Práctico 1: Predicados simples. Listas. Árboles binarios.

Ejercicio 1

Suponga definidos los siguientes predicados:

progenitor(X, Y)	←	X es el progenitor (madre o padre) de Y.
distintos(X, Y)	←	X e Y son diferentes.
casados(X, Y)	←	X e Y están casados.

Defina los siguientes predicados:

hermano(X, Y)	←	X e Y son hermanos.
tio(X, Y)	←	X es tío(a) de Y.
tio_politico(X, Y)	←	X es tío(a) político(a) de Y.
cuñado(X, Y)	←	X e Y son cuñados.
concuñado(X, Y)	←	X e Y son concuñados.
suegro(X, Y)	←	X es suegro(a) de Y.
consuegro(X, Y)	←	X e Y son consuegros.

En caso de ser necesarios otros predicados, defínalos.

Ejercicio 2

Sea la siguiente definición de los números naturales:

nat(0).
nat(s(X)) :- nat(X).

Defina los siguientes predicados:

suma(X, Y, S)	←	S es la suma de X e Y (S=X+Y).
resta(X, Y, R)	←	R es la diferencia entre X e Y (X=Y+R)
producto(X, Y, P)	←	P es el producto entre X e Y (P=X*Y)
distintos(X, Y)	←	X e Y son distintos ($X \neq Y$).
mayor(X, Y)	←	X es mayor que Y (X>Y).
factorial(X, Y)	←	Y es igual al factorial de X (Y=X!).
potencia(X, Y, Z)	←	Z es igual a X elevado a la Y ($Z = X^Y$).

En todos los casos X, Y y Z son números naturales. En caso de ser necesarios otros predicados auxiliares, defínalos.

Ejercicio 3

a) Defina los siguientes predicados sobre listas:

largo(L, N)	←	La lista L tiene N elementos (siendo $N=s^n(0)$).
ultimo(L,X)	←	X es el último elemento de la lista L.
sin_ultimo(L,S)	←	S es la lista que se obtiene de suprimir el último elemento de L.
reverso(L,R)	←	La lista R contiene los elementos de la lista L en orden inverso.
subsecuencia(L,S)	←	La lista S contiene elementos (no necesariamente consecutivos) de la lista L. Estos elementos preservan el orden de aparición que poseen en L.
sublista(L,S)	←	La lista S contiene elementos consecutivos de la lista L. Estos elementos preservan el orden de aparición que poseen en L.

prefijo(L, P) ← La lista P es un prefijo de la lista L.
sufijo(L, S) ← La lista S es un sufijo de la lista L.
borrar_todos(L,X,B) ← La lista B es la lista L sin ocurrencias del elemento X.
(Suponga que L es una lista de naturales definidos como en el ejercicio 2).
sin_repetidos(L,S) ← La lista S es la lista L sin elementos repetidos.
(Suponga que L es una lista de naturales definidos como en el ejercicio 2).

b) Suponga que utilizamos listas para representar conjuntos. Se deben cumplir dos propiedades para esto. En primer lugar, no pueden repetirse elementos en la lista. En segundo lugar, no importa el orden de aparición de un elemento en la lista. Teniendo en cuenta esto, defina los siguientes predicados sobre conjuntos (de naturales) representados mediante listas:

conjunto(C) ← La lista C representa un conjunto.
conj_iguales(C1, C2) ← Las listas C1 y C2 representan el mismo conjunto.
subconjunto(C, S) ← La lista S representa un subconjunto del representado por la lista C.
intersección(C1, C2, I) ← La lista I representa el conjunto intersección de los conjuntos representados por las listas C1 y C2.
union(C1, C2, U) ← La lista U representa el conjunto unión de los conjuntos representados por las listas C1 y C2.

Ejercicio 4

Defina el siguiente predicado sin utilizar predicados auxiliares, y utilizando únicamente 3 reglas o hechos:

sumaLista(Ns, N) ← N es la suma de los elementos de la lista de naturales Ns (los naturales están representados como en el Ej.2: N≡s^n(0)).

Ejercicio 5

Escriba el predicado **ancestro**(X, Y, L), que retorna una lista L con la línea de descendencia entre un ancestro X y su descendiente Y. Suponga definido el predicado *progenitor*, idéntico al de ejercicio 1 de este práctico.

Por ejemplo:

progenitor(juan, jose).
progenitor(jose, pedro).
progenitor(pedro, maria).

Si la consulta es ancestro(juan, maria, L), la lista L tiene los elementos [juan, jose, pedro,maría].

Ejercicio 6

Sea la siguiente definición de árbol binario (AB):

arbol_bin(Arbol) ← Arbol es un árbol binario
arbol_bin(vacio).
arbol_bin(arbol(Raiz, Izq, Der)) :- arbol_bin(Izq), arbol_bin(Der).

Defina los siguientes predicados sobre árboles binarios de naturales:

member_ab(X, A)	←	X es un elemento del AB A.
cambiar(X, Y, Ax, Ay)	←	El AB Ay se logra al cambiar en el AB Ax los X's por Y's.
sumar(A, S)	←	S tiene el valor de la suma de los elementos del AB A.
abb(A)	←	El AB A es un árbol binario de búsqueda (ABB).

Ejercicio 7

Un alfabeto Σ es un conjunto finito de símbolos, y ε la tira vacía. Se define el conjunto de expresiones regulares (e.r.) sobre Σ de la siguiente forma:

- 0 es una e.r. que denota al conjunto vacío, $0 \notin \Sigma$
- e es una e.r. que denota al conjunto $\{\varepsilon\}$, $e \notin \Sigma$
- a es una e.r. que denota al conjunto $\{a\}$, $\forall a \in \Sigma$
- Dadas r y s, e.r. sobre Σ que denotan a L_R y L_S respectivamente:
  i. (r|s) es una e.r. que denota a $(L_R \cup L_S)$
  ii. (r.s) es una e.r. que denota $(L_R.L_S)$
  iii. (r*) es una e.r. que denota a L_R*.
- Estas son todas las e.r. sobre Σ.

En donde la operación **.** es la concatenación, y ***** es la clausura de Kleene de lenguajes.

Defina los siguientes predicados:

sigma(S)	←	S es el alfabeto de las e.r.
exp_reg(S, R)	←	R es una e.r. sobre el alfabeto S.
pertenece(S, R, X)	←	X es una tira denotada por la expresión regular R sobre el alfabeto S.