ALEJANDRO SANTORUM	VARELA	44090946-5	Gr. 320 (DG)
2	e entrega e	JERCICIOS I.A.	
[1] a) Formalizacio	in base de	conocimiento	•
I) Vm 7 Iqual (II) Vn [7 Iqual (n, 1) III) 7 Par (1) 1	$) \implies \exists m \exists$	= -1 Par(4)	
IV) Vn [7Par(suc(n)) V) Vn [Par(suc(n)) VI) Vn [Igual(sun	$\Rightarrow Par(n)$ $\Rightarrow \neg Par(n)$ $(n,0), n)$,7	c (0))
VII) Vn,m [Igual (s		~	(adicienal por el enunciado)
b) (onversión a f I) Vm [¬Iqual(o, • Eliminación	suc(m))] del V: 719	ual (0, suc(m)) [1]	
II) Vn (Tqual (n.0) Eliminación + reducción amb	=> Im Ique Jel =>: Vn vito 7 Vn Iqual(n	d (n, suc(m)) [Iqual(n,0) V = Jm 10) V = Iqual(n, suc) 1) V = Iqual(n, suc)	Iqual (n, suc(m))] (c (f(n))) (f(n))) [2]
II) TPar (1) A Igual [V) Vn [TPar (suc(n)) Eliminación + reducción Eliminación	=> Par(n)] del => 1	TPar(1) [3] TPar (suc(0)) [3] In [Par (suc(n)) V Par (suc(n)) V Par(n)	x(n)

```
V) Vn | Par (suc(n)) => 7Par (n) |
   · Eliminación del ⇒: ∀n [¬Par (suc(n)) ∨ ¬Par (n)]
   · Eliminación de V: | 7Par(suc(n)) v 7Par(n) [6]
VI) Vn [Igual (sum (n,0), n)]
     · Eliminación del V: [Igual(sum(n,0),n] [7]
VII) Vn,m [Igual (suc (sum (n,m)), sum (n, suc (m)))
     · Eliminación del V: Igual (suc (sum (n,m)), sum (n, suc (m))) [8]
VIII) Vrim (Iqual(n,m) 1 Par(m)) => Par(n)
      · Eliminación del =>: Vn,m [7 (Iqual(n,m) ~ Par(m)) v Par(n)]
      · Ley de De Morgan: Vn,m (-Igual(n,m) v rPar(m) v Par(vi))
      · Eliminación del V: TIgual(nim) v TPar(m) v Par(n) [9]
c) Demostrar que la suma de un nº natural par con uno es impar.
   \forall n \mid Par(n) \Rightarrow \gamma Par(sum(n, suc(0)))
                                          (META)
  Negamos la META:
   TYN [Par(n) => TPar (sum (n, succo)))
  Y la transformamos en FNC:
     · Eliminación del =>: TYN [TPar(n) V TPar (sum (n, suc(o)))]
    · Ambito 7: In Par(n) 1 Par (sum (n, suc co))
     · Skolemnitación: Par (sk1) 1 Par (sum (sk1, suc (01))
                     Par (SK1) [10]
                      Par (sum (SK1, suc(0)) [11]
```

```
[SV] (0) (0) \Lambda (((0) f) \Lambda ((0) f) \Lambda [Si) O = V [Si) O = V
de Green le amole el O
                                                mpos
       of purco
                                                                                                                                                                   \frac{3}{12} \frac{1}{|u:=0|} \frac{1}{|u
                            [4] [ May (...) [24]
                                                                   [5] | RES | Par (suc(n)) V | Touch (n, suc(p(n))) V Pes (n) [13]
                                                                        Thar (n) v Igual (n, suc(fin)) v 1202 (n) [m]
                                                                                                  1 far (n) v 1 far (fcn) v hes (n) [40]
                                                                                                                            · Elim V+ Respuestaln) truco de Green:
                                                            An [ - Par(n) v (n) rear(f(n)) n tqual(n, suc (f(n)))]
                                                                                                                                                           · Amhito_negaciones + distributiva:
                                                            NEG META: 7 = [ [Rus(n) A (7 = mer) A Igual(n, suc(m))]] + FNC
                         d) Teveo de Green, determinar si existe alquin ruimero ratural es.

par que no sea sucesor de un ruimero impar y cual es.

NETA: In [Par(n) \ (13m (1Par(m) \ Igual (n, suc(m)))]]
                                                                                      [27]
[27]
[27]
[27]
[27]
                                                                                                                                                                                              [13] (ch) [or (suc) [14]
                                                                                                                                                     [8] | equiv. Par (suc (sum (skr,0))) [13]
                                                                                                                                                                           [6] RES | TRES | [10] | N:= SKT | [15]
```

Litizanos ahora resolución:

