

5A) AGENI3.PRO

```
/*
Maquina de inferencia c/base de datos dinamica para sistema experto
Programada por C. Reynoso, febrero de 1988 - Revision dic/1990
Incluye inferencias aproximadas, seguimiento de casos sin solución,
omisión de reglas excluyentes, tratamiento de similitudes
explicación de hipótesis y consulta por menús.
*/
```

```
% Definir stack=1500
diagnostics
nowarnings
nobreak
code = 3310 % con trace 5600
```

DOMAINS

```
Historia = Rnumero*
Rnumero, Bnumero, FNO = INTEGER
Factor = REAL
Categoria = SYMBOL
archivo_datos = string
file = maniobra
slist = string*
numlista = integer*
```

```
include "tdoms.pro"
include "tpreds.pro"
include "menu.pro"
include "longmenx.pro"
```

DATABASE - principal

```
archivo_datos(archivo_datos)
asunto(string)
cons(Categoria,STRING)
cond(Bnumero,STRING)
excluyentes(integerlist)
regla(Rnumero,Categoria,Categoria,integerlist)
similares(integerlist)
```

DATABASE - maniobra

```
confianza(Bnumero,real)
e(string,integerlist)
grama(string)
interm(Bnumero,STRING)
no(Bnumero)
posible(Bnumero,Factor)
preframe(integer,string)
si(Bnumero)
reglaimbo(Rnumero,Categoria,Categoria,integerlist)
tabla(integer,Categoria)
toggle
yacaso(integer,Categoria)
yanum(integer)
yasim(Categoria)
```

database - setup

```
v(integer,integer,integer)
```

predicates

```
/* Comandos, heurísticas y procedimientos */
```

```
abducciones(Categoria)
actual(integer)
actualiza
alarma
asertsim(Bnumero)
borrar
bucle(string,string,integer,Categoria)
buclecons(string,string,Categoria,Categoria)
calculo(integerlist,Factor,Real)
carga_conoc
chance
ciclarma(integer)
compara(string,string,string,integer,Categoria)
comparacons(string,string,string,Categoria,Categoria)
comparar(integer)
componer(integerlist)
condiciones(integer)
correcto(string)
cuestion(integer,string)
derivaciones(Categoria)
desc
dev(integer,string)
elegida(integer,string)
errocons
esclista(integerlist)
evalrespu(integer,Categoria)
evaluar(real,string)
expresion(string)
frames
frase(string)
hacelis(integer,integerlist)
helper
imagen
imposible
imprime
ingresa(Categoria)
insert(integer,integerlist,integerlist)
interrogatorio
limpieza
listafinal
longlista(integerlist,integer)
loop
memoria(Categoria,integerlist)
miembro(integer,integerlist)
miembresp(integer,integerlist)
ok
parecidos(Categoria)
precomparar
preframes(integer)
preframes_2
pregunta_inicial(string)
procedimiento
proceso(integer)
proselec(integer,Categoria)
proselesp(integer,Categoria)
prosetup(integer)
purificar(Bnumero)
raya(integer)
recomparcial
recomponer
relacionadas
relaciones
reverse(integerlist,integerlist)
reverse1(integerlist,integerlist,integerlist)
rutoken(string)
rusubtoken(string)
ruconsetoken(string)
rucontoken(string)
```

```

setup
sintagma(integer,string)
sintesis(Categoria)
subraya(integer)
superconjuntos(Categoria)
tokenizar
totalizar(real)
vacio

/* Mecanismos de inferencia */

abduccion(Historia,Categoria)
dudas(Categoria)
generar_respuesta(Historia,Rnumero,STRING,Bnumero,INTEGER)
incertidumbres(Categoria)
inpq(Historia,Rnumero,Bnumero,STRING)
notest(Bnumero)
verificacion(Rnumero,Historia,integerlist)

/* Explicaciones */

mostrar_cond(Bnumero,string)
mostrar_condiciones(integerlist,string)
mostrar_regla(Rnumero,string)
pregunta(Categoria,integer,integer,Categoria)
sub_cat(Categoria,Categoria,Categoria)
trayectoria(Historia,string)

/* Actualizar el conocimiento */

edicion
guarda_basecon
leerconcl( integerlist )
obten_rnumero(Rnumero,Rnumero)
obten_bnumero(Bnumero,Bnumero)
obt_condic(Bnumero,STRING)
salva(integer)
salvar_s(integer,string,archivo_datos)

goal

trace(off),
consult("setup.dbx", setup),
v(1,A,B),
makewindow(1,A,B,
" Sistema experto con base dinámica - (c) C.Reynoso 1990",
0,0,24,80),
v(2,C,D),
makewindow(2,C,D," Explicaciones ",10,1,13,78),
v(3,E,F),
makewindow(3,E,F,
" Actualización de Conocimientos (Esc:fin - ?:help) ",
1,1,22,78),
v(5,G,H),
makewindow(5,G,H,
" Casos no resueltos - Soluciones parciales ",1,1,22,78),
v(6,I,J),
makewindow(6,I,J,
" Definiciones (PgUp, PgDn, Ctl-PgDn, Ctl-PgUp, Ctl-
F3=Search, Esc=Quit) ",
1,1,22,78),
v(7,K,L),
makewindow(7,K,L," Diagnósis y evaluación ",1,1,22,78),
v(8,M,N),
makewindow(8,M,N,
" Listados (PgUp, PgDn, Ctl-PgDn, Ctl-PgUp, Ctl-F3=Search,
Esc=Quit)",
1,1,22,78),
v(9,O,P),

```

```

makewindow(9,O,P," Mensajes ",19,8,4,65),
v(13,Q,R),
makewindow(13,Q,R," Help (PgUp, PgDn, Ctl+PgDn, Ctl+PgUp,
Ctl+F3=Search, Esc=Quit) ",1,1,22,78),
procedimiento.

```

clauses

```
/* Interface usuario */
```

```

procedimiento :-
shiftwindow(1), clearwindow,
menu(2,21,7,4, ["X. Cargar conocimiento",
"C. Consulta          *",
"S. Salvar conocimiento",
"M. Imagen interpretada de memoria",
"A. Actualizar conocimiento   *",
"B. Blanquear conocimiento en memoria ",
"E. Editar base",
"R. Metarreglas          *",
"I. Imprimir base de conocimientos",
"D. Describir y comparar entidades  **",
"T. Tokens y palabras",
"-----",
"H. Help",
"W. Switch alarma",
"U. Setup colores",
"Q. Finalizar"],
" Operaciones ",1,OPCION),
proceso(OPCION),!, procedimiento.
procedimiento :- proceso(16).

```

```

proceso(1):-!,limpieza, borrar, carga_conoc,
shiftwindow(OLD),shiftwindow(9),
storage(_H,_),
clearwindow,
write(" >> Base de conocimientos activa en memoria. "),nl,
write(" >> Quedan ",H," caracteres libres para datos."),
readchar(_),
shiftwindow(OLD),!.
proceso(2):-!, ok, recomponer, interrogatorio.
proceso(3):-!, ok, limpieza, recomponer, guarda_basecon.
proceso(4):-!, ok, limpieza, recomponer, imagen.
proceso(5):-!,limpieza, recomponer, actualiza.
proceso(6):-!, ok, limpieza, borrar,
shiftwindow(OLD),shiftwindow(9),
storage(_H,_),
clearwindow,
write(" >> La memoria del sistema ha sido blanqueada "),nl,
write(" >> Hay ",H," caracteres libres para datos."),
readchar(_),
shiftwindow(OLD),!.
proceso(7):-!,limpieza, edicion.
proceso(8):-!,ok, limpieza, relaciones.
proceso(9):-!,ok, limpieza, recomponer, imagen, imprime.
proceso(10):-!,ok, limpieza,recomponer,
openwrite(maniobra,"compara.dba"),
closefile(maniobra),precomparar.
proceso(11) :-!,ok, limpieza,recomponer, tokenizar.
proceso(12):-!.
proceso(13) :-!,helper.
proceso(14) :-!,desc.
proceso(15) :-!,setup.
proceso(16):-!,
removewindow(1,1),
removewindow(2,1),
removewindow(3,1),
removewindow(5,1),
removewindow(6,1),

