( A ) :- R is A+1,
                            natural(R).
```

```
%--------------- Nao se podem adicionar cuidados para os quais
nao existam utentes/prestadores -------------------


+cuidado( _,_,IdU,_,_,_,_ ) :: (solucoes( IdU,demonstrador(
utente( IdU,A,_,_ )+excecao( utente( IdU,_,_,_ ) ),verdadeiro
),L ),
                                comprimento( L,X ),
                                X >= 1).


+cuidado( _,_,_,IdP,_,_,_ ) :: (solucoes( IdP,demonstrador(
prestador( IdP,A,_,_ )+excecao( prestador( IdP,_,_,_ )
),verdadeiro ),L ),
                                comprimento( L,X ),
                                X >= 1).


%--------------- Nao se pode adicionar um cuidado cujo custo
seja negativo ------------------------


+cuidado( _,_,_,_,_,C,_ ) :: (C >= 0).


%----------------- Nao se pode adicionar um cuidado com os
campos todos iguais --------------------

+cuidado( Id,D,IdU,IdP,Desc,C,I ) :: (solucoes( LI,cuidado(
Id,D,IdU,IdP,Desc,C,I ),L ),
                                comprimento( L,X ),
                                X == 1).


%--------------- Nao se pode remover um utente/prestador para o
qual existam cuidados relativos ---------------


-utente( IdU,_,_,_ ) :: (solucoes( D,cuidado(_,D,IdU,_,_,_,_ ),L
),
                                comprimento( L,X ),
                                X == 0).
```

```
-prestador( IdP,_,_,_ ) :: (solucoes( D,cuidado(_,D,_,IdP,_,_,_
),L ),
                                comprimento( L,X ),
                                X == 0).


%-------------------- Invariantes de conhecimento imperfeito ---
-------------------------------------


% ----- Nao pode haver conhecimento negativo igual aquele
cuidado se queremos adicionar positivo -------------

+utente( Id,No,I,M ) :: (solucoes( A,-utente( Id,No,I,M ),L ),
                                                comprimento( L,N
),
                                                N == 0).


% ----- Nao pode haver conhecimento positivo igual aquele utente
se queremos adicionar negativo -------------

+( -utente( Id,No,I,M ) ) :: (solucoes( A,utente( Id,No,I,M ),L
),
                                                comprimento( L,N
),
                                                N == 0).


% ----- Nao pode haver conhecimento negativo igual aquele utente
se queremos adicionar negativo -------------

+( -utente( Id,No,I,M ) ) :: (solucoes( A,-utente( Id,No,I,M ),L
),
                                                comprimento( L,N
),
                                                N =< 2). % -----
aqui é menor ou igual a 2 pois ele encontra sempre o do fecho
transitivo
```

```
% ----- Nao pode haver conhecimento negativo igual aquele
cuidado se queremos adicionar positivo -------------

+prestador( Id,No,I,M ) :: (solucoes( A,-prestador( Id,No,I,M
),L ),
                                                    comprimento( L,N
),
                                                    N == 0).


% ----- Nao pode haver conhecimento positivo igual aquele
prestador se queremos adicionar negativo -------------

+( -prestador( Id,No,I,M ) ) :: (solucoes( A,prestador(
Id,No,I,M ),L ),
                                                    comprimento( L,N
),
                                                    N == 0).


% ----- Nao pode haver conhecimento negativo igual aquele
prestador se queremos adicionar negativo -------------

+( -prestador( Id,No,I,M ) ) :: (solucoes( A,-prestador(
Id,No,I,M ),L ),
                                                    comprimento( L,N
),
                                                    N =< 2).



% ----- Nao pode haver conhecimento negativo igual aquele
cuidado se queremos adicionar positivo -------------

+cuidado( Id,Data,U,P,D,C,I ) :: (solucoes( A,-cuidado(
Id,Data,U,P,D,C,I ),L ),
                                                    comprimento( L,N
),
                                                    N == 0).