Añadimos a la base de conocimiento inicial la negación de la siguiente meta: $\forall x,y \left[\left(P(x) \wedge P(y) \wedge 7 \right] z \quad \text{Suc}(x,z) \wedge 7 \right] \forall w(y,w) \Longrightarrow (x=y) \right]$ Llegamos a que el 0 es el único newtro.

```
[2.] Resolucion de un problema con resolución + refutación:
       <u>Jer</u> paso: Indicar constantes, predicados y funciones usados
                                          Greucho (persona)
      CONSTANTES:
                                 X_1, X_2, \dots (personas)
      VARIABLES:
                                        Cs, Cz,... (clubs)
     PREDICADOS: SC(xs,xz): evalúa a verdadero si Xz es como Xz.
                                          SM(X1,C1): evalua a verdadero si X1 es miembro de C1
                                        QSM (X1, C1): evalua a verdadero si X1 quiere ser miembro de C1
                                        A (Cs, Xs): evalua a verdadero si Cs ha aceptado/acepta a Xs
 2º paso: Formalización de la base de conocimiento
    I) \x SC(xix)
 I) \forall x,c \mid SM(x,c) \iff (QSM(x,c) \land A(c,x))
III) Groucho ya es una constante en el ambito de personas
IV) \forall x_1 \in \left[ \left( SC(x, Grancho) \wedge A(c, x) \right) \implies 7QSM(Groucho, c) \right]
3er paso: Conversion a FNC
   I) [SC(x,x) [4]
I) \forall x_i c \ \left[ SM(x_i c) \Rightarrow (QSM(x_i c) \land A(c_i x)) \right] \land \left[ (QSM(x_i c) \land A(c_i x)) \Rightarrow SM(x_i c) \right] \right\}
           VX,C \[\langle \left[\langle SM(x,c) \right\ \left(\alpha \right) \right] \Langle \left[\langle \alpha \right] \left(\langle \alpha \right) \right] \langle \left(\langle \alpha \right) \right] \langle \left(\langle \alpha \right) \right] \langle \left(\alpha \right) \right] \langle \left(\alpha \right) \right) \langle \left(\alpha \right) \langle \langle \left(\alpha \right) \langle \langle \left(\alpha \right) \langle \langle
          (7SM(xic) V QSM(xic)) A(7SM(xic) VA((ix))) A (7QSM(xic) V 7A((ix)) V SM(
      Entonces:
                7 SM(x,c) v QSM(x,c) [2]
               75M(x,c) v A(c,x) [3]
                  TQSM(x,c) V TA(c,x) V SM(x,c)
```

III) Yx,c \(\gamma \text{SC}(x, Groucho) \) \(\gamma \text{A(c,x)} \) \(\gamma \text{QSM}(Groucho, c) \)
TSC(X, Groucho) V 7A(C, X) V TQSM (Groucho, C) [5]
1º pa80: Formalizarnos meta, la negamos y la convertimos a FNC
META: Vc 75M (Groucho, C)
META NEGADA IC SM (Groudho, C)
FNC: SM (Groucho, SK) [6]
7º paso: Resolución + refutación buscando la clánsula vacía 3] x:= Groucho 6] RES _{SM} A (SKc, Groucho) [7]
2] X:= Groucho 6] RESSM QSM (Groucho, SKc) [8]
i) x:= Groudo c:= SKc 7A(SKc, Groudo) v SM(x, c) [9] RESON
F] x:= Groucho C:= 5Kc SM (Groucho, SKc) [10] (inutil, ya la tinia) T KESA SM (Groucho, SKc) [10]
] X:=Groudo 7A(c, Groudo) V 7QSM (Groudo, C) [M]
i] c:= SKc 1A(SKc, Grouelo) [12] I] RESOST 1A(SKc, Grouelo) [12]
1] RESOSTI 1] RESOSTI 2] RESOSTI NO es miembro de ningún club.

```
[3.] Resolución de un problema con resolución + refutación:

1er paso: Indicar constantes, variables, predicados y funciones
    CONSTANTES:
   VARIABLES: X,y, Z (números reales)
   PREDICADOS: Iqual(xiy): evalúa a T si x=y
Neutro(x): evalúa a T si x es el elem. neutro
de la suma
  FUNCIONES: Resta(XIY): evalua al número (X-Y).
\frac{2^{\circ} \text{ paso}}{\text{J}}: Formalización de la base de conocimiento 
 J) \forall x,y [Iqual (Resta (x,y), 0) \iff Iqual (x,y)]
  II) \forall x \left[ \text{Neutro}(x) \iff \left( \forall y \text{ Janal } \left( \text{Resta}(y, x), y \right) \right) \right] =
          = \forall \text{X,y} \[ Neutro (x) \leftrightarrow \text{Fqual (Resta (y,x),y)} \]
3^{er} paso: Conversion a FNC

I) \forall x,y \mid (\text{Iqual}(\text{Resta}(x,y),0) \Rightarrow \text{Iqual}(x,y)) \wedge (\text{Iqual}(x,y) \Rightarrow \text{Iqual}(\text{Resta}(x,y),0))
  Vx,yd[7Iqual(Resta(x,y),0) v Iqual(x,y)],[7Iqual(x,y) v Iqual(Resta(x,y),0)])
      1 Igual (Resta(XIY), 0) V Igual (XIY) [1]
      7 Igual (Xiy) v Igual (Resta (Xiy), 0) [2]
I) VX,y [(Neutro(x) => Iqual(Restaly,x),y)) \( (Iqual(Restaly,x),y) => Neutro(x))]
 Vx,y of [n Neutro(x) v Iqual(Resta(y,x),y)] n [n Iqual(Resta(y,x),y) v Neutro(x)] o
     7 Neutro (x) v Igual (Resta (y,x),y) [3]
     7 Iqual (Resta (y1x)1y) V Neutro (x) [4]
```

