|  |  |  |  |
| --- | --- | --- | --- |
| **Alumno(s):** | Marco Adrian Rodriguez Lopez | **Fecha:** | 20 / Nov / 2017 |

|  |
| --- |
| OBJETIVO |
| Implementar los diferentes programas echos en prolog y fomentar el desarrollo de los programas. |

|  |
| --- |
| MARCO TEÓRICO |
| **Prolog**  es un lenguaje de programación lógico e interpretado usado habitualmente en el campo de la Inteligencia artificial.  Lenguaje declarativo (opuesto a procedimentales) basado en reglas de la lógica.  En el cual se hicimos un programa en prolog de una carta de un restaurante donde estuvimos probando como funcionaba Prolog. |

|  |
| --- |
| DESARROLLO |
| Fueron 11 programas hechos en prolog y los cuales son los siguientes:  1. MONTY PYTHON AND THE HOLY GRAIL (1975)  El programa siguiente es saber sobre un a bruja si es mujer y el cual se quemara.    witch(X) :- burns(X),female(X).  burns(X) :- wooden(X).  wooden(X) :- floats(X).  floats(X) :- sameweight(duck, X).  female(girl).  sameweight(duck,girl).  ? witch(girl).    2. CANCIÓN ESTÚPIDA  Es encontrar una cancion y escribir en el nombre de la cancion.  cancionestupida(0):-nl,write('Gomo ya no gueda shevvezza, -hic- be boy a doddmig...').  cancionestupida(N):-N>1,nl,write(N),write(' botellas de cerveza en el suelo'),nl,  write(N),write(' botellas de cerveza'),nl,  write('Cojo una y me la bebo'),nl,  A is N-1, cancionestupida(A).  cancionestupida(N):-N=1,nl,write(N),write(' bodellia de shegvezza en el zsduelo'),nl,  write(N),write(' bodella de segbezha'),nl,  write('La gojo y be la bhebo'),nl,  A is N-1, cancionestupida(A).  3. DIAGNÓSTICO DE ENFERMEDADES  Determinar si el paciente cuneta con alguna enfermedad, el sintoma y .  enfermo\_de(manuel,gripe).  tiene\_sintoma(alicia,cansancio).  sintoma\_de(fiebre,gripe).  sintoma\_de(tos,gripe).  sintoma\_de(cansancio,anemia).  elimina(vitaminas,cansancio).  elimina(aspirinas,fiebre).  elimina(jarabe,tos).  recetar\_a(X,Y):-enfermo\_de(Y,A),alivia(X,A).  alivia(X,Y):-elimina(X,A),sintoma\_de(A,Y).  enfermo\_de(X,Y):-tiene\_sintoma(X,Z),sintoma\_de(Z,Y).  4. PLANIFICADOR DE GASTOS  Programar el viaje hacia donde sera y cual seta el alojamiento que tendra.  transporte(roma,20000).  transporte(londres,25000).  transporte(tunez,15000).  alojamiento(hotel,roma,25000).  alojamiento(hotel,londres,15000).  alojamiento(hotel,tunez,10000).  alojamiento(hostal,roma,15000).  alojamiento(hostal,londres,10000).  alojamiento(hostal,tunez,8000).  alojamiento(camping,roma,10000).  alojamiento(camping,londres,5000).  alojamiento(camping,tunez,5000).  viaje(W,X,Y,Z):-transporte(W,A),alojamiento(Y,W,C),B is C\*X,Z is A+B.  5. ÁRBOL GENEALÓGICO  Mostar el arbol genealogico de la familia lo que es el padre y el ancestro del padre.  marco. padre(celso,marco).  padre(celso,mayra).  padre(abraham,celso).  padre(marco,alvaro).  padre(X,Y).  ancestro(X,Y):-padre(X,Y).  ancestro(X,Y):-padre(X,Z),padre(Z,Y).  6. RED SEMÁNTICA  Determinar los tipos de animales y que fue tipo y que es lo tiene o no tiene.  marc(pajaro,animal).  