```

```

removewindow(7,1),
removewindow(8,1),
removewindow(9,1),
removewindow(13,1),
exit.

/* Mecanismo de inferencia */

interrogatorio:-
    !,abducciones(Hipotesis),      % corte nuevo
    nl,nl,abduccion([],Hipotesis),!

vacio :-
    !,shiftwindow(OLD),
    shiftwindow(9),
    clearwindow,
    write(" >> No hay información en memoria."),nl,
    write(" >> Cargar o actualizar base de conocimientos. "),
    alarma,
    readchar(_),
    shiftwindow(OLD).

abducciones(Hipotesis):-
    !,limpieza,
    clearwindow,
    repeat,nl,nl,nl,nl,
    assert(tabla(1,"fin de consulta"),maniobra),
    proselec(1,Hipotesis),!

proselec(N,Hipotesis) :-
    regla(_,Categoria,_,_),
    not(tabla(_,Categoria)),
    NN=N+1,
    assert(tabla(NN,Categoria), maniobra),
    proselec(NN,Hipotesis).

proselec(_,Hipotesis) :-
    findall(X, tabla(_,X), ESQ),
    longmenu(1,1,20,7,7,ESQ," Objetos ",1,OP),
    tabla(OP,Hipotesis).

proselesp(N,Hipotesis) :-
    findall(Cate, regla(_,_,Cate,_), ListaCat),
    longmenu(1,1,20,7,15,
        ListaCat,
        " Elegir entidad ",0,Opcion),
    regla(Opcion,_,Hipotesis,_).

evalrespu(1,Hipotesis):-
    write(" si."),
    nl,nl,
    retractall(yanum(_),maniobra),
    shiftwindow(7),
    openwrite(maniobra,"eval.txt"),
    writedevicemaniobra,
    sintesis(Hipotesis),nl,
    dudas(Hipotesis),
    retractall(yanum(_),maniobra),
    incertidumbres(Hipotesis),nl,
    recomponer,
    parecidos(Hipotesis),nl,
    derivaciones(Hipotesis),nl,
    subraya(78),nl,
    write("\t\t\t*** Proceso terminado ***"),nl,
    subraya(78),
    closefile(maniobra),
    file_str("eval.txt",ST),
    writedevicemscreen,
    display(ST),
    shiftwindow(1),clearwindow,

superconjuntos(Hipotesis),nl,
limpieza,
clearwindow.

evalrespu(2,Hipotesis):-
    write(" no."),
    imposible,
    !,limpieza,
    procedimiento.
evalrespu(2,Hipotesis):- !,proceso(2).

/* Eleccion más probable */

abduccion(_, "fin de consulta") :-
    limpieza,
    !, procedimiento.
abduccion(_, Hipotesis):-
    not(regla(_,Hipotesis,_,_)),!nl,
    write("Es posible que sea ",Hipotesis),nl,nl,
    alarma,
    write("Es la respuesta correcta?"),
    menu(19,60,7,7,["Si (o posible) ", "No (o dudoso)"],
        " Diagnóstico ",1,OP),
    evalrespu(OP,Hipotesis).
abduccion(Historia, Hipotesis) :-
    regla(Rnumero,Hipotesis,NY,COND),
    verificacion(Rnumero,Historia, COND),
    abduccion([Rnumero|Historia],NY).

/* Casos con respuestas positivas incompletas */

abduccion(,_):-
    imposible.

imposible :-
    openwrite(maniobra,"manio.dba"),
    writedevicemaniobra,
    write("El sistema no puede resolver el caso."),nl,
    write("Actualice o modifique la base de conocimientos."),
    nl,nl,
    si(Asertado),
    regla(_,Hipotesis,_,Condiciones),
    miembro(Asertado,Condiciones),
    cond(Asertado,Texto),
    not(yacaso(Asertado,Hipotesis)),
    assert(yacaso(Asertado,Hipotesis),maniobra),
    write("El caso ",Hipotesis),nl,
    write(" satisface la condicion ",Asertado,":"),nl,
    write(" -> ",Texto),nl,nl,
    fail.

imposible :-
    si(Asertado),
    regla(,_,_),Hipotesis,Condiciones),
    miembro(Asertado,Condiciones),
    cond(Asertado,Texto),
    not(yacaso(Asertado,Hipotesis)),
    assert(yacaso(Asertado,Hipotesis),maniobra),
    write("El caso ",Hipotesis),nl,
    write(" satisface la condicion ",Asertado,":"),nl,
    write(" -> ",Texto),nl,nl,
    fail.

imposible :- nl,
closefile(maniobra),
file_str("manio.dba",ST),
shiftwindow(5),
display(ST),
shiftwindow(1),

```

```

!,procedimiento.

verificacion(Rnumero, Historia, [Bnumero|RESTO]):-
si(Bnumero), !,
verificacion(Rnumero, Historia, RESTO).
verificacion( _, _, [Bnumero|_]):-
no(Bnumero), !,fail.
verificacion(Rnumero, Historia, [Bnumero|RESTO]):-
cond(Bnumero,NCOND),
fronttoken(NCOND,"no",_COND),
frontchar(_COND,_COND),
cond(Bnumero1,COND),
notest(Bnumero1), !,
verificacion(Rnumero, Historia, RESTO).
verificacion( _,_, [Bnumero|_]):-
cond(Bnumero,NCOND),
fronttoken(NCOND,"no",_COND),
frontchar(_COND,_COND),
cond(Bnumero1,COND),
si(Bnumero1), !,fail.
verificacion(Rnumero, Historia, [Bnumero|RESTO]):-
cond(Bnumero,TEXTO),
inpq(Historia,Rnumero,Bnumero,TEXTO),
verificacion(Rnumero, Historia, RESTO).
verificacion( _, _, [] ).

notest(Bnumero):-no(Bnumero),!.
notest(Bnumero):-not(si(Bnumero)),!.

inpq(Historia,Rnumero,Bnumero,TEXTO):-
!,pregunta_inicial(Frase),
write(Frase," ",TEXTO,"? : "),
menu(1,65,7,7,["si","es posible ","no","por qué?"],
" Resp ",1,OPCION),
generar_respuesta(Historia,Rnumero,TEXTO,Bnumero,OPCION).

pregunta_inicial(Frase) :-
!,random(5,Azar), cuestion(Azar,Frase).

cuestion(0,"¿Es verdad que"). % preguntas variables
cuestion(1,"¿Puede asegurar que").
cuestion(2,"¿Es cierto que").
cuestion(3,"¿Afirmaría que").
cuestion(4,"¿Es correcto que").

generar_respuesta( _,_,_,0):-
limpieza,!,procedimiento.
generar_respuesta( _,_,_,Bnumero,1):-
assert(si(Bnumero),maniobra),
asertsim(Bnumero),
shiftwindow(1),write(si),nl,
purificar(Bnumero).
generar_respuesta( _,_,_,Bnumero,2):-
assert(si(Bnumero),maniobra),
asertsim(Bnumero),
shiftwindow(1),nl,write(" ". posible),
menu(1,24,7,7,
["1. Altamente improbable",
"2. Improbable",
"3. Dudoso",
"4. Algo dudoso",
"5. Certidumbre intermedia",
"6. Alguna certidumbre ",
"7. Cierta seguridad",
"8. Muy seguro",
"9. Casi totalmente seguro",
"10. Probabilidad casi absoluta "],
" Magnitud de Posibilidad ",1,Factint),
Factor = Factint/10,write(" ",Factor),

assert(posible(Bnumero,Factor)),nl,
purificar(Bnumero).
generar_respuesta( _,_,_,Bnumero,3):-
assert(no(Bnumero),maniobra),
shiftwindow(1),write(no),nl,fail.
generar_respuesta(Historia,Rnumero,TEXTO,Bnumero,4):- !,
shiftwindow(2),
regla( Rnumero, Hipotesis1, Hipotesis2, _ ),
sub_cat(Hipotesis1,Hipotesis2,Lstr),
concat("Se intenta demostrar que ",Lstr,Lstr1),
concat(Lstr1,"\nUsando la regla número ",Ls1),
str_int(Str_num,Rnumero),
concat(Ls1,Str_num,Ans),
mostrar_regla(Rnumero,Ls1),
concat(Ans,Ls1,Ans1),
trayectoria(Historia,Sng),
concat(Ans1,Sng,Answ),
display(Answ),
shiftwindow(1),
menu(1,65,7,7,["si","es posible ","no","por qué?"],
" Resp ",1,OPCION),
generar_respuesta(Historia,Rnumero,TEXTO,Bnumero,OPCION).

asertsim(Bnumero) :-
similares(Conjunto),
miembro(Bnumero,Conjunto),
miembro(Otro,Conjunto),
Bnumero<>Otro,
assert(si(Otro),maniobra),
fail.
asertsim(_).

purificar(Bnumero) :-
excluyentes(Lista),
miembro(Bnumero,Lista),
miembro(X,Lista), X <> Bnumero,
cond(X,Datos),
assert(interm(X,Datos),maniobra),
retract(cond(X,_),principal),
fail.
purificar(_).

/* Mecanismo de explicacion */

trayectoria([], "").
trayectoria([Rnumero|RESTO],Strg) :-
regla( Rnumero, Hipotesis1, Hipotesis2, _ ),
sub_cat(Hipotesis1,Hipotesis2,Lstr),
concat("\nYa se ha demostrado que ",Lstr,L1),
concat(L1,"\nUsando regla número ",L2),
str_int(Str_Rnumero,Rnumero),
concat(L2,Str_Rnumero,L3),
concat(L3,"\n ",L4),
mostrar_regla(Rnumero,Str),
concat(L4,Str,L5),
trayectoria(RESTO,Sigte_strg),
concat(L5,Sigte_strg,Strg).

sub_cat(Hipotesis1,Hipotesis2,Lstr):-
concat(Hipotesis1," es ",Str),
concat(Str,Hipotesis2,Lstr).

mostrar_regla(Rnumero,Strg):-
regla(Rnumero, Hipotesis1, Hipotesis2, CONDINGELSER),
str_int(Rnumero_str,Rnumero),
concat("\n Regla ",Rnumero_str,Ans),
concat(Ans,": ",Ans1),
sub_cat(Hipotesis1,Hipotesis2,Lstr),
concat(Ans1,Lstr,Ans2),