º paso: Formalizamos la meta, la negamos y la convertimos a FNE $\forall x [Neutro(x) \implies Igual(x,0)]$ META NEGADA: $\exists x \mid \neg (\text{Neutro}(x) \Rightarrow \text{Igual}(x,0))$ 3x \[\(\gamma \left(\gamma \text{Neutro}(x) \) \] \[\text{Igual}(x_10) \] Ix (Neutro(x) 1 TIgual (x,0)) 1-Tqual(SK10) Neutro (SK) [5] Igual (Resta (y,x), y) 7 Igual (Resta(SK,0), 0) - Igual (Resta (x,y), 0) v Igual (x,y) Igual (Resta (X14), y) 1] | y:=0 (x,0) 7] | RES [qual (x,0) à con sustitución y:=0 podemos hacer resolución y obtener a Igual (Resta(x,0),0)? demostrar que no hay ningun elemento neutre aparte del cero.

```
[4.]
Primero programaremos un predicado que evalue a verdadero
```

rnmero programaremos un predicado que evalue a verdadero auando se le pase el mínimo de le lista, e.d. list-min (L,A) evalua a verdadero sui A es el mínimo elemento de L:

list_min ([LILS], Min): - list_min (LS, L, Min).

list_min ([], Min, Min).

list-min ([LILS], MinO, Min):-

Mind is min (L, Mind),

list-min (LS, Min1, Min).

Otro predicado auxiliar delete-one (A, L, Ls) evalúa a verdadero sii

A es el elemento eliminado de la lista L, resultando en Ls:

delete-one (_, [], []).

delete-one (Term, [Term|Tail], Tail).

delete-one (Term, [Head|Tail], [Head|Desult]):
delete-one (Term, Tail, Desult).

Ahora ya podemos hacer los predicados propuestos:
minimo (Ls, A, Ys): - list-min (Xs, A), delete-one (A, Ls, Ys).

ordenar ([], []).
ordenar (L, [MIRS]): - minimo (L, M, Rest), ordenar (Rest, Rs).

5. Vamos a crear dos predicados auxiliares antes de find: indexOf(Ls, Elem, Index) evaluia a verdadero si la primera aparición de Elem en LS es Index (empezando en 1). Por otro lado, checkFirstOccur (LS, E, N, Rest) evalua a verdadero si, la primera aparicion del elemento E en ls es en el indice N y la lista restante es Rest: index Of ([Element], Element, 1):-!. index Of [[ITail], Element, Index):index of (Tail, Element, Index 1), Index is Index1+4. checkFirstOccur (WholeList, Elem, Pos, Rest):chec KFirst Occur (Whole List, Elem, Pos, Rest, Whole List).

checkFirstOccur (WholeList, Elen, Pos, R, [Elem[R]):indexOf (Whole List Elem, Pos). check FirstOccur (WholeList, Elem, Pos, Rest, [-182]):checkFirstOccur (WholeList, Elen, Pos, Rest, R2).

find(Ls, E, Rs): - find(Ls, E, Rs, O). find(Ls, E, [], -):- \+ member(E, Ls). find (Ls, E, [NIM], Acum):check First Occur (Ls, E, Naux, Rest), N is Naux + Acum, find (Rest, E, M, N).

[6.] Para este ejercicio vamos a crear un predicado auxilia Mamado check Elem (Ls, I, Elem) que evaluta a verdadero sii I esta presente en Ls en el indice I. check Elem ([Elem] -], 1, Elem).