marc(pinguino,pajaro).  marc(mamifero,animal).  marc(reptil,animal).  marc(arboles,vegetal).  marc(pinguinomagico,fantasia).  marc(pinguinomagico,pajaro).  has(pajaro,volar).  hasnot(pinguino,volar).  has(pajaro,pluma).  has(reptil,ponehuevos).  has(pajaro,ponehuevos).  es(X,Y):-marc(X,Y).  es(X,Y):-marc(X,Z),es(Z,Y).  tiene(X,Y):-has(X,Y).  tiene(X,Y):-marc(X,Z),tiene(Z,Y),not(hasnot(X,Y)).  7. HORÓSCOPO  Encontrar cual es el horoscopo de acuerdo al me, dia y nombre.  /\* horoscopo(Signo,DiaIni,MesIni,DiaFin,MesFin)  <- son del signo Signo los nacidos entre el DiaIni/MesIni y el DiaFin/MesFin \*/  horoscopo(aries,21,3,21,4).  horoscopo(tauro,21,4,21,5).  horoscopo(geminis,21,5,21,6).  horoscopo(cancer,21,6,21,7).  horoscopo(leo,21,7,21,8).  horoscopo(virgo,21,8,21,9).  horoscopo(libra,21,9,21,10).  horoscopo(escorpio,21,10,21,11).  horoscopo(sagitario,21,11,21,12).  horoscopo(capricornio,21,12,21,1).  horoscopo(acuario,21,1,21,2).  horoscopo(piscis,21,2,21,3).  /\* signo(Dia,Mes,Signo) <- los nacidos el Dia/Mes pertenecen al signo Signo \*/  signo(Dia,Mes,Signo) :- horoscopo(Signo,D1,M1,D2,M2),  ( (Mes=M1,Dia>=D1) ; (Mes=M2,Dia=<D2) ).  ?- signo(8,5,tauro).  ?- signo(7,8,Signo).  ?- signo(7,X,Signo).  ?- signo(X,7,Signo).  /\* mes(Mes,Dias,Nombre) <- el mes Nombre, de número Mes tiene Dias \*/  mes(1,31,enero).  mes(2,28,febrero).  mes(3,31,marzo).  mes(4,30,abril).  mes(5,31,mayo).  mes(6,30,junio).  mes(7,31,julio).  mes(8,31,agosto).  mes(9,30,septiembre).  mes(10,31,octubre).  mes(11,30,noviembre).  mes(12,31,diciembre).  signo(Dia,Mes,Signo) :- horoscopo(Signo,D1,M1,D2,M2),  ( (Mes=M1,Dia>=D1,mes(M1,D,\_),Dia=<D) ;  (Mes=M2,Dia=<D2, Dia>0) ).  8. OPERADOR DE CORTE  Hacer una sumatoria desdeel 1 y ver cuales pertencen a los numeros naturales y sacar el cociente de la sumatoria.  /\* sumatorio(Num,Sum) <- Sum es el sumatorio desde 1 hasta Num \*/  sumatorio(1,1) :- !.  sumatorio(N,S) :- N1 is N-1,  sumatorio(N1,S1),  S is N+S1.  /\* natural(Num) <- Num es un número perteneciente a los Naturales \*/  natural(0).  natural(X) :- natural(Y),  X is Y+1.  /\* diventera(Dividendo,Divisor,Cociente) <- Cociente es el resultado de la división \*/  diventera(A,B,C) :- natural(C),  Y1 is C\*B,  Y2 is (C+1)\*B,  9. MANEJO DE LISTAS  Crear una lista de acuerdo al miembro y concetenar ea lista, y mostrar el elemento de la ultima lista, invertir la lista, borrar los elementos, insertar un elemento, sustituir un elemento.  /\* miembro(Elem,Lista) <- el término Elem pertenece a la lista Lista \*/  miembro(X,[X|\_]).  miembro(X,[\_|Y]) :- miembro(X,Y).  /\* nel(Lista,N) <- el número de elementos de la lista Lista es N \*/  nel([],0).  nel([\_|Y],N) :- nel(Y,M),  N is M+1.  /\* es\_lista(Lista) <- Lista es una lista \*/  es\_lista([]).  es\_lista([\_|\_]).  /\* concatena(L1,L2,L3) <- concatenación de las listas L1 y L2 dando lugar a L3 \*/  concatena([],L,L).  concatena([X|L1],L2,[X|L3]) :- concatena(L1,L2,L3).  /\* ultimo(Elem,Lista) <- Elem es el último elemento de la lista Lista \*/  ultimo(X,[X]).  