```

```

concat(Ans2,"n si ",Ans3),
reverse(CONDINGELSER,CONILS),
mostrar_condiciones(CONILS,Con),
concat(Ans3,Con,Strg).

mostrar_condiciones([], "").
mostrar_condiciones([COND],Ans):-
  mostrar_cond(COND,Ans),!.
mostrar_condiciones([COND|RESTO],Ans):-
  mostrar_cond(COND,Text),
  concat("\n y ",Text,Nstr),
  mostrar_condiciones(RESTO,Next_ans),
  concat(Next_ans,Nstr,Ans).

mostrar_cond(COND,TEXTO):-cond(COND,TEXTO).

/* Sintesis del diagnostico */

sintesis(Hipotesis) :-
  write("La entidad es ",Hipotesis),nl,
  regla(_,_,Hipotesis,Condiciones),
  memoria(Hipotesis,Condiciones).

memoria(Hipotesis,Condiciones) :-
  miembro(X,Condiciones),
  cond(X,TEXTO),
  write(X," -> ",TEXTO),nl,
  fail.
memoria(_,_).

/* Conocimiento imperfecto y respuestas variadas */

dudas(Hipotesis) :-
  si(Asertado),
  regla(_,Clase,Hipotesis,Condiciones),
  regla(_,_,Clase,Cond_clase),
  not(miembro(Asertado,Condiciones)),
  not(miembro(Asertado,Cond_clase)),
  cond(Asertado,STRING),
  not(yanum(Asertado)),
  assert(yanum(Asertado)),
  expresion(Frase),
  write(Frase),nl,
  write(" * ",STRING, " *"),nl,
  write("es inespecífica, redundante o no aparece en
definiciones"),nl,
  write("alternativas del mismo caso."),
  nl,nl,fail.
dudas(Hipotesis):-
  si(Asertado),
  regla(_,Clase,Hipotesis,Condiciones),
  regla(_,_,Clase,Cond_clase),
  not(miembro(Asertado,Condiciones)),
  miembro(Asertado,Cond_clase),
  cond(Asertado,STRING),
  not(yanum(Asertado)),
  assert(yanum(Asertado)),
  expresion(Frase),
  write(Frase),nl,
  write(" * ",STRING, " *"),nl,
  write("se deriva por herencia de la clase"),
  write(" a la que pertenece el ejemplar."),
  nl,nl,fail.
dudas(_).

expresion(Frase) :- random(4,S),
  sintagma(S,Frase).

```

```

sintagma(0,"Se señala que la regla asertada siguiente :").
sintagma(1,"Se advierte que la regla asertada que se indica :").
sintagma(2,"Tener en cuenta que esta regla :").
sintagma(3,"Obsérvese que la regla :").

```

```

incertidumbres(Hipotesis) :-
  posible(Asertado,Factor),
  cond(Asertado,Texto),
  not(yanum(Asertado)),
  assert(yanum(Asertado),maniobra),
  frase(Frase),
  write(Frase," ",Factor," posible:"),nl,
  write(" -> ",Texto,";"),nl,
  regla(_,_,Hipotesis,Condiciones),
  calculo(Condiciones,Factor,RES),
  write("no es segura y otorga "),
  writef("%-4.2",RES),
  write(" de certidumbre al diagnóstico."),
  retract(posible(Asertado,_),maniobra),
  assert(confianza(Asertado,RES),maniobra),
  nl,nl,
  !,incertidumbres(Hipotesis).
incertidumbres(_) :- totalizar(1).

```

```

parecidos(Hipotesis) :-
  similares(Lista),
  miembro(A,Lista),
  miembro(B,Lista),
  A<>B,
  regla(_,_,Hipotesis,Condiciones),
  regla(_,_,Otrahip,OtraCond),
  Hipotesis<>Otrahip,
  miembro(A,Condiciones),
  miembro(B,OtraCond),
  not(yasim(Otrahip)),
  assert(yasim(Otrahip),maniobra),
  write("El caso ",Otrahip),
  write(" satisface una condición similar:"),nl,
  cond(A,T1),
  write("Orig.: ",A," -> ",T1),nl,
  cond(B,T2),
  write("Caso : ",B," -> ",T2),nl,nl,
  fail.
parecidos(_).

```

```

frase(Frase) :-
  !,random(5,Azar),elegida(Azar,Frase).

```

```

elegida(0,"La siguiente regla, definida como").
elegida(1,"Se advierte que la condición siguiente, especificada").
elegida(2,"La regla que se indica, a la cual se definió como").
elegida(3,"La expresión condicional que se especifica, definida").
elegida(4,"La siguiente condición incierta, especificada").

```

```

totalizar(TOT) :-
  confianza(AS,CONF),
  TOTAL=TOT*CONF,
  retract(confianza(AS,_),maniobra),
  !,totalizar(TOTAL).
totalizar(TOTAL) :-
  evaluar(TOTAL,TEXTO),
  write("Confianza total del diagnóstico : "),
  writef("%-4.2",TOTAL),
  write(" - ",TEXTO),nl.

```

```

evaluar(TOTAL,"altísima.") :- TOTAL > 0.95.
evaluar(TOTAL,"muy alta.") :- TOTAL > 0.9.
evaluar(TOTAL,"alta.") :- TOTAL > 0.75.
evaluar(TOTAL,"intermedia.") :- TOTAL > 0.5.

```

```

evaluar(TOTAL,"baja.") :- TOTAL > 0.25.
evaluar(TOTAL,"muy baja.") :- TOTAL > 0.10.
evaluar(TOTAL,"bajisima.").

calculo(Condiciones,Factor,RES) :-
!,longista(Condiciones,Longitud),
RES = ((Longitud-1) + Factor)/Longitud.

/* Actualizar la base de conocimientos */

actualiza:-
menu(7,25,7,7,["A. Definiciones Condicionales ",
"B. Consecuentes de clases",
"C. Consecuentes especificos",
"Q. Fin"],
" Actualizar ",1,OP),
limpieza,
actual(OP).
actualiza.

actual(1) :-
shiftwindow(3), clearwindow,
write("\n Actualizar datos\n "),
subraya(16),
cursor(1,20),
write("Categoría : "),
raya(41),
cursor(3,20),
write("Subcategoría: "),
raya(41),
cursor(1,34),
readln(KAT1),KAT1><"",
pregunta(KAT1,1,34,KAT),
cursor(3,34),
readln(SUB1),SUB1><"",
pregunta(SUB1,3,34,SUB),
leercondl(CONDL),
obten_rnumero(1,Rnumero),
assert(regla(Rnumero,KAT,SUB,CONDL)),
actual(1).

actual(1) :-
menu(7,27,7,7,["Si ", "No"],
" Guarda los datos? ",1,OP),
salva(OP).

actual(2) :-
!,assert(tabla(1,"fin de proceso"),maniobra),
proselec(1,Hipotesis),
ingresa(Hipotesis).

actual(3) :-
!,assert(tabla(1,"fin de proceso"),maniobra),
proselesp(1,Hipotesis),
ingresa(Hipotesis).

actual(4).

ingresa("fin de proceso").
ingresa(Hipotesis) :-
write("Escribir texto y <enter>, o <enter> para fin : "),nl,
readln(T),
T<>"",
assert(cons(Hipotesis,T),principal),
!,ingresa(Hipotesis).
ingresa(_) :- clearwindow.

salva(1) :- chance.
salva(2).