check Elem ([-|Rest], I, Elem):
New I is I-1,

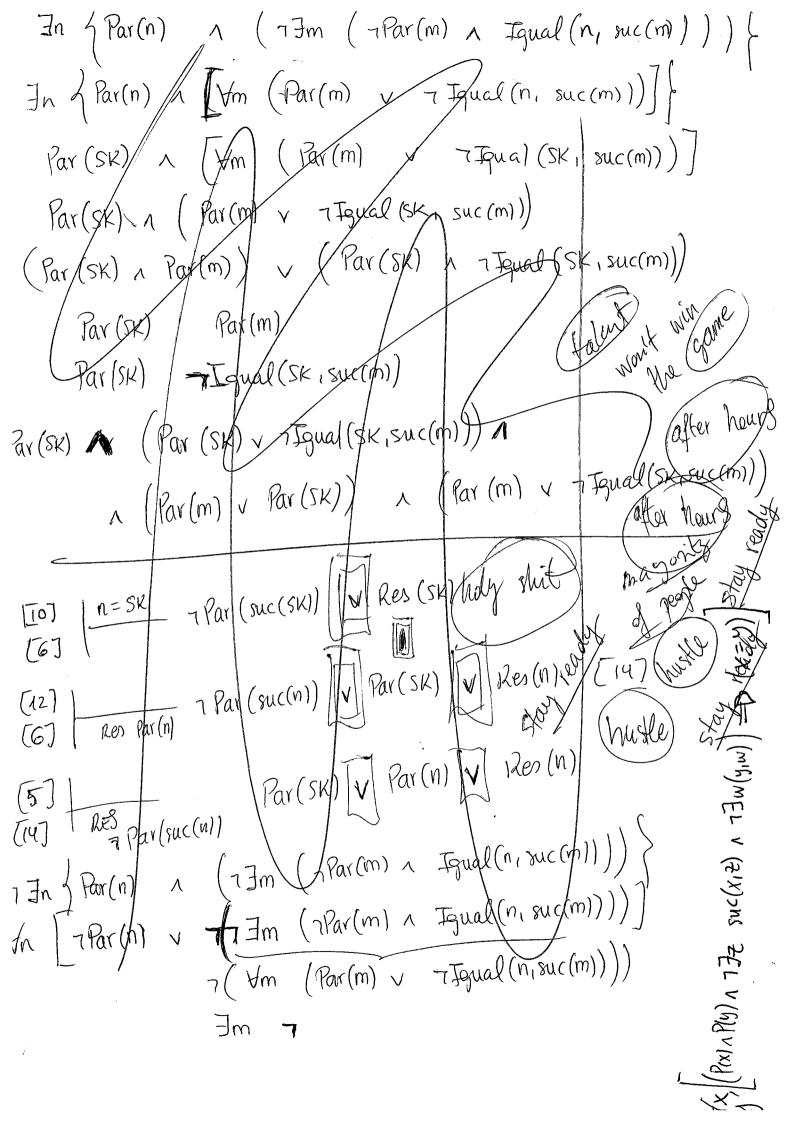
check Elem (Rest, New I, Elem).

seleccionar (-, [], []).

seleccionar (Ls, [IIIs], [RIRS]):
checkElem (Ls, I, R),

seleccionar (Ls, Is, Rs).

7. (omo no podía ser de otra forma, para este ejercicio vamos a implementar un predicado auxiliar checklep(L, R, Rest) que evaluará a T sii L aparece al principio de R y Rest est el resto de R, sin L. checklep([], R, R). checklep([], R, R). checklep([], R, R). checklep([Lls], [LIRs], Rest):- checklep([s, Rs, Rest) replicar(Ls, N, Rs):- checklep([s, Rs, Rest), NewN is N-1, replicar(Ls, NewN, Rest).



[1] Tqual(0, suc(m))

[2] Igual (n,0) v Igual (n, suc (f(n)))

(3) 7 Par (suclo)

[4] Igual (1, suc (0))

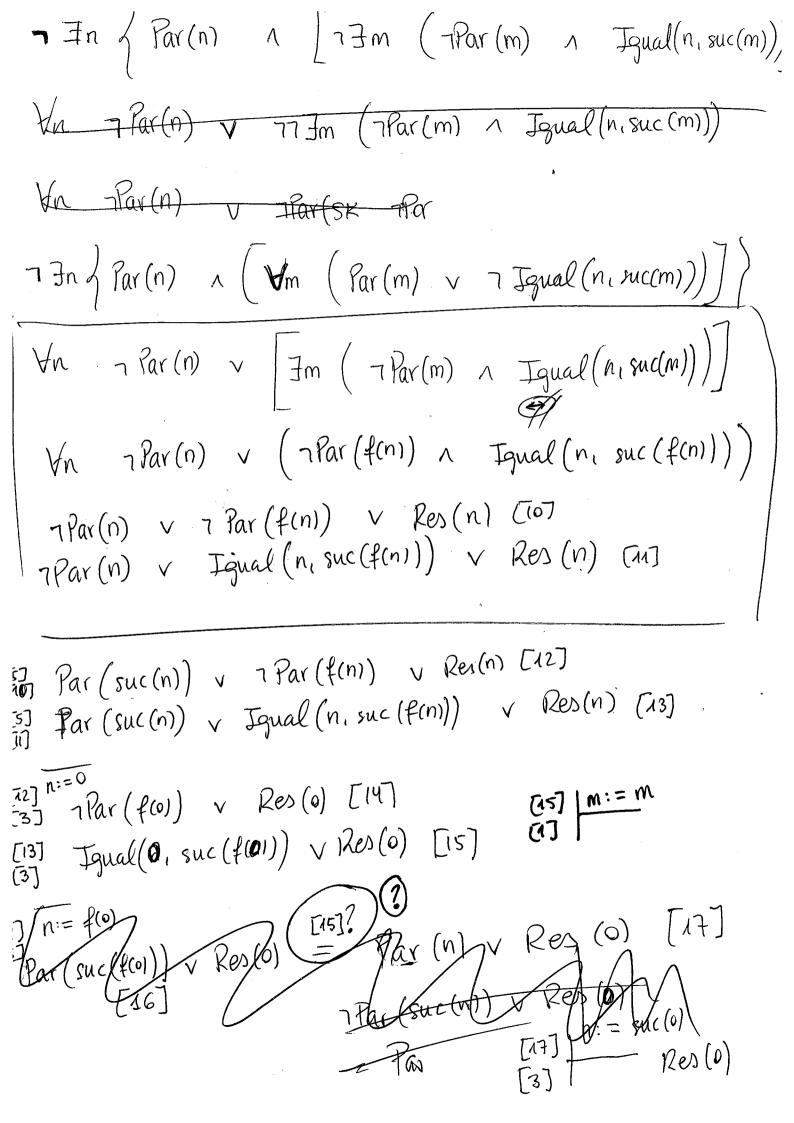
[5] Par(suc(n)) v Par(n)