ultimo(X,[\_|Y]) :- ultimo(X,Y).  /\* inversa(Lista,Inver) <- Inver es la inversa de la lista Lista \*/  inversa([],[]).  inversa([X|Y],L) :- inversa(Y,Z),  concatena(Z,[X],L).  /\* borrar(Elem,L1,L2) <- se borra el elemento Elem de la lista L1 obteniendose L2 \*/  borrar(X,[X|Y],Y).  borrar(X,[Z|L],[Z|M]) :- borrar(X,L,M).  /\* subconjunto(L1,L2) <- la lista L1 es un subconjunto de la lista L2 \*/  subconjunto([X|Y],Z) :- miembro(X,Z),  subconjunto(Y,Z).  subconjunto([],\_).  /\* insertar(Elem,L1,L2) <- se inserta el elemento Elem en la lista L1 obteniendose L2 \*/  insertar(E,L,[E|L]).  insertar(E,[X|Y],[X|Z]) :- insertar(E,Y,Z).  /\* permutacion(L1,L2) <- la lista L2 es una permutación de la lista L1 \*/  permutacion([],[]).  permutacion([X|Y],Z) :- permutacion(Y,L),  insertar(X,L,Z).  /\* sust(E1,E2,L1,L2) <- L2 es L1 sustituyendo las ocurrencias del elemento E1 por E2 \*/  sust(\_,\_,[],[]).  sust(E1,E2,[E1|L1],[E2|L2]) :- !, sust(E1,E2,L1,L2).  sust(E1,E2,[Y|L1],[Y|L2]) :- sust(E1,E2,L1,L2).  /\* union(L1,L2,L3) <- L3 es la lista-conjunto unión de L1 y L2 \*/  union([],L,L).  union([X|L1],L2,L3) :- miembro(X,L2), !,  union(L1,L2,L3).  union([X|L1],L2,[X|L3]) :- union(L1,L2,L3).  ?- miembro(d,[a,b,c,d,e]).  ?- miembro(d,[a,b,c,[d,e]]).  ?- miembro(d,[a,b,c]).  ?- miembro(E,[a,b,[c,d]]).  ?- nel([a,b,[c,d],e],N).  ?- es\_lista([a,b,[c,d],e]).  ?- concatena([a,b,c],[d,e],L).  ?- concatena([a,b,c],L,[a,b,c,d,e]).  ?- concatena(L1,L2,[a,b]).  /\* miembro(Elem,Lista,Pos) <- el término Elem pertenece a la lista Lista y  ocupa la posición Pos \*/  miembro(X,[X|\_],1).  miembro(X,[\_|Y],N) :- miembro(X,Y,N1),  N is N1+1.  /\* borrarN(Pos,L1,L2) <- se borra el elemento Elem que ocupa la posición  Pos de la lista L1 obteniendose la lista L2 \*/  borrarN(1,[\_|Y],Y).  borrarN(N,[Z|L],[Z|M]) :- N1 is N-1,  borrarN(N1,L,M).  10. ANALIZADOR MORFOLÓGICO  Hacer un de analisis de varios cosas y representarlos en modelo.  % modelo “perro”  analisis(perro,nms,o). -- nombre masculino singular  analisis(perro,nmp,os). -- nombre masculino plural  analisis(perro,nfs,a). -- nombre femenino singular  analisis(perro,nfp,as). -- nombre femenino plural  %  modelo(perr,perro).  modelo(niñ,perro).  % modelo “señor”  analisis(señor,nms,’’). -- nombre masculino singular  analisis(señor,nmp,es). -- nombre masculino plural  analisis(señor,nfs,a).  analisis(señor,nfp,as).  %  modelo(señor,señor).  % modelo “balón”  analisis(balón,nms,’’). -- nombre masculino singular  analisis(balón,nmp,es). -- nombre masculino plural  %  modelo(balón,señor).  modelo(balcón,señor).  % modelo “cantar”  analisis(cantar,presenteindicativo1s,o). -- 1ª persona singular presente indicativo  analisis(cantar,presenteindicativo2s,as). -- 1ª persona singular presente indicativo  analisis(cantar,presenteindicativo3s,a). -- 1ª persona singular presente indicativo  …  analisis(cantar,futurosimple1s,aré). -- 1ª persona singular futuro simple indic.  …  modelo(cant,cantar).  modelo(am,cantar).  modelo(gust,cantar).  modelo(estudi,cantar).  % modelo “temer”  analisis(temer,presenteindicativo1s,o). -- 1ª persona singular presente indicativo  analisis(temer,presenteindicativo2s,es). -- 1ª persona singular presente indicativo  analisis(temer,presenteindicativo3s,e). -- 1ª persona singular presente indicativo  …  modelo(tem,temer).  modelo(beb,temer).  modelo(com,temer).  analisismorfologico(soy,presenteindicativo1s).  analisismorfologico(fui,presenteindicativo1s).  11. JUEGO DE LÓGICA  Hacer un juego basado en la logica de conocimientos.  /\*---- JUEGO DE LÓGICA ----\*/  mensaje :- nl,write('Ejemplo "Juego Lógico" cargado.'),nl,  write('Se lanza con ?- iniciar.'),  nl,nl.  /\*---- PROGRAMA PRINCIPAL ----\*/  /\* iniciar <- llamada inicial del programa \*/  iniciar :- write('Base de Conocimientos: '),  read(BC),  consult(BC),!,  nl,write('Base de Conocimientos '),write(BC),  write(' consultada'),nl,nl,  numeroPropiedades(N),  objetosUniverso(M),  iniciar(N,M).  iniciar :- nl,write('ERROR: Base de Conocimientos no encontrada'),nl.  iniciar(2,M) :- !,ini(M,[],[]).  iniciar(3,M) :- !,ini(M,[],[],[]).  iniciar(4,M) :- !,ini(M,[],[],[],[]).  iniciar(5,M) :- !,ini(M,[],[],[],[],[]).  iniciar(N,\_) :- nl,write('ERROR: Número de Propiedades incorrecto = '),  write(N),nl.  /\* ini(Sol1,Sol2,...) <- Sol1 es una lista con los objetos del dominio 1,  Sol2 la lista con los objetos del dominio 2, ...  con las soluciones respectivamente relacionadas. \*/  /\* Correlacionar 2 propiedades \*/  ini(M,L1,L2) :- nel(L1,M),escribir(L1,L2),nl,pausa,fail.  ini(M,L1,L2) :- r1(Obj1,Obj2),  nopertenece(Obj1,L1),  nopertenece(Obj2,L2),  ini(M,[Obj1|L1],[Obj2|L2]).  /\* Correlacionar 3 propiedades \*/  ini(M,L1,L2,L3) :- nel(L1,M),escribir(L1,L2,L3),nl,pausa,fail.  ini(M,L1,L2,L3) :- r1(Obj1,Obj2),  nopertenece(Obj1,L1),  nopertenece(Obj2,L2),  r2(Obj1,Obj3),  nopertenece(Obj3,L3),  r3(Obj2,Obj3),  ini(M,[Obj1|L1],[Obj2|L2],[Obj3|L3]).  /\* Correlacionar 4 propiedades \*/  ini(M,L1,L2,L3,L4) :- nel(L1,M),escribir(L1,L2,L3,L4),nl,pausa,fail.  ini(M,L1,L2,L3,L4) :- r1(Obj1,Obj2),  nopertenece(Obj1,L1),  nopertenece(Obj2,L2),  r2(Obj1,Obj3),  nopertenece(Obj3,L3),  r3(Obj1,Obj4),  nopertenece(Obj4,L4),  r4(Obj2,Obj3),  r5(Obj2,Obj4),  r6(Obj3,Obj4),  ini(M,[Obj1|L1],[Obj2|L2],[Obj3|L3],[Obj4|L4]).  /\* Correlacionar 5 propiedades \*/  ini(M,L1,L2,L3,L4,L5) :- nel(L1,M),escribir(L1,L2,L3,L4,L5),nl,pausa,fail.  ini(M,L1,L2,L3,L4,L5) :- r1(Obj1,Obj2),  nopertenece(Obj1,L1),  nopertenece(Obj2,L2),  r2(Obj1,Obj3),  nopertenece(Obj3,L3),  r3(Obj1,Obj4),  nopertenece(Obj4,L4),  r4(Obj1,Obj5),  nopertenece(Obj5,L5),  r5(Obj2,Obj3),  r6(Obj2,Obj4),  r7(Obj2,Obj5),  r8(Obj3,Obj4),  r9(Obj3,Obj5),  r10(Obj4,Obj5),  ini(M,[Obj1|L1],[Obj2|L2],[Obj3|L3],[Obj4|L4],[Obj5|L5]).  /\*---- RUTINAS GENERALES ----\*/  /\* escribir(Lista1,Lista2,...) <- escribe las soluciones correlacionadas  de las listas: Lista1, Lista2 ... \*/  escribir([],[]).  escribir([Obj1|Resto1],[Obj2|Resto2]) :-  write(Obj1), write(' - '),write(Obj2),nl,  escribir(Resto1,Resto2).  