```

```

salva(3).

pregunta(Q,X,Y,Q2):- Q = "?",
shiftwindow(2),clearwindow,
write("Las categorías y subcategorías son objetos. P.ej.: \n"),nl,
write("categoría-----subcategoría---[condición 1 -----
condición 2]\n"),
write("-----Á-----Á-----Á---Á-----Á-----Á-----Á-----"),nl,
l,
write("animal ³es un³ mamífero ³si ³tiene pelo ³y si ³ da leche\n"),
write("animal ³es un³ pájaro ³si ³tiene plumas ³y si ³ pone
huevos"),nl,
write("mamífero ³es un³ perro ³si ³ladra ³ ³"),
readchar(_),clearwindow,
shiftwindow(3),
cursor(X,Y),
readln(Q2).
pregunta(Q,_,_,Q).

obten_rnumero(N,N):-not(regla(N,_,_,_)),!.
obten_rnumero(N,N1):-H=N+1,obten_rnumero(H,N1).

obten_bnumero(N,N):-not(cond(N,_)),!.
obten_bnumero(N,N1):-H=N+1,obten_bnumero(H,N1).

leercondl([BnumeroR]):-
nl,write(" Condición: "),
cursor(A,B),
raya(63),
cursor(A,B),
readln(COND),
COND >< "", !,
obt_condic(Bnumero,COND),
leercondl( R ).
leercondl([]).

obt_condic(Bnumero,COND):-cond(Bnumero,COND),!.
obt_condic(Bnumero,COND):-obten_bnumero(1,Bnumero),
assert(cond(Bnumero,COND)).

/* Edicion de la base de conocimientos */

edicion :-
makewindow(11,7,7, "",1,1,22,78),
dir("", "*.dba",Base,1,1,1),
file_str(Base,Datos),
edit(Datos,Nuevosdat,"Editor interno", "",
"",1, "",1,
0,1,1,_,_),
menu(10,20,7,7,["Si ", "No"],
"¿Resguarda Base de Conocimientos?",1,Respu),
salvar_s(Respu,Nuevosdat,Base),
removewindow.

edicion:-
existwindow(11),removewindow(11,1).
edicion.

salvar_s(1,D,Base):-
openwrite(maniobra,Base),
writedevic(maniobra),
write(D),
closefile(maniobra).
salvar_s(.,.,.).

/* Comandos usuario */

recomparcial :-
interm(A,B),

```

```

assert(cond(A,B), principal),
retract(interm(A,B),maniobra),
fail.
recomparcial.

recomponer :-
interm(A,B),
assert(cond(A,B), principal),
retract(interm(A,B),maniobra),
fail.
recomponer :-
reglaimbo(A,B,C,D),
assert(regla(A,B,C,D)),
retract(reglaimbo(A,B,C,D)),
fail.
recomponer.

carga_conoc :-
makewindow(10,7,7,"",10,8,10,64),
retractall(regla(_,_,_),principal),
retractall(cond(_,_),principal),
dir("","*.dba",Base,1,1,1),
trap(consult(Base,principal),_,errocons),
removewindow.
carga_conoc :-
removewindow(10,1),!,procedimiento.

errocons :- !,nl,
attribute(OLD), attribute(15),      % antes sin corte
clearwindow,
nl,nl,
write("    *** Error de consulta en base de datos *** "),nl,
write("    ***** Verificar estructura del archivo ***** "),nl,
alarma, readchar(_),
attribute(OLD), clearwindow,
!,procedimiento.

guarda_basecon :-
archivo_datos(Datos), bound(Datos),!,
save(Datos,principal),clearwindow,
shiftwindow(OLD),
shiftwindow(9),
writef(" La base % ha sido resguardada ",Datos),
readchar(_),clearwindow,
shiftwindow(OLD).
guarda_basecon :- chance.

chance :-
makewindow(4,7,7," Nombre del archivo ",10,20,4,45),
correcto(Datos),
assert(archivo_datos(Datos)),
save(Datos,principal),clearwindow,nl,
writef("La base % ha sido resguardada",Datos),
readchar(_),
removewindow,
clearwindow.

correcto(Datos) :-
write("Ingrese Nombre de Base de Conocimientos : "),nl,
readln(Datos),
str_len(Datos,L), L<13.
correcto(Datos) :- !,correcto(Datos).

borrar:-
!,retractall(_,principal),
retractall(_,maniobra).

limpieza:-
!,retractall(grama(_), maniobra),

```

```

retractall(preframe(_,_), maniobra),
retractall(yacaso(_,_),maniobra),
retractall(yasim(_),maniobra),
retractall(tabla(_,_),maniobra),
retractall(si(_),maniobra),
retractall(no(_),maniobra),
retractall(confianza(_,_),maniobra),
retractall(possible(_,_),maniobra),
retractall(yanum(_),maniobra),
retractall(e(_,_),maniobra).

```

/* Imagen interpretada de las reglas en Memoria */

```

imagen :-
!,
attribute(OLD),Blink=OLD+128,
attribute(Blink),
nl,write(" ** Se están interpretando los datos ** "),
attribute(OLD),
openwrite(maniobra,"manio.dba"),
writedevicemaniobra,
loop,nl,
derivaciones(Categoria),
storage(S,H,T),nl,
write("Stack : ",S," caracteres"),nl,
write("Heap : ",H," caracteres"),nl,
write("Trail : ",T," caracteres"),nl,
closefile(maniobra),
file_str("manio.dba",ST),
writedevicemaniobra,
shiftwindow(VIE),shiftwindow(8),
display(ST), shiftwindow(VIE), !.

```

```

loop :-
regla(Rnumero,Cate1,Cate2,Condiciones),nl,
write("Regla N° ",Rnumero," -> "),
write(Cate1," es ",Cate2," si :"),nl,
miembro(X,Condiciones),
cond(X,STRING),
write(" ",X," <- ",STRING),nl,
fail.
loop :-
excluyentes(Lista),nl,
write("Condiciones excluyentes :"),nl,
miembro(Cond,Lista),
cond(Cond,Texto),
write(" ",Cond," -> ",Texto,"."),nl,fail.

```

```

loop :-
similares(L),nl,
write("Condiciones similares : "),nl,
eslista(L),
fail.
loop.

```

```

derivaciones(Cat) :-
cons(Cat,Texto),
write(Cat," >> ",Texto),nl,nl,
fail.
derivaciones(_).

```

/* Relaciones entre reglas */

```

relaciones :-
recomponer,
clearwindow,
menu(4,19,7,7,
["1. Lista de relaciones entre reglas",
"2. Definición de condiciones excluyentes ",
"3. Definición de condiciones similares",

```

```

"4. Eliminar relaciones de exclusión",
"5. Eliminar relaciones de similitud",
"6. Frames",
"Q. Menú principal"],
" Reglas y condiciones ",1,OPCION),
condiciones(OPCION),!,relaciones.

condiciones(1) :-
!,openwrite(maniobra,"manio.dba"),
writedevic(maniobra),
relacionadas.

condiciones(2) :-
!,findall(REG, cond(_,REG), LISTAREG),
longmenu_mult(1,1,20,7,7,LISTAREG,
" EXCLUYENTES -- Sel/Dsel <Enter> -- OK <F10> -- Quit
<Esc> ",
[],Talis),
longlista(Talis,Long),
Long>0,
reverse(Talis,Lista),
memoria(_,Lista),
menu(1,61,7,7,["Si","No"],
" ¿Confirma? ",1,SN),
SN=1,
assert(excluyentes(Lista),principal),
clearwindow.

condiciones(3) :-
!,findall(REG, cond(_,REG), LISTAREG),
longmenu_mult(1,1,20,7,7,LISTAREG,
" SIMILARES -- Sel/Dsel <Enter> -- OK <F10> -- Quit <Esc>
",
[],Talis),
longlista(Talis,Long),
Long>0,
reverse(Talis,Lista),
memoria(_,Lista),
menu(1,61,7,7,["Si","No"],
" ¿Confirma? ",1,SN),
SN=1,
assert(similares(Lista),principal),
clearwindow.

condiciones(4) :- !,retractall(excluyentes(_),principal),
shiftwindow(OLD),shiftwindow(9),
clearwindow,
write(" >> Se ha desactivado la exclusión de reglas."),nl,
write(" >> La consulta considerará todas las opciones. "),
readchar(_),clearwindow,
shiftwindow(OLD),clearwindow.
condiciones(5) :- !,retractall(similares(_),principal),
shiftwindow(OLD), shiftwindow(9),
clearwindow,
write(" >> Se ha desactivado la asimilación de reglas."),nl,
write(" >> La consulta no producirá indicaciones de similitud. "),
readchar(_),clearwindow,
shiftwindow(OLD),
clearwindow.

condiciones(6) :-
!,
menu(1,61,7,7,["Si","No"],
" ¿Confirma? ",1,SN),
SN=1,
retractall(excluyentes(_)),
write(" >>> Seleccionar exclusiones con <Enter> y fin con
<F10>"),
preframes(0).