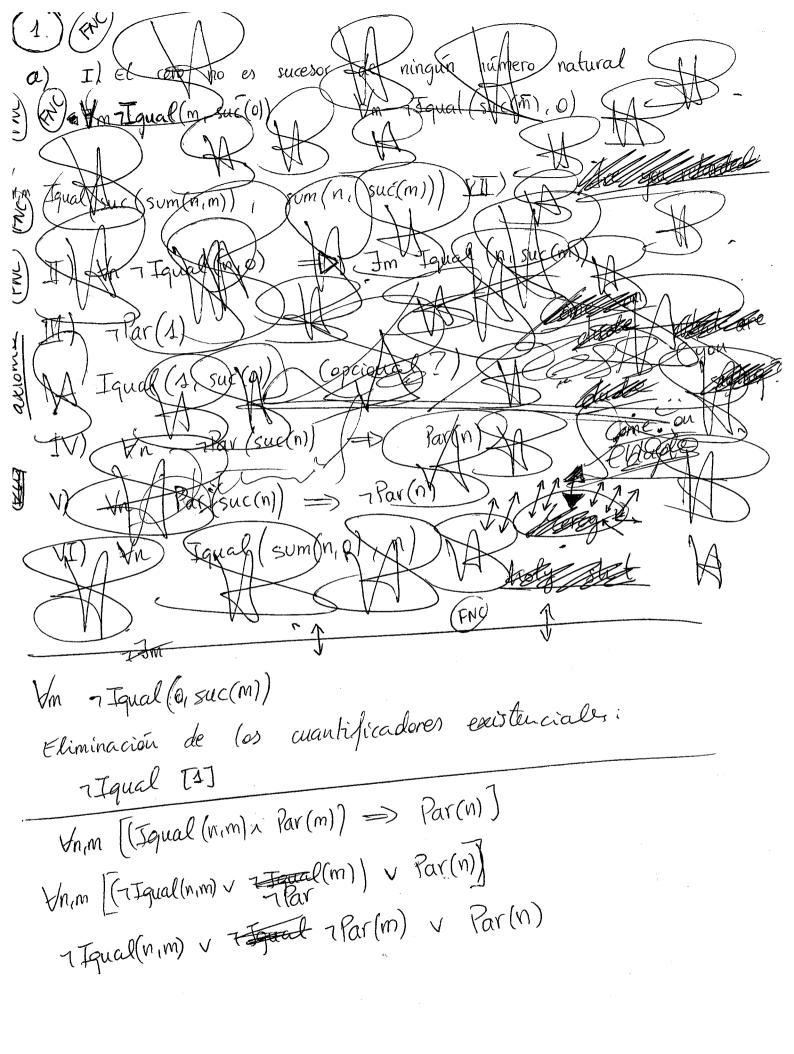
[6] 7Par(suc(n)) v 7Par(n)

[7] Igual (svm(n,0),n)

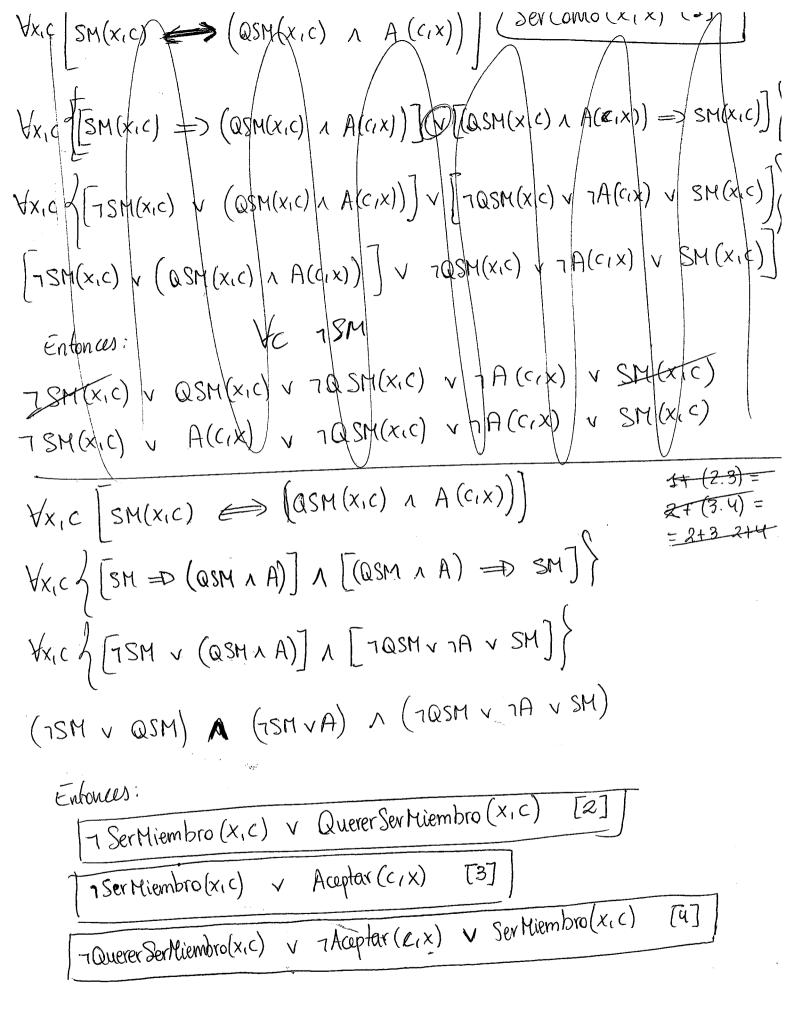
[8] Iqual(suc(sum(n,m)), sum(n, suc(m)))

