Inteligencia Artificial 2017-2018

Práctica 3: Prolog

Grupo: 2213

Celia San Gregorio Moreno

Álvaro Martínez Morales

Índice

[Ejercicio 1: predicado ‘pertenece’ 3](#_Toc511756377)

[Código 3](#_Toc511756378)

[Comentario 3](#_Toc511756379)

[Ejercicio 2: predicado ‘invierte’ 4](#_Toc511756380)

[Código 4](#_Toc511756381)

[Comentario 4](#_Toc511756382)

[Ejercicio 3: predicado ‘insert’ 5](#_Toc511756383)

[Código 5](#_Toc511756384)

[Comentario 5](#_Toc511756385)

[Ejercicio 4: conteo de elementos 6](#_Toc511756386)

[Ejercicio 4.1: predicado ‘elem\_count’ 6](#_Toc511756387)

[Código 6](#_Toc511756388)

[Comentario 6](#_Toc511756389)

[Ejercicio 4.2: predicado ‘list\_count’ 6](#_Toc511756390)

[Código 6](#_Toc511756391)

[Comentario 6](#_Toc511756392)

[Ejercicio 5: predicado ‘sort\_list’ 7](#_Toc511756393)

[Código 7](#_Toc511756394)

[Comentario 7](#_Toc511756395)

[Ejercicio 6: predicado ‘build\_tree’ 8](#_Toc511756396)

[Código 8](#_Toc511756397)

[Comentario 8](#_Toc511756398)

[Ejercicio 7: codificación de elementos mediante Árboles de Huffman 9](#_Toc511756399)

[Ejercicio 7.1: predicado ‘encode\_element’ 9](#_Toc511756400)

[Código 9](#_Toc511756401)

[Comentario 9](#_Toc511756402)

[Ejercicio 7.2: predicado ‘encode\_list’ 9](#_Toc511756403)

[Código 9](#_Toc511756404)

[Comentario 9](#_Toc511756405)

[Ejercicio 8: predicado ‘encode’ 10](#_Toc511756406)

[Código 10](#_Toc511756407)

[Comentario 10](#_Toc511756408)

# Ejercicio 1: predicado ‘pertenece’

### Código

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

% Ejercicio 1 %

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

pertenece\_m(X, [X|\_]) :- X \= [\_|\_].

pertenece\_m(X, [L|Rs]) :- pertenece\_m(X, L); pertenece\_m(X, Rs).

### Comentario

# Ejercicio 2: predicado ‘invierte’

### Código

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

% Ejercicio 2 %

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

concatena([], L, L).

concatena([X|L1], L2, [X|L3]) :- concatena(L1, L2, L3).

invierte([], []).

invierte([X|R], L) :- invierte(R, L1), concatena(L1, [X], L).

### Comentario

# Ejercicio 3: predicado ‘insert’

### Código

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

% Ejercicio 3 %

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

insert([X-P], [], [X-P]).

insert([X-P], [A-Q|Ls], R) :- P=<Q, concatena([X-P], [A-Q|Ls], R).

insert([X-P], [A-Q|Ls], [A-Q|Rs]) :- P>Q, insert([X-P], Ls, Rs).

### Comentario

# Ejercicio 4: conteo de elementos

## Ejercicio 4.1: predicado ‘elem\_count’

### Código

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

% Ejercicio 4.1 %

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

elem\_count(\_, [], 0).

elem\_count(X, [X|Ls], C1) :- elem\_count(X, Ls, C), C1 is C + 1.

elem\_count(X, [Y|Ls], C1) :- X\=Y, elem\_count(X, Ls, C1).

### Comentario

## Ejercicio 4.2: predicado ‘list\_count’

### Código

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

% Ejercicio 4.2 %

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

list\_count([], [\_|\_], []).

list\_count([X|Ls], L2, [X-C|Rs]) :- elem\_count(X, L2, C), list\_count(Ls, L2, Rs).

### Comentario

# Ejercicio 5: predicado ‘sort\_list’

### Código

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

% Ejercicio 5 %

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

sort\_list([],[]).

sort\_list([X-P|L1], L3) :- sort\_list(L1, L2), insert([X-P], L2, L3).

### Comentario

# Ejercicio 6: predicado ‘build\_tree’

### Código

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

% Ejercicio 6 %

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

build\_tree([X-\_], tree(X, nil, nil)).

build\_tree([X-P|RL], tree(1, L1, L2)) :- RL \= [], build\_tree([X-P], L1), build\_tree(RL, L2).

### Comentario

# Ejercicio 7: codificación de elementos mediante Árboles de Huffman

## Ejercicio 7.1: predicado ‘encode\_element’

### Código

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

% Ejercicio 7.1 %

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

encode\_elem(\_, [], tree(\_, \_, nil)).

encode\_elem(X, R, tree(\_, tree(V, \_, \_), \_)) :- X = V, R = [0].

encode\_elem(X, R, tree(\_, \_, tree(V, \_, \_))) :- X = V, R = [1].

encode\_elem(X, [R1|R2], tree(\_, \_, tree(V, N1, N2))) :- X \= N2, N2 \= nil, R1 = 1, encode\_elem(X, R2, tree(V, N1, N2)).

### Comentario

## Ejercicio 7.2: predicado ‘encode\_list’

### Código

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

% Ejercicio 7.2 %

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

encode\_list([], [], \_).

encode\_list([X|RL], [R1|R2], T) :- encode\_elem(X, R1, T), encode\_list(RL, R2, T).

### Comentario

# Ejercicio 8: predicado ‘encode’

### Código

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

% Ejercicio 8 %

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

dictionary(X) :- X = [a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z].

encode(L1, L2) :- dictionary(D), list\_count(D, L1, RLC), sort\_list(RLC, RSL), invierte(RSL, RI), build\_tree(RI, T), encode\_list(L1, L2, T).

### Comentario