% ----- Nao pode haver conhecimento positivo igual aquele
cuidado se queremos adicionar negativo -------------

+( -cuidado( Id,Data,U,P,D,C,I ) ) :: (solucoes( A,cuidado(
Id,Data,U,P,D,C,I ),L ),
```

```
                                                    comprimento( L,N
),
                                                    N == 0).


% ----- Nao pode haver conhecimento negativo igual aquele
cuidado se queremos adicionar negativo -------------

+( -cuidado( Id,Data,U,P,D,C,I ) ) :: (solucoes( A,-cuidado(
Id,Data,U,P,D,C,I ),L ),
                                                    comprimento( L,N
),
                                                    N =< 2).


% --------------- Penso que na remocao de prestadores e utentes
temos de ter atencao em eles estarem também em cuidados mas n se
saber se sao mm eles------------------------

-utente( A,N,I,M ) :: ( solucoes( D,( excecao( cuidado(
ID,Data,A,P,D,C,In ) ) ),L ),
                                    comprimento( L,Aux ),
                                    Aux == 0
                                    ).



% --------------- Penso que na remocao de prestadores e utentes
temos de ter atencao em eles estarem também em cuidados mas n se
saber se sao mm eles------------------------

-prestador( A,N,I,M ) :: ( solucoes( D,( excecao( cuidado(
ID,Data,U,A,D,C,In ) ) ),L ),
                                    comprimento( L,Aux ),
                                    Aux == 0
                                    ).


%------------------------- Predicados de insercao --------------
----------------------


%---------------- Adicionar utente ----------------------------
-------
```

```
% Extensao do predicado adicionarUtente: Id,Nome,Idade,Morada ->
{V,F}


adicionarUtente( IdUtente,Nome,Idade,Morada ) :-

     evolucao( utente( IdUtente,Nome,Idade,Morada ) ).



%---------------- Adicionar prestador -------------------------
----------

% Extensao do predicado adicionarPrestador:
Id,Nome,Especialidade,ListaInstituicao -> {V,F}


adicionarPrestador( IdPrestador,Nome,Especialidade,ListaI ) :-

     evolucao( prestador( IdPrestador,Nome,Especialidade,ListaI
) ).


%---------------- Adicionar cuidado ---------------------------
--------

% Extensao do predicado adicionarCuidado:
Data,IdUtente,IdPrestador,Descricao,Custo,Instituicao -> {V,F}


adicionarCuidado(
Id,Data,IdUtente,IdPrestador,Descricao,Custo,Instituicao ) :-

     evolucao( cuidado(
Id,Data,IdUtente,IdPrestador,Descricao,Custo,Instituicao ) ).


%------------------------- Predicados de remocao ---------------
--------------------


%---------------- Remover utente ------------------------------
-----

% Extensao do predicado retirarUtente: Id -> {V,F}


retirarUtente( IdUtente ) :-
```

```
                            inevolucao( utente( IdUtente,_,_,_ )
).



%---------------- Remover prestador ----------------------------
--------

% Extensao do predicado retirarPrestador: Id -> {V,F}


retirarPrestador( IdPrestador ) :-
                                        inevolucao( prestador(
IdPrestador,_,_,_ ) ).


%---------------- Remover cuidado ------------------------------
------

% Extensao do predicado retirarCuidado:
Data,IdUtente,IdPrestador,Descricao,Custo,Instituiao -> {V,F}

retirarCuidado(
Id,Data,IdUtente,IdPrestador,Descricao,Custo,Instituiao ) :-

     inevolucao( cuidado(
Id,Data,IdUtente,IdPrestador,Descricao,Custo,Instituiao ) ).




%--------------------------------- - - - - - - - - - -  -  -  -
-   -

% Extensao do meta-predicado demo: Questao,Resposta -> {V,F}


demo( Questao,verdadeiro ) :-

    Questao.

demo( Questao,falso ) :-

    -Questao.

demo( Questao,desconhecido ) :-

    nao( Questao ),

    nao( -Questao ).
```

```
demonstrador( A+B,R ) :- demonstrador( A,T ),
                                 demonstrador( B,Q ),
                                 ou( T,Q,R ).
demonstrador( A^B,R ) :- demonstrador( A,T ),
                                 demonstrador( B,Q ),
                                 e( T,Q,R ).
demonstrador( A,R ) :- demo( A,R ).