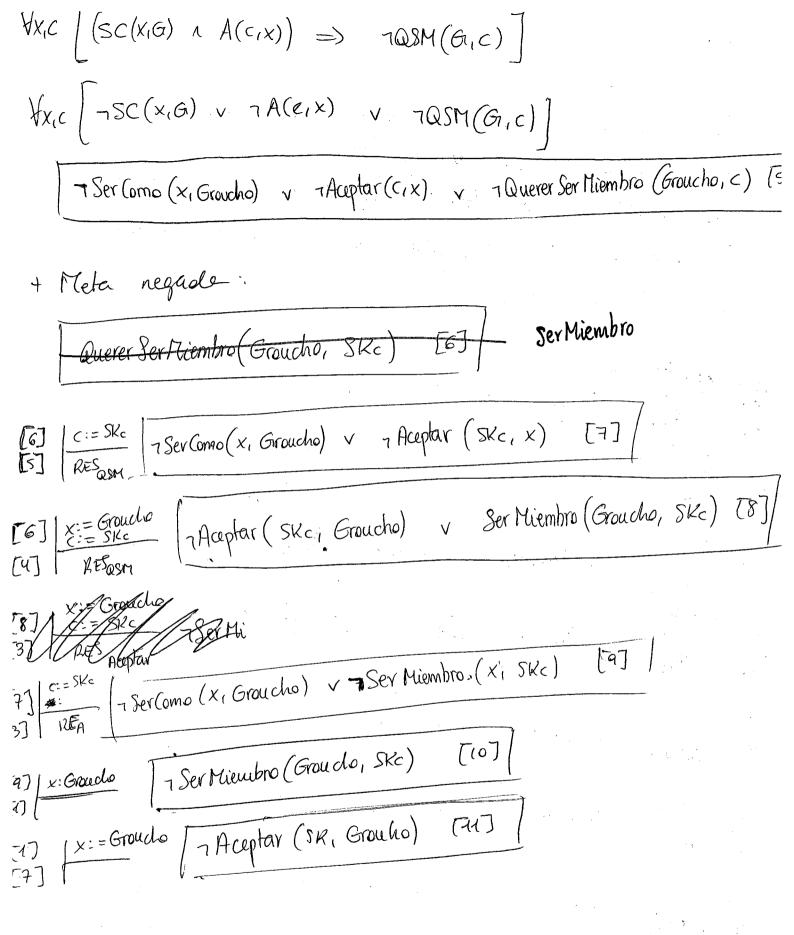
[9] Tgual (n,m) v TPar(m) v Par(n)





Xy: personas X1, X2, personas
Groucho: una persona concreta (constante)
Cs, Cz, dubs
Ser Como (X1, X2): Evalua T ni Xi es como X2 Ser Miembro (X1, C1): (X1 es miembro de C1
Querer Per Miembra (VI. CI): XI quieve ser miembre de CI
Aceptar (C1, X1): (1 ha aceptado Xs // C1 acepta Xs
2.I) $\forall x \in Ser(omo(x,x)] \cup \mathcal{W}$ 2.II) $\forall x, c \in SerMiembro(x, c) \iff (Querer Ser Miembro(x, c) \land Aceptar(C, x))$
le de finicion al
2. IV) LC, X (Ser (omo (X, Groucho)) 1 Aceptar (C,X)) =>
=> 7 Querer Ser Miembro (Groudho, C)] F3
Jes Der Miembro (Groucho, C) 7 Ser Miembro (Groudo, C)
A Chinary XIII (Chinary Constitution of the co
FNC: Quiere Ser Miembro (Groucho, SKC) Ser Miembro (Gr, SKC)
Pasamos a PNC le base de conocimiento:
I) Ser (omo (x,x) [4]
I) TSer (omo (xix) [4] [1) Yx,c (Ser Miembro (xic) A Querer Ser Miembro (xic) Acaplar (cix)) V (- Overer Sort Tiembro (xic) V7 Acaptar (cix)))
V (- Ser Miembro (XiC) / (record of the service) \ \ \(\tag{SM(XiC)} \) \ \ \(\tag{SM(XiC)} \) \ \ \(\tag{SM(XiC)} \) \ \(\tag{SM(XiC)} \) \\ \(\tag{SM(XiC)} \) \) \\ \(\tag{SM(XiC)} \) \(\tag{SM(XiC)} \) \\ \(\tag{SM(XiC)} \) \(\tag{SM(XiC)} \) \\ \(\tag{SM(XiC)} \) \(\tag{SM(XiC)} \) \(\tag{SM(XiC)} \) \\ \(\tag{SM(XiC)} \) \(\