```

```
condiciones(7) :- !,procedimiento.
```

```

relacionadas :-
excluyentes(L),
write("Reglas excluyentes : "),nl,
eslista(L),nl,
fail.
relacionadas :-
similares(L),
write("Reglas similares : "),nl,
eslista(L),nl,
fail.
relacionadas :-
closefile(maniobra),
writedevic(screen),
file_str("manio.dba",ST),
shiftwindow(OLD),shiftwindow(8),
display(ST),
shiftwindow(OLD),
clearwindow.

```

```
/* Frames */
```

```

preframes(N) :-
cond(_,Texto1),
fronttoken(Texto1, Token1, _),
not(preframe(_,Token1)),
Num=N+1,
assert(preframe(Num,Token1)),
!,preframes(Num).
preframes(N) :- preframes_2.

```

```

preframes_2 :-
findall(Token, preframe(_,Token), Listoken),
longmenu_mult(2,2,18,7,7,Listoken," Slots ",[],Listagrama),
componer(Listagrama),
frames.

```

```

componer(Listagrama) :-
miembresp(X,Listagrama),
preframe(X,Token),
assert(grama(Token)),
fail.
componer(_).

```

```

frames :-
cond(N1,Texto1),
fronttoken(Texto1, Token1, _),
not(e(Token1,_)),
not(grama(Token1)),
assert(yanum(N1)),
hacelis(N1,L1),
assert(e(Token1,L1),maniobra),
fail.

```

```

frames :-
cond(N2,Texto2),
fronttoken(Texto2, Token, _),
e(Token,N),
not(yanum(N2)),
assert(yanum(N2)),
insert(N2,N,L2),
retract(e(Token,N)),
assert(e(Token,L2)),
fail.

```

```

frames :-
openwrite(maniobra,"manio.dba"),
writedevic(maniobra),
listafinal.

```

```

listafinal :-
    e(Token,N),nl,
    write("Frame : ",Token),nl,
    esclista(N),
    assert(excluyentes(N)),
    fail.
listafinal :-
    closefile(maniobra),
    writedevic(screen),
    file_str("manio.dba",ST),
    shiftwindow(OLD),shiftwindow(8),
    display(ST),shiftwindow(OLD),
    clearwindow.

hacelis(Token1,[Cabeza]) :-
    Cabeza=Token1,!.

/* Descripcion y comparacion de entidades */

precomparar :-
    menu(8,25,7,7,["A. Categorías generales",
        "B. Categorías específicas ",
        "Q. Menú principal"],
        " Nivel ",1,NIV),
    comparar(NIV).
precomparar.

comparar(1) :-
    shiftwindow(6),
    findall(Cate, regla(_,_Cate,_,_), ListaCat),
    longmenu(1,1,20,7,15,
        ListaCat,
        " Elegir entidad ",0,Opcion),
    regla(Opcion,Catdef,Subcate,Listacond),
    openappend(maniobra,"compara.dba"),
    writedevic(maniobra),
    write("La entidad ",Catdef," es ",
        Subcate," <",Opcion,"> si :"),nl,
    esclista(Listacond),nl,
    derivaciones(Catdef),
    closefile(maniobra),
    file_str("compara.dba",ST),
    writedevic(screen),
    display(ST),
    shiftwindow(1),
    !,comparar(1).
comparar(1) :-
    shiftwindow(1).

comparar(2) :-
    shiftwindow(6),
    findall(Cate, regla(_,_Cate,_), ListaCat),
    longmenu(1,1,20,7,15,
        ListaCat,
        " Elegir entidad ",0,Opcion),
    regla(Opcion,_Catdef,Listacond),
    openappend(maniobra,"compara.dba"),
    writedevic(maniobra),
    write("La entidad es ",Catdef," <",Opcion,"> si :"),nl,
    esclista(Listacond),nl,
    derivaciones(Catdef),
    closefile(maniobra),
    file_str("compara.dba",ST),
    writedevic(screen),
    display(ST),
    shiftwindow(1),
    !,comparar(2).
comparar(2) :-

```

```

    shiftwindow(1).
    comparar(3).

/* Tokenizar */

tokenizar :-
    !,write("Indicar slot o palabra : "),
    cursor(RR,CC),
    raya(30),
    cursor(RR,CC),
    readln(Tokin),nl,
    attribute(OLD),Blink=OLD+128,
    attribute(Blink),
    write(" *** Se están procesando los datos *** "),
    attribute(OLD),
    openwrite(maniobra,"manio.dba"),
    writedevic(maniobra),
    write("Reglas categoriales :"),nl,
    subraya(78),nl,
    rutoken(Tokin),nl,
    write("Reglas subcategoriales :"),nl,
    subraya(78),nl,
    rusubtoken(Tokin),nl,
    write("Condiciones :"),nl,
    subraya(78),nl,
    rucontoken(Tokin),nl,
    write("Consecuencias :"),nl,
    subraya(78),nl,
    ruconsetoken(Tokin),
    closefile(maniobra),
    file_str("manio.dba",ST),
    writedevic(screen),
    shiftwindow(VIE),shiftwindow(8),
    display(ST),shiftwindow(VIE),
    clearwindow.

rutoken(Tokin) :-
    regla(Numero,Categoria,_,_),
    upper_lower(Mayutok,Tokin),
    upper_lower(Mayucat,Categoria),
    bucle(Mayutok,Mayucat,Numero,Categoria),
    fail.
rutoken(_).

rusubtoken(Tokin) :-
    regla(Numero,_Categoria,_),
    upper_lower(Mayutok,Tokin),
    upper_lower(Mayucat,Categoria),
    bucle(Mayutok,Mayucat,Numero,Categoria),
    fail.
rusubtoken(_).

rucontoken(Tokin) :-
    cond(Numero,Categoria),
    upper_lower(Mayutok,Tokin),
    upper_lower(Mayucat,Categoria),
    bucle(Mayutok,Mayucat,Numero,Categoria),
    fail.
rucontoken(_).

ruconsetoken(Tokin) :-
    cons(Madre,Categoria),
    upper_lower(Mayutok,Tokin),
    upper_lower(Mayucat,Categoria),
    buclecons(Mayutok,Mayucat,Madre,Categoria),
    fail.
ruconsetoken(_).

bucle(Mayutok,Mayucat,Numero,Categoria) :-

```

```

fronttoken(Mayucat, Token, Resto),
compara(Token,Mayucat,Mayutok,Numero,Categoria),
bucle(Mayutok,Resto,Numero,Categoria).
bucle(_,_,_).

buclecons(Mayutok,Mayucat,Madre,Categoria) :-
fronttoken(Mayucat, Token, Resto),
comparacons(Token,Mayucat,Mayutok,Madre,Categoria),
buclecons(Mayutok,Resto,Madre,Categoria).
buclecons(_,_,_).

compara(Token,Mayucat,Mayutok,Numero,Categoria) :-
Token=Mayutok,
write(" N° ",Numero," -> ",Categoria),nl.
compara(_,_,_).

comparacons(Token,Mayucat,Mayutok,Madre,Categoria) :-
Token=Mayutok,
write(" Madre : ",Madre),nl,
write(" -> ",Categoria),nl,nl.
comparacons(_,_,_).

/* Superconjuntos */

superconjuntos(Hipotesis) :-
menu(18,48,71,14,["Si (otras soluciones) ",
" No (menú principal)",
" ¿Prosigue la búsqueda? ",1,SN),
SN=2.
superconjuntos(Hipotesis) :-
regla(Num,Madre,Hipotesis,Condiciones),
assert(reglalimbo(Num,Madre,Hipotesis,Condiciones)),
retract(regla(Num,Madre,Hipotesis,Condiciones)),
recomparcial,
!,interrogatorio.

/* Ok */

ok :- regla(_,_,_).
ok :- vacio, !, procedimiento.

/* Help */

helper :-
file_str("ageni.hlp",ST),
shiftwindow(13),
display(ST),
shiftwindow(1).
helper.

/* Impresión */

imprime :-
!,clearwindow,
menu(7,30,7,7,["LPT1", "LPT2", "COM1", "PRN", "KBPRINT.PRN
"],
" Dispositivo ",5,DEV),
dev(DEV,DISPO),
concat("COPY MANIO.DBA ",DISPO,UNO),
concat(UNO," > NUL",COMA),
system(COMA,0,_).

dev(1,"LPT1").
dev(2,"LPT2").
dev(3,"COM1").
dev(4,"PRN").
dev(5,"KBPRINT.PRN").

% Setup

setup :-
miembro(X, [1,2,3,5,6,7,8,9,13]),
prosetup(X),
fail.
setup :-
save("setup.dbx", setup),
shiftwindow(1).

prosetup(X) :-
shiftwindow(X),
clearwindow,
write("\t\t\t*** Set up window ",X," ***"),nl,
write("\t\t\t*** Seleccione y <Enter> ***"),
colorsetup(0),
clearwindow,
write("\t\t\t*** Set up frame ",X," ***"),nl,
write("\t\t\t*** Seleccione y <Enter> ***"),
colorsetup(1),
makewindow(Z,Y,W,_,_,_),
retractall(v(Z,_)),
assert(v(X,Y,W)),
clearwindow.
prosetup(X).

/* Sonidos */

alarma :- toggle.
alarma :-
random(6,P), PA=P+1,
ciclalarma(PA).

desc :- toggle, retract(toggle).
desc :- assert(toggle).

ciclalarma(PA) :-
random(400,S),SO=S+40,
sound(15,SO),
PAA=PA-1,PAA>0,
!,ciclalarma(PAA).
ciclalarma(_).

raya(A) :-
A>0, write(""),
AA=A-1,!,raya(AA).
raya(_).

subraya(A) :-
A>0, write("-"),
AA=A-1,!,subraya(AA).
subraya(_).

/* Procesamiento de listas */

longlista([],0).
longlista(_[_]Cola,K):-
longlista(Cola,J,K=J+1).

miembro(X,[Cabeza_]):- Cabeza=X.
miembro(X,_[_]Cola):- miembro(X,Cola).

miembresp(X,[Cabeza_]):- Cabeza=X.
miembresp(X,_[_]Cola):- miembresp(X,Cola).

reverse(X,Y):-
reverse1([],X,Y).
reverse1(Y,[],Y).
reverse1(X1,[U|X2],Y):-reverse1([U|X1],X2,Y).