escribir([],[],[]).  escribir([Obj1|Resto1],[Obj2|Resto2],[Obj3|Resto3]) :-  write(Obj1), write(' - '),write(Obj2),  write(' - '), write(Obj3),nl,  escribir(Resto1,Resto2,Resto3).  escribir([],[],[],[]).  escribir([Obj1|Resto1],[Obj2|Resto2],[Obj3|Resto3],[Obj4|Resto4]) :-  write(Obj1), write(' - '),write(Obj2),  write(' - '), write(Obj3),write(' - '),write(Obj4),nl,  escribir(Resto1,Resto2,Resto3,Resto4).  escribir([],[],[],[],[]).  escribir([Obj1|Resto1],[Obj2|Resto2],[Obj3|Resto3],[Obj4|Resto4],[Obj5|Resto5]) :-  write(Obj1), write(' - '),write(Obj2),write(' - '),  write(Obj3),write(' - '),write(Obj4),write(' - '),  write(Obj5),nl,  escribir(Resto1,Resto2,Resto3,Resto4,Resto5).  /\* pausa <- detiene la ejecución del programa hasta que se pulse una tecla \*/  pausa :- write('Pulsa <return> para buscar otra solucion'),  skip(10),nl.  /\*---- RUTINAS DE MANEJO DE LISTAS ----\*/  /\* nopertenece(Elem,Lista)) <- el elemento Elem no pertenece a la lista Lista \*/  nopertenece(\_,[]).  nopertenece(E,[X|L]) :- E\=X,  nopertenece(E,L).  /\* nel(Lista,N) <- el número de elementos de la lista Lista es N \*/  nel([],0).  nel([\_|L],N) :- nel(L,M),  N is M+1.  :- mensaje.  /\*---- BASE DE CONOCIMIENTOS ----\*/  numeroPropiedades(3).  objetosUniverso(3).  /\*- PROPIEDADES -\*/  /\* prof(Profesor) <- Profesor es el nombre de un profesor \*/  prof(maria).  prof(jesus).  prof(faraon).  /\* asig(Asignatura) <- Asignatura es el nombre de una asignatura \*/  asig(logica).  asig(programacion).  asig(matematicas).  /\* dia(Dia) <- Dia es un día de la semana que hay alguna clase \*/  dia(lunes).  dia(miercoles).  dia(jueves).  /\*- RELACIONES -\*/  /\* r1(Profesor,Asignatura) <- Profesor imparte la Asignatura \*/  r1(maria,programacion).  r1(Profesor,Asignatura) :- prof(Profesor), Profesor\=maria,  asig(Asignatura).  /\* r2(Profesor,Dia) <- Profesor imparte sus clases el Dia de la semana \*/  r2(faraon,Dia) :- dia(Dia), Dia\=lunes.  r2(Profesor,Dia) :- prof(Profesor), Profesor\=faraon,  dia(Dia).  /\* r3(Asignatura,Dia) <- Asignatura se imparte el Dia de la semana \*/  r3(logica,Dia) :- dia(Dia), Dia\=lunes, Dia\=jueves.  r3(programacion,Dia) :- dia(Dia), Dia\=lunes.  r3(Asignatura,Dia) :- asig(Asignatura), Asignatura\=logica,  Asignatura\=programacion, dia(Dia). |

|  |
| --- |
| CONCLUSIONES |
| Fueron un poco complicados pero el final me ayudaron a entender un poco mas sobre como funciona prolog. |

|  |
| --- |
| BIBLIOGRAFÍA |
| https://e.edim.co/21312783/Ejercicios\_Prolog\_1.pdf?response-content-disposition=filename%3D%22Ejercicios\_Prolog.pdf%22%3B%20filename%2A%3DUTF-8%27%27Ejercicios\_Prolog.pdf&Expires=1511316385&Signature=xnwXVMdbBziisb1H0qTZRFoTXcatMSmCVgSnwwkTajjply~eLvqzJ9aO8m9OYrbqJ2fBPvplDxDQ2WUQTfh-WrMpb~nyvDU2PEuum67w1KFLCYQaF~y2hB~S-50ndInCE03INTkF0sYgjCqQ2v7Wc4cs9yauG6JTPxEKT3qU5qG-ZeQBd-DSweoV-qEW6tdx8Xe8BSpn~THs82aMRqDGQoCHjOhMGfdENIoOyMetI1JGcqcn4UQ1gWwJZFcNDkVDYg3UcGkBXeCNh9CmAiGu0VQdGOzTcIYPWCyEwVrrQDh-Mys1VSxedkpuI5GeAAIrR3p~oi7ZRMlwXOnxhuLkEw\_\_&Key-Pair-Id=APKAJMSU6JYPN6FG5PBQ. |