Δ :	**
[1] Ser Como (x,x)	
[z] 7 Ser Miembro (x,c) V Querer Ser Miembro (x,C)	
[3] 7 Ser Miembro(x,c) v Aceptar (e,x)	
TIT TOSM(x c) , TAEWHON (C(X) V SET COUNTY	
(5) 7 Ser Como (x,G) V 7 Aeaptar (C,X) V 1000 (C)	
[6] SerMiembro (G, SKc)	
[6] X:= Groudio [3] Res sn Aceptar (SKc, G) [7]	
[6] X:= SKC QSM (G, SKC) [8]	
[8] x:= G C:= SKC -A (SKc,G) v SM (x,c) [9] 4] 1285asm	
[9] RESA SM (G1, SKc) [10] (inutil) ya la Gente [7] RESA	'- ,
[1] X==G (S) Ser(omo TA(C,G) V TQSM(G,C) [11]	
[11] C:= SKC 1000 A(SKC, G) [12] [8] QSM (SKC, G) [12]	es
[12] QSM Grouche [12] RESA Demostrado Serouche alquien que miembro de nino club.	ro e

constantes: 0 X: números reales Igual (X14): evalue T si x=y - Resta (x14): e Nautro (x): evalue 7 Resta(x,y): evalua al número (x-y) I) I Resta (xiy) Vxy Igual (Resta(xiy), 0) > Iqual(xiy) I) \(\text{\text{X}} \) Neutro(\text{\text{X}}) \(\lefta \) \(\text{\text{Yy}} \) Igual \(\text{Resta}(y, \text{\text{X}}), \(\begin{align*} \begin{align*} \text{Yy} \] \(\text{Yy} \) \\ \end{align*} $\forall x \in Neutro(x) \Rightarrow Iqual(x,0)$ NEG META: Jx / 7 (Newtro(x) => Iqual (x,0)) FNC META: $\exists x \left[7 \left(\neg Neutro(x) \vee Iqual(x,0) \right) \right]$ Ix (Neutro(x) 1 rIqual(x,0)/ Neutro (SK) 1 7 Igual (SK,0) Vx,y (Iqual(Resta(x,y),0) => Iqual(x,y)) / (Iqual(x,y) => Iqual(Resta(x,y),0)) try of [-Igual(Resta(X14),0) v Igual(X14)] 1 [-Igual(X14) v Igual(Resta(X14),0)] 7 Igual (Resta(X, y), 0) v Igual (X, y) [1] /7 Iqual(x,y) v Iqual (Resta(x,y),0) (2)

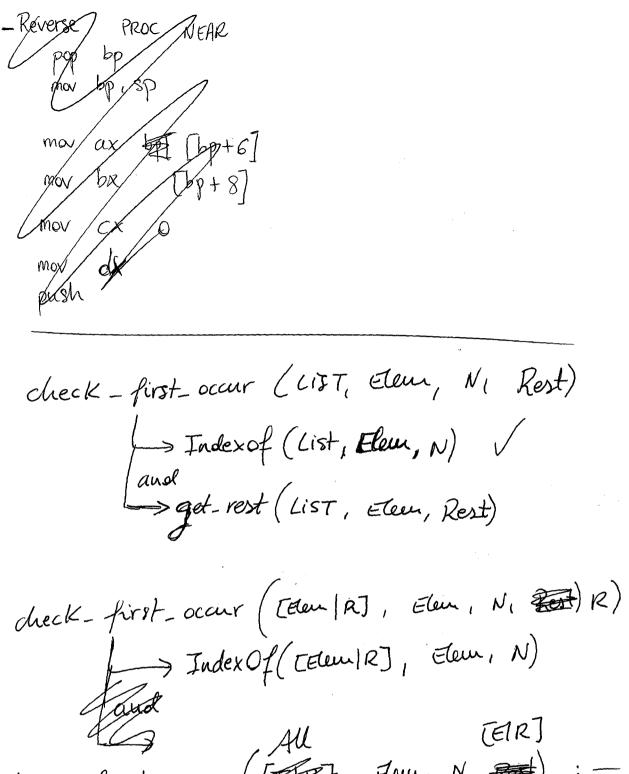
fx of [Neutro(x) => (by Iqual(Resta(x,y),y))/((by Iqual(Resta(x,y),y)) => => Neutro(x)] } Vx of [- Neutro(x) v (by Iqual (Resta(x14), y))] 1 [(7 by Iqual (Resta(x14)), y))] 1 v Neutro (x)] "Un nº es neutro sii al restarlo de cualquier otro x da x". Pregunta: à son equivalentes? Vx [Neutro(x) => (Yy Iqual(Kesta(y,x), y))] Vx,y [Neutro(x) => Iqual (Resta(y,x),y)] $\forall x, y \mid (\text{Neutro}(x) \Rightarrow \text{Igual}(\text{Resta}(y, x), y)) \wedge (\text{Igual}(\text{Resta}(y, x), y) \Rightarrow \text{Neutro}(x)) \mid$ Yx,y [-Neutro(x) v Iqual (Resta (y,x),y)] 1 [7 Iqual (Resta (y,x),y) v Neutro(x)] 6 TNeutro(x) v Igual (Resta (9, x), y) Tiqual (Resta (4,x),y) v Neutro(x) [4] Renmen: + Meta negade: [1] - Igual (Resta (x14),0) V Igual (X14) Iqual (Resta (x,y), 0) v 7 Iqual (x,y) TNeutro(x) v Iqual (Resta (x)x), y) v Neutro(x) [4] TIqual (Resta(y,x), y) [5] Neutro (SK) [6] 7 Igual (SK,0)