```

esclista([]):-!.
 esclista([H/T]) :-
 cond(H,Texto),
 write(" ",H," -> ",Texto),nl,
 !,esclista(T).

insert(X,[],[X]).
 insert(X,[YIYs],[YIZs]) :- X>Y, insert(X,Ys,Zs).
 insert(X,[YIYs],[X,YIYs]) :- X<=Y.

5B) BASE DE CONOCIMIENTOS DE PRUEBA

archivo_datos("ageni.db")
 asunto("ceramica")
 regla(1,"diagnosis","ceramica peruana",[1])
 regla(2,"diagnosis","ceramica argentina",[2])
 regla(3,"diagnosis","ceramica boliviana",[12])
 regla(4,"ceramica peruana","Inca",[5])
 regla(5,"ceramica peruana","Chimu",[6,10])
 regla(6,"ceramica peruana","Chimu",[6,11])
 regla(7,"ceramica peruana","Mochica",[7,10])
 regla(8,"ceramica peruana","Mochica",[7,11])
 regla(9,"ceramica boliviana","Tiawanaco",[13,14])
 regla(10,"ceramica boliviana","Tiawanaco",[13,15])
 regla(11,"ceramica argentina","Cienaga",[46,8,3,4])
 regla(12,"ceramica argentina","Cienaga",[46,3,4])
 regla(13,"ceramica argentina","Santa Maria",[74,8,9])
 regla(14,"ceramica argentina","Llajta Mauca o Sunchituyoc",
 [16,17,18,19,20])
 regla(15,"ceramica argentina","Llajta Mauca o Sunchituyoc",
 [16,17,18,19,20])
 regla(16,"ceramica argentina","Calingasta",[21,22,23])
 regla(17,"ceramica argentina","Sanagasta, Aimogasta o Angualasto",
 [24,8,25,26,27])
 regla(18,"ceramica argentina","Sanagasta, Aimogasta o Angualasto",
 [24,28,25,26,27])
 regla(19,"ceramica argentina","Sanagasta, estilo San Jose",
 [29,8,30,31,32])
 regla(20,"ceramica argentina","Sanagasta, estilo San Jose",
 [29,8,30,31,33])
 regla(21,"ceramica argentina","Belen",[34,17,8,27,35,36,37])
 regla(22,"ceramica argentina","Tafi",[38,39])
 regla(23,"ceramica argentina","Tafi",[38,40])
 regla(24,"ceramica argentina","La Candelaria",[41,8,42,43,44,45])
 regla(25,"ceramica argentina","San Francisco",[47,48,49,50])
 regla(26,"ceramica argentina","Condorhuasi
 Policromo",[34,51,52,53])
 regla(27,"ceramica argentina","Condorhuasi Monocromo
 Rojo",[34,54,55])
 regla(28,"ceramica argentina","Condorhuasi Tricolor",[34,56,57,58])
 regla(29,"ceramica argentina","Condorhuasi Rojo sobre Ante",
 [34,44,59])
 regla(30,"ceramica argentina","El Alamito",[60,61,62])
 regla(31,"ceramica argentina","Las Mercedes",[16,3,4])
 regla(32,"ceramica argentina","La Aguada",[63,22,64])
 regla(33,"ceramica argentina","La Aguada Bicolor",[63,64,65])
 regla(34,"ceramica argentina","La Isla",[66,67,68,70])
 regla(35,"ceramica argentina","La Isla",[66,69,68,70])
 regla(36,"ceramica argentina","La Isla",[66,71,68,70])
 regla(37,"ceramica argentina","Alfarcito Policromo",[66,71,72])
 regla(38,"ceramica argentina","Alfarcito Policromo",[66,67,72])
 regla(39,"ceramica argentina","Alfarcito Policromo",[66,73,72])
 regla(40,"ceramica argentina","Famabalasto",[74,75,76])
 regla(41,"ceramica argentina","Yavi",[66,19,77])
 regla(42,"ceramica argentina","Yocavil Policromo o Rojo sobre
 Blanco",

[74,73,78,79])
 regla(43,"ceramica argentina","Mancapa o Averias",[16,80,81,82])
 regla(44,"ceramica argentina","Mancapa o Averias",[16,80,81,83])
 cons("ceramica argentina","Fuente: Rex Gonzalez y J. Perez,
 passim.")
 cons("Condorhuasi Monocromo Rojo","Cf. Ethnos, v.22, n.4, pp.18-
 21")
 cond(1,"la pieza procede de los andes centrales")
 cond(2,"la pieza procede de territorio argentino")
 cond(3,"la superficie es de color gris")
 cond(4,"posee decoración incisa")
 cond(5,"tiene forma de arballo")
 cond(6,"la superficie es de color negro")
 cond(7,"posee representaciones figurativas")
 cond(8,"la forma es la de una urna funeraria")
 cond(9,"la decoración incluye motivos oritomorfos")
 cond(10,"posee asa en forma de estribo")
 cond(11,"posee asa vertedera")
 cond(12,"la pieza procede del altiplano")
 cond(13,"tiene fondo plano")
 cond(14,"incluye representaciones de cóndores")
 cond(15,"incluye representaciones de máscaras")
 cond(16,"procede de la provincia de Santiago del Estero")
 cond(17,"la pasta es rojiza o amarillenta, cocida en horno abierto")
 cond(18,"la superficie es amarillenta, muy brillante y bien pulida")
 cond(19,"los motivos decorativos están pintados en negro")
 cond(20,"los motivos consisten en una o más aves estilizadas")
 cond(21,"procede de la provincia de San Juan")
 cond(22,"la pieza es de color gris, con decoración incisa")
 cond(23,"el motivo prevaleciente es en espina de pescado o
 espigado")
 cond(24,"se localiza entre La Rioja y el SO de San Juan")
 cond(25,"la decoración es pobre, sin figuras zoomorfas ni
 antropomorfas")
 cond(26,"los motivos se disponen en paneles")
 cond(27,"la decoración está pintada en negro sobre el rojo del
 fondo")
 cond(28,"la forma es de puco con boca amplia y base pequeña")
 cond(29,"procede del valle de Abaucán")
 cond(30,"incluye motivos de serpientes y pisadas de puma")
 cond(31,"el cuello de la urna es muy largo")
 cond(32,"hay caras en relieve en el cuello de la pieza")
 cond(33,"en el cuello hay personajes en relieve tocando flautas de
 pan")
 cond(34,"se localiza en Catamarca y La Rioja")
 cond(35,"la base es un cono truncado, el cuerpo es globular y el
 cuello cilíndrico")
 cond(36,"tiene asas horizontales eventualmente con figuras
 antropomorfas")
 cond(37,"los motivos son geométricos, dispuestos en tres bandas")
 cond(38,"procede de la provincia de Tucumán")
 cond(39,"la cerámica es de tipo tosco, sin decoración y paredes
 alisadas")
 cond(40,"está pintada de rojo, sin decoración")
 cond(41,"procede del este y sur de Salta o norte de Tucumán")
 cond(42,"la decoración no es pintada; hay guardas geométricas en
 torno
 al cuello")
 cond(43,"el color de la superficie es gris rojizo o negrusco")
 cond(44,"las paredes de la pieza son sumamente delgadas")
 cond(45,"hay aplicaciones o saliencias bulbosas")
 cond(46,"procede de valles calchaquíes, Catamarca, Rioja o San
 Juan")
 cond(47,"procede del este de Jujuy o de la provincia de Salta")
 cond(48,"las piezas son gris-negras con decoración incisa")
 cond(49,"las formas son troncocónicas o globulares")
 cond(50,"las asas están modeladas con motivos zoomorfos")
 cond(51,"la superficie es pulida, de color rojizo")