%----------- ou: Termo,Termo,Resposta -> {V,F}
ou( verdadeiro,_,verdadeiro ).
ou( _,verdadeiro,verdadeiro ).
ou( desconhecido,desconhecido,desconhecido ).
ou( desconhecido,falso,desconhecido ).
ou( falso,desconhecido,desconhecido ).
ou( falso,falso,falso ).

%----------- e: Termo,Termo,Resposta -> {V,F}
e( falso,_,falso ).
e( _,falso,falso ).
e( desconhecido,desconhecido,desconhecido ).
e( desconhecido,verdadeiro,desconhecido ).
e( verdadeiro,desconhecido,desconhecido ).
e( verdadeiro,verdadeiro,verdadeiro ).



%---------------------------- Predicados de Evolucao -----------
--------

%-------------------------------------------------
% Extensao do predicado evolucao: Termo -> {V,F}
evolucao( Termo ) :- solucoes( Inv,+Termo::Inv,S ),
                           inserir( Termo ),
```

```
                        testar( S ).

%-----------------------------------------------------
% Extensao do predicado inserir: Predicado -> {V,F}
inserir( P ) :- assert( P ).
inserir( P ) :- retract( P ), !, fail.


%-----------------------------------------------------
% Extensao do predicado involucao: Termo -> {V,F}
inevolucao( Termo ) :- solucoes(Inv,-Termo::Inv,S),
                       remover(Termo),
                       testar(S).


%-----------------------------------------------------
% Extensao do predicado remover: Predicado -> {V,F}
remover( P ) :- retract( P ).
remover( P ) :- assert( P ), !, fail.

%-----------------------------------------------------
% Extensao do predicado testar: ListaPredicado -> {V,F}
testar( [] ).
testar( [X|R] ) :- X,
                   testar( R ).

%------------------------- Predicado nao - negacao por falha na
prova ----------------
% Extensao do predicado nao: Termo -> {V,F}

nao( T ) :- T, !, fail.
nao( T ).
```

```
%-----------------------------------------------------

% Extensao do predicado solucoes:
Termo,Questao,Solucao(ListaQuestao) -> {V,F}

solucoes(X,Y,Z) :- findall(X,Y,Z).



comprimento(R,S) :- length(R,S).



%-----------------------------------------------------

% Extensao do predicado natural: Numero -> {V,F}

natural( 1 ).

natural( N ) :- R is N-1,

                R > 0,

            natural(R).



%-------------------------------------------------------------

% Extensao do predicado comparaData: Criterio(>=,<=), Data, Data
-> {V,F}


comparaData(igual,data(A1, M1, D1, H1), data(A2, M2, D2, H2)) :-

     A1 == A2,

     M1 == M2,

     D1 == D2,

     H1 == H2.

comparaData(>=,data(A1, M1, D1, H1), data(A2, M2, D2, H2)) :-

     A1 > A2.

comparaData(>=,data(A1, M1, D1, H1), data(A2, M2, D2, H2)) :-

     A1 == A2,

     M1 > M2.

comparaData(>=,data(A1, M1, D1, H1), data(A2, M2, D2, H2)) :-

     A1 == A2,
```

```
      M1 == M2,
      D1 > D2.
comparaData(>=,data(A1, M1, D1, H1), data(A2, M2, D2, H2)) :-
      A1 == A2,
      M1 == M2,
      D1 == D2,
      H1 >= H2.
comparaData(<=,data(A1, M1, D1, H1), data(A2, M2, D2, H2)) :-
      A1 < A2.
comparaData(<=,data(A1, M1, D1, H1), data(A2, M2, D2, H2)) :-
      A1 == A2,
      M1 < M2.
comparaData(<=,data(A1, M1, D1, H1), data(A2, M2, D2, H2)) :-
      A1 == A2,
      M1 == M2,
      D1 < D2.
comparaData(<=,data(A1, M1, D1, H1), data(A2, M2, D2, H2)) :-
      A1 == A2,
      M1 == M2,
      D1 == D2,
      H1 =< H2.
```