

Ejercicio 10.1. Traducir las siguientes reglas gramaticales a reglas Prolog:

```
t --> t1, t2.
t --> t1, t, t2.
t1 --> [x].
t2 --> [y].
```

Ejercicio 10.2. Escribir una gramática que reconozca el lenguaje $a^m b^n c^m d^n$ ($m, n \geq 0$). Por ejemplo,

```
?- p([a,a,b,c,c,d], []).
Yes
?- p([a,a,b,c,d,d], []).
No
?- findall(L, (length(L,4), p(L, [])), X).
X = [[b, b, d, d], [a, b, c, d], [a, a, c, c]]
```

Ejercicio 10.3. Definir una gramática que reconozca los movimientos consistentes en listas de las palabras arriba y abajo. Por ejemplo,

```
?- movimiento_1([arriba,arriba,abajo], []).
Yes
?- movimiento_1([arriba,arriba,izquierda], []).
No
?- findall(L, (length(L,2), movimiento_1(L, [])), X).
X = [[arriba,arriba], [arriba,abajo], [abajo,arriba], [abajo,abajo]]
```

Ejercicio 10.4. Modificar la gramática anterior para que, además de reconocer los movimientos, devuelva el árbol de análisis. Por ejemplo,

```
?- movimiento_2(A, [arriba,abajo,arriba], []).
A = mov(paso(arriba), mov(paso(abajo), mov(paso(arriba))))
?- movimiento_2(A, [arriba,arriba,izquierda], []).
No
```

Ejercicio 10.5. Definir la relación `distancia(A,D)` que se verifique si D es la distancia de la posición obtenida partiendo de la posición 0 y realizando los pasos indicados en el árbol A, donde cada paso `arriba` supone un desplazamiento de +1 y cada paso `abajo` un desplazamiento de -1. Por ejemplo,

```
?- distancia(mov(paso(arriba), mov(paso(abajo), mov(paso(arriba)))), D).
D = 1
?- movimiento_2(A, [arriba,abajo,arriba], []), distancia(A,D).
A = mov(paso(arriba), mov(paso(abajo), mov(paso(arriba))))
D = 1
```

Ejercicio 10.6. Modificar la gramática de los movimientos para que, además de reconocer los movimientos, devuelva la distancia. Por ejemplo,

```
?- movimiento_3(D,[arriba,arriba,abajo],[ ]).
D = 1
?- movimiento_3(D,[arriba,arriba,izquierda],[ ]).
No
?- findall(D-L,(length(L,2),movimiento_3(D,L,[ ])),X).
X = [2-[arriba,arriba],0-[arriba,abajo],0-[abajo,arriba],-2-[abajo,abajo]]
```

Mediante una pregunta, calcular todos los movimientos con 3 pasos como máximo que terminan a distancia 1.

Ejercicio 10.7. Ampliar la gramática del ejercicio anterior de forma que se admitan dos velocidades (v_1 y v_2) de forma que los desplazamientos en la v_1 se multiplican por 1 en la v_2 por 2 e interprete la palabra **fin** como el fin del programa. Por ejemplo,

```
?- programa(D,[v1,arriba,arriba,v2,abajo,abajo,fin],[ ]).
D = -2
Yes
```

Ejercicio 10.8. Supongamos que se definen las gramáticas mediante reglas de producción con el operador `--->` como, por ejemplo, la definida en el fichero `ejemplo_gramatica.pl`

```
:- op(1200,xfx,--->).

oración      ---> sintagma_nominal, sintagma_verbal.
sintagma_nominal ---> nombre.
sintagma_nominal ---> artículo, nombre.
sintagma_verbal ---> verbo, sintagma_nominal.
artículo     ---> [el].
nombre       ---> [gato].
nombre       ---> [perro].
nombre       ---> [pescado].
nombre       ---> [carne].
verbo        ---> [come].
```

Definir la relación `deriva(+C,?LD)` que se verifica si la lista de diferencia `LD` es de la categoría gramatical `C`. Por ejemplo,

```
?- ['ejemplo_gramatica.pl'].
?- deriva(oración,[el,gato,come,pescado]-[ ]).
Yes
?- findall(_0,deriva(oración,_0-[ ]),_L), length(_L,N).
N = 64
?- deriva(sintagma_nominal,SN-[ ]).
SN = [gato] ;
SN = [perro]
Yes
```