cond(52,"hay guardas geométricas pintadas en negro con orla blanca")
 cond(53,"la forma es de figura humana, sentada o gateando")
 cond(54,"el color es rojo uniforme, eventualmente morado")
 cond(55,"la forma es de cuerpo globular con cuello cilíndrico")
 cond(56,"la forma es cilíndrica o subcilíndrica")
 cond(57,"la superficie está cubierta por engobe blanco-crema")
 cond(58,"los motivos son figuras geométricas y escalonados")
 cond(59,"hay líneas o triángulos rojos; la técnica sugiere pintura negativa")
 cond(61,"la cerámica es tosca")
 cond(60,"procede de la provincia de Catamarca")
 cond(62,"la decoración consiste en bandas verticales rojas, violáceas o negras")
 cond(63,"procede de Catamarca, La Rioja o San Juan")
 cond(64,"los motivos son geométricos o representan figuras felínicas")
 cond(65,"los dibujos son negros sobre fondo amarillento rojizo")
 cond(66,"procede de la Puna o de la Quebrada de Humahuaca")
 cond(67,"es un jarro en forma de reloj de arena")
 cond(68,"hay caras con ojos oblicuos modeladas en las paredes de los vasos")
 cond(69,"es un timbal con asa lateral")

cond(70,"está pintada de blanco con decoración en negro")
 cond(71,"la pieza es una olla de cuerpo globular con asas verticales")
 cond(72,"hay triángulos negros con orla blanca, sobre fondo rojizo oscuro")
 cond(73,"posee forma de puco o kero")
 cond(74,"procede de Tucumán, Salta o Catamarca")
 cond(75,"la forma es de puco, olla de cuerpo globular o kero cilíndrico")
 cond(76,"la decoración consiste en manos pintadas en negro sobre rojo")
 cond(77,"hay reticulados, círculos, triángulos o espirales de líneas muy finas")
 cond(78,"la decoración interior está dividida en cuatro paneles en cruz")
 cond(79,"los colores son rojo y negro subido sobre blanco espeso")
 cond(80,"la forma es de puco, kero o bol semiglobular")
 cond(81,"está pintada en negro o rojo brillante sobre engobe blanco")
 cond(82,"hay líneas oblicuas paralelas en series rojas y negras")
 cond(83,"hay manos pintadas con un triángulo y líneas paralelas")
 excluyentes([46,74,16,21,24,29,34,38,41,47,60,63,66])
 excluyentes([3,17,43])
 similares([74,38,41])

SISTEMA DE AYUDA (Help)

Sistema Experto AGENI3 - Programado por Carlos Reynoso, 1990.

Consulta del sistema de Help

PgUp, Ctl-PgUp : scroll hacia arriba, posicionamiento al inicio.

PgDn, Ctl-PuDn : scroll hacia abajo, posicionamiento al final.

Esc : fin de consulta.

Ctl-F3 : búsqueda de ristra de caracteres o frases.

Shift-F3 : repite búsqueda.

F5 : Zoom, un-zoom.

Shift-F10 : Redimensionar y posicionar ventana.

Indice (H)

Actualización del conocimiento (A)

Alarma (W)

Carga de base de conocimientos en memoria (X)

Consulta (C)

Descripción y comparación de entidades (D)

Diagnóstico y evaluación (C)

Edición de la base de conocimientos (E)

Frames (R)

Impresión de la base de conocimientos (I)

Metarreglas (R)

Tokens y palabras (T)

Especificaciones técnicas

Actualización del Conocimiento (A)

Existen tres niveles de actualización de la base de conocimientos. En el primero se introducen las categorías y subcategorías y las condiciones que las definen. En el segundo se agrega información adicional referida a las categorías o clases. En el tercero se añaden especificaciones referidas a las entidades terminales. La segunda y tercera opción pueden interpretarse como una especificación de consecuencias (acciones a seguir, comentarios, tratamientos).

La estructura de los diferentes niveles responde a la siguiente configuración:

regla(número, categoría gral, categoría particular, [cond1, ... condn])
condición(número, descripción)
cons(clase, [especificación_1, ... especificación_n])
cons(cat.particular, [especificación_1, ... especificación_n])

Se puede acceder a la descripción de consecuencias mediante el menú de comparaciones (D), haciendo un mapa interpretado de la memoria (M) o como resultado de un diagnóstico o consulta (C).

Nunca es necesario preocuparse por la asignación de los números identificatorios. El sistema maneja internamente dichas funciones.

Cada nombre de categoría, subcategoría o condición puede medir hasta 127 caracteres. Si se introduce una condición idéntica a otra ya ingresada, el sistema considera que se trata de la misma y le asigna el mismo número. Se pueden introducir cláusulas mutuamente negadas, y es conveniente hacerlo para ajustar las relaciones entre entidades: afirmar la condición "tiene esqueleto interno", por ejemplo, desactiva la cláusula "no tiene esqueleto interno". En general, si existe la cláusula afirmada correspondiente, el sistema no pregunta por la versión negada.

Se sugiere introducir datos organizados jerárquicamente, de modo que toda categoría (a excepción de la primera) haya sido subcategoría en una definición anterior. La sesión de actualización agrega reglas y condiciones a las que ya estuvieran asertadas en la memoria, numerándolas consecutivamente. Al final de la sesión (que se comanda introduciendo una categoría en blanco) el sistema permite optar por el resguardo o no de la información actualizada.

Si la nueva definición no se resguarda, se puede consultar de todas maneras en la misma sesión de diagnosis, en tanto no se blanquee la memoria o no se cargue una base de conocimientos adicional.

En la presente versión del programa no es posible eliminar ni modificar condiciones o reglas, salvo manualmente, editando la base de conocimientos (Opcion "E" del Menú Principal). Si se interviene manualmente sobre dichas bases, se deberá preservar la numeración secuencial de reglas y condiciones, ya que debido a su tratamiento recursivo no es conveniente que falte ningún número entre el primero y el último.

Al guardar la base de conocimientos se deberá indicar un nombre aceptable por DOS, de ocho dígitos de nombre y tres de extensión (DBA), separados por un punto. El sistema controla solamente la longitud de la cadena (nombre + "." + extensión).

Alarma (W)

El aviso de alarma advierte con un patrón sonoro aleatorio la ejecución de comando contextualmente impropios. El mismo patrón se produce cuando el proceso de diagnóstico da con una respuesta. La alarma puede desactivarse y activarse desde el menú principal. En estado normal la alarma se encuentra activada.

Carga de base de conocimientos en memoria (X)

El sistema permite escoger bases de conocimiento cuyo nombre de archivo DOS finalice con extensión DBA. El programa verifica la adecuación de la estructura de los datos. Antes de cargar una base de conocimientos en memoria, los datos que hubiere anteriormente son eliminados. No es posible cargar bases heterogéneas simultáneamente, ya que la referencia

interna entre sus entidades se realiza en base a su numeración.

El proceso de carga de base de conocimientos muestra los archivos disponibles que poseen la extensión .DBA en el directorio local. Se puede recorrer todo el directorio del disco subiendo por los indicadores \. Es posible asimismo presionar <F4> para modificar disco, path y máscara de extensión del nombre del archivo.

Consulta (C)

La consulta se encuentra totalmente orientada por menús. En el primer menú se agrega "fin de consulta" como dominio elegible, a fin de enfatizar la correspondencia lógica entre estructuras de inferencia y procedimientos en el lenguaje PROLOG.

Si en una consulta el sistema responde que carece de información suficiente, habrá que cargar una base de conocimientos en la memoria (X) o actualizar la información en forma manual (A).

Cada consulta de una condición admite cuatro respuestas: "si", "no", "es posible" y "por qué?". Se pueden seleccionar las respuestas con el cursor o la letra inicial. Si se indica al sistema que "es posible", se deberá indicar la posibilidad, probabilidad o factor de confianza de la respuesta, tanto sea sobre una ponderación numérica como en base a una aproximación cualitativa.