find (LS, E, RS) first ocar [1,2,3,4], 2, 2 first_occur ([30,20,30,40], 20, 2, [30,40])

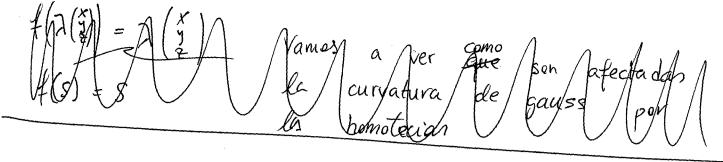
list elem pos rest after element

find ([El Rest], E, [NIM]) Laud find (Rest, E, M)

find $(L, E, C]):= a \frac{deck(L, E)}{not member(E, L)}$



theck-first-occur (EER), Hem, N, Est):wheck-first-occur ([



seleccionar (Ls, [IIIs], [RIRS])

check (Ls, I, R),

seleccionar (Ls, Is, Rs).

seleccionar (Ls, [], []).

check ([Elen], 1, I)

check ([Elen], 1, Elen).

check ([ElTail], I, Elen):
new is [-1,

check (Tail, new E, Elen).

replicar (LS, N, RS)

check (Ls, Rs) Rest comprobamos si New N is N-1, replicar (Ls, New N, Rest). Principio de RS replicar (Ls, New N, Rest). y devolvemos" el resto de Rs en Rest

replicar(Ls, o, []).

à check (Ls, Rs, Rest)?

quita:

replicar(Ls,0,[]):-!.

para evitar budes

infinitos en

segundos intentos.

aso

check (Ls, Rs, Rest)

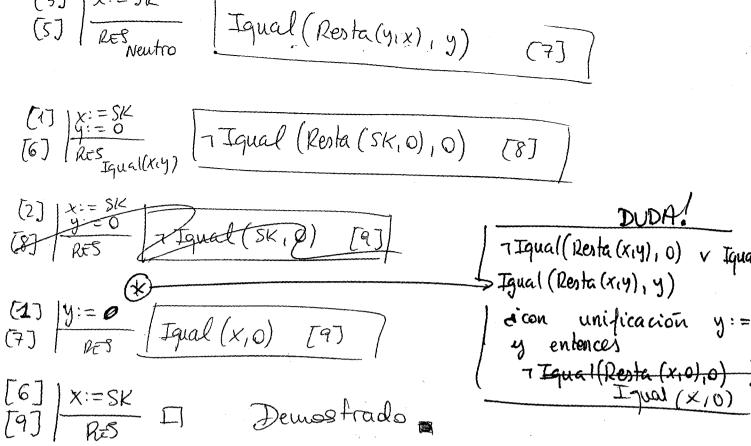
check ([LILS], [LIRS], RS):-

deck([], -, -).

mejora

check ([LILs], [LIRs], Rest):check (Ls, Rs, Rest).

deck([], R, R).



(5,2,5] ((3,2,5)), ordeugr ((3,2,5), ps) ([s's'z] / [s't'z's]) maps (SZ (SZ) KE) Min-aux is to min (Min-aux) (A,0MM, [SIR], MMO, A) · (A, A, []) wim-tow (x) List-min ([FIR], A):- List-min (R, F, A). ominim b es A ins T suichet (4,21) minimo (s) / 57) owinim 57

[T'S'E']

(PROLOG) d'Que hace exactamente el "=" y el "is"? Puede ayudar el ejemplo de las diapositivas con: cosecutivos (NIM) :- N is M+1 consecutivos 2 (NIM) :- N = M+1 [RES+REF] Cuando intentamos demostrar cierta FIBF, la negamos y la introducimos en la base de conocimiento. Si llegamos a la cláusula vacia entonces la meta (sin negar) era conse-cuencia lógico de la base de conocimiento... pero, si no llegamos à la clausula vacia, à quiere esto de cir que era = o que no tenemos evidencia alguna de fuese o no F? s como los contrastes de hipótesis en estadístico [EJERCICIOS 1] ise pærmite cierto razonamiento informal? ETERLICIOS] -> predicado Iqual La hay que interpretarlo como una iqualdad maternatica y por lo tanto podemos sustituir los términos que el predicado dice que intercambiar son iquals o hay que basar la resolución únicamente en la grafia? inifición en GENERAL] Algoritmo de unificación custitución a la iza vaniable := algo algo constante variable evaluación de una funcion i f(constant)? if(variable)? ERcicios] Duda ejercicio 3 marcada en verde Prolog eu general