Si se pregunta al sistema "por qué?", el programa presenta una explicación de la hipótesis que se está tratando de demostrar y de los pasos ya establecidos por la consulta, en caso que los hubiere. Con la tecla <F5> se realiza o se desactiva el Zoom sobre la ventana de explicaciones, así como sobre toda otra ventana del programa. Con <Esc> se termina la lectura de la explicación.

Descripción y comparación de entidades (D)

Se selecciona con menú el nombre de una entidad, finalizándose la totalidad de la consulta con la tecla <Esc>. Con las teclas de cursor puede accederse a entidades u objetos que no alcanzan a representarse en la pantalla del menú o en la ventana de descripciones.

En caso de seleccionar otra entidad sin salir de la rutina, el sistema agrega la descripción resultante al final del archivo de maniobra, de manera que las descripciones de las sucesivas entidades pueden verse simultáneamente en pantalla. Se puede escoger un número indefinido de entidades en tanto haya lugar en el soporte para escribir el archivo correspondiente. Cada nueva sesión global de consulta borra las comparaciones anteriormente acumuladas.

Dentro de la pantalla de descripción están activadas las mismas teclas y funciones de scrolling, zoom, un-zoom, redimensión de ventanas y búsqueda que valen para este archivo de Help.

Diagnóstico y evaluación (C)

En caso de no poder formalizar un diagnóstico, el sistema señala si existe alguna condición positiva cumplida por alguna entidad. Las respuestas posibles equivalen lógicamente a las afirmativas, por más que su probabilidad sea baja.

El sistema calcula el factor de confianza para cada condición por separado en función del número total de condiciones de la categoría específica, presuponiendo una confianza absoluta en las restantes (hipótesis de independencia), y luego evalúa el arrastre de las posibilidades en el

conjunto del proceso de diagnosis (hipótesis de interdependencia). De este modo, la posibilidad menor al 100% de una condición aislada se disuelve conforme aumenta el número de condiciones, mientras que la probabilidad o confianza de un diagnóstico se decrementa de acuerdo con el número de condiciones inciertas y su margen de probabilidad.

Edición de la base de conocimientos (E)

El sistema incluye un poderoso sistema de edición, incorporando las funciones del editor de Borland adaptadas a la gestión sobre los archivos de conocimiento. Si se desea utilizar el editor sobre otros archivos, modificar provisionalmente el nombre de la extensión del archivo a editar como *.DBA, o presionar <F4> en la ventana correspondiente para modificar disco, path y máscara de nombre del archivo.

La edición de la base no implica modificación de los contenidos de memoria. En la pantalla de edición (que se puede ampliar con <F5> o mover y redimensionar con <Shift-F10>) los datos aparecen tal cual están guardados, sin interpretación alguna. Para observar los datos interpretados se deberá escoger la opción de Imagen de Memoria ("M") del Menú Principal.

Teclas Funciones de edición

F1 : Help y funciones automáticas.
Ctl-F3 : Buscar.
Shift-F3 : Repetir búsqueda.
F4 : Reemplazar
Shift-F4 : Repetir reemplazo.
Ctl-F5 : Copiar bloque.
F6 : Ver otras ventanas del programa.
Alt-F6 : Mover bloque.
F7 : Editor auxiliar (para copiar bloques de otro archivo).
Shift-F10 : Redimensionar y posicionar ventana de edición.

Frames (R)

Un frame es una estructura de representación del conocimiento en la cual cada ítem de una entidad está categorizado. Cada categoría de una entidad se denomina slot.

El sistema de tratamiento de frames se ha incorporado como parte de la definición de metarreglas. Para transformar el conjunto de datos representados en un frame se deben especificar las condiciones con una palabra clave en el momento de actualizar la base de conocimientos. Luego debe ejecutarse la opción "Frames" del menú de metarreglas, con lo cual se eliminan otras modalidades de interexclusión fuera de los que determinan los nombres de los slots.

La rutina de construcción del frame aniquila las definiciones de reglas excluyentes comunes y construye un esquema transitorio utilizando la primera palabra de cada condición como token de clave. Este esquema puede resguardarse luego junto con la totalidad de la base de conocimientos, de manera que el agregado de nuevas reglas y condiciones puede integrarse al frame volviendo a ejecutar la rutina correspondiente.

Por ejemplo, si se agrega al frente de la condición una palabra-token (especie, origen, período, clase) y luego se ejecuta la rutina para la construcción del frame, las condiciones precedidas por la misma palabra quedan automáticamente interexcluidas en caso de aserción positiva de cualquiera de ellas.

Dado que el tratamiento de frames modifica radicalmente las estrategias de diagnóstico, se aconseja especial cuidado en la escritura de las con-

diciones, evitando diferencias de puntuación y ortografía entre tokens correspondientes al mismo slot o categoría. El programa considera que el primer token de una condición es la ristra de caracteres que precede al primer espacio.

El procedimiento de generación de frames incluye un menú de elecciones múltiples que permite excluir condiciones que comienzan con palabras que no se desea asociar a un slot. Si se escoge una o más palabras iniciales, esta(s) no será(n) considerada(s) interexcluyente(s). La exclusión se activa o desactiva con <Enter> y la totalidad de la lista se indica con la tecla de función <F10>. Si no se desea excluir ningún slot, presionar <Esc> o <F10>.

Se aconseja asimismo no repetir los nombres de slot en distintos niveles de la organización jerárquica. Por ejemplo, si se prefija una entidad de alto nivel con el token "origen", no es conveniente repetir la misma palabra en la caracterización de un hipónimo o subordinado, aunque se refiera a una especificación de detalle del mismo concepto.

Impresión de la base de conocimientos (I)

La base de conocimientos que está cargada en memoria se puede escribir sobre una impresora en forma interpretada. Un menú permite seleccionar entre LPT1, LPT2, COM1, PRN y un archivo convencionalmente llamado KBPRINT.PRN. Si el dispositivo de impresión no está disponible, el sistema operativo de encarga de presentar opciones para cancelar, reintentar, ignorar o seguir.

Para imprimir la base de conocimientos en formato real, copiar (fuera del programa) el archivo *.DBA correspondiente sobre un dispositivo de impresión (LPT1, PRN, etc).

Metarreglas (R)

Hay dos tipos de relaciones entre reglas: interexclusión (la afirmación de una condición, con cualquier grado de certidumbre, excluye las condiciones que pertenezcan al mismo conjunto) y similitud (el sistema indica la existencia de entidades que posean una o más condiciones similares a la de la respuesta correcta en modalidad de consulta). Lógicamente no es necesario crear otras formas de interrelación.

La aserción de conjuntos de reglas excluyentes tiene por efecto evitar que el sistema realice preguntas que quedan sistemáticamente negadas por una consulta positiva anterior. Son excluyentes, por ejemplo, las condiciones "es originario de América" y "es originario de Asia"; si no se indica que lo son, el sistema preguntará por la segunda aún cuando se haya respondido positivamente a la primera. Las reglas excluyentes que complementan a la asertada se eliminan transitoriamente de la memoria en un proceso de consulta, restaurándose cuando el mismo finaliza.

Las reglas interexcluyentes se seleccionan mediante un menú múltiple marcando o desmarcando con <Enter> las diversas entidades y definiendo cada conjunto con <F10>.

La especificación de conjuntos de reglas similares ofrece al sistema la posibilidad de realizar una indicación aproximativa de casos similares en la eventualidad de un diagnóstico acertado. Esta indicación no altera el curso de los procesos de inferencia, sino que simplemente proporciona indicios sobre la articulación del dominio. El proceso de selección de reglas similares es el mismo que para el caso de las reglas interexcluyentes.

Se ha previsto un tratamiento especial para las condiciones excluyentes articuladas en forma de Frame. Véase el ítem correspondiente en este

sistema de Help.

Tokens y palabras (T)

Esta rutina permite rastrear palabras sueltas a través de toda la base de conocimientos, ya sea entre las reglas, las condiciones o las consecuencias. La palabra o token puede introducirse en mayúscula o minúscula, con o sin acentos. Se admiten también números, en tanto figuren en alguno de los textos especificados. Las palabras siguientes al primer espacio no serán tenidas en cuenta. Se considera que los signos no alfabéticos (paréntesis, guiones, barras, asteriscos) delimitan tokens diferentes.

Especificaciones técnicas

Diseño y Programación: Carlos Reynoso (CONICET-U.B.A.)
Lenguaje fuente: PROLOG (Borland Turbo Prolog, v.2.0, 1989).
Requerimiento mínimo de memoria: 512 Kb (programa + base).
Uso máximo de memoria: 640 Kb menos Sistema Operativo.
Archivo máximo de Help o Edición: 64 Kb.
Longitud máxima de línea de texto: 127 caracteres.
Tamaño máximo de base de conocimientos: ca. 384 Kb.