



## Programación Declarativa

Ingeniería Informática  
Especialidad de Computación  
Cuarto curso. Primer cuatrimestre



Escuela Politécnica Superior de Córdoba  
Universidad de Córdoba

Curso académico: 2022 - 2023

---

### Práctica número 6.- Introducción al lenguaje Prolog

- **Observaciones:**

- Se deben presentar en un mismo fichero los ejercicios indicados con (\*).
- Cada predicado debe tener un comentario de cabecera como el siguiente

/\*

*factorial(N,R)*

*Predicado que comprueba si R es el factorial de N*

*Argumentos*

+ *N*:

- *Significado: número natural*

- *Tipo: entrada*

+ *R*:

- *Significado: número*

- *Tipo: entrada y salida*

*Variables locales*

+ *N1*:

- *Significado: número*

+ *R1*:

- *Significado: número*

\*/

#### 1. Amantes

- Escribe un fichero denominado “**amantes.pl**” que contenga los siguientes hechos

- *ama(juan,ana).*
- *ama(ana,miguel).*
- *ama(luis,isabel)*
- *ama(miguel,ana).*
- *ama(laura,juan).*
- *ama(isabel,luis).*

donde el predicado *ama(X, Y)* indica que *X ama a Y*.

- Escribe en **prolog** las siguientes preguntas
  - *¿A quién ama “Juan”?*
  - *¿Quién ama a “Ana”?*

- ¿Quién ama a alguien?
- ¿Quién es amado por alguien?
- ¿Quiénes se aman mutuamente?
- ¿Quién ama sin ser correspondido?
- Añade al fichero amantes.pl una regla que permita describir a los “**amantes**”, es decir, aquellas personas que se aman mutuamente.

## 2. Familia

- Escribe un fichero denominado “**familia.pl**” que contenga los siguientes hechos:
  - *hombre(antonio).*
  - *hombre(juan).*
  - *hombre(luis).*
  - *hombre(rodrigo).*
  - *hombre(ricardo).*
  - *mujer(isabel).*
  - *mujer(ana).*
  - *mujer(marta).*
  - *mujer(carmen).*
  - *mujer(laura).*
  - *mujer(alicia).*
- Define hechos en los que se afirmen los siguientes enunciados:
  - *Antonio y Ana son matrimonio*
  - *Juan y Carmen son matrimonio.*
  - *Luis e Isabel son matrimonio*
  - *Rodrigo y Laura son matrimonio.*
  - *Juan, Rodrigo y Marta son hijos de Antonio y Ana.*
  - *Carmen es hija de Luis e Isabel.*
  - *Esteban es hijo de Juan y Carmen.*
  - *Alicia es hija de Rodrigo y Laura.*
- Define una regla que indique que el predicado “*matrimonio*” es reflexivo, es decir, si X e Y forma un matrimonio entonces Y y X también lo forman.
- Define reglas para obtener:
  - *los nietos de una persona*
  - *los abuelos de una persona*
  - *los hermanos de una persona*
  - *los tíos de una persona*
  - *las tías de una persona*
  - *los primos de una persona*
  - *las primas de una persona*
  - *los suegros de una persona*

### 3. (\*) Disposición de imágenes<sup>1</sup>



- Describe con hechos la disposición de las imágenes<sup>2</sup> en la figura.
  - Usa los predicados
    - *izquierdaDe(imagen1,imagen2)*
    - *encimadeDe(imagen1,imagen2)*
- Define nuevos predicados *derechaDe* y *debajoDe* a partir de los predicados *izquierdaDe* y *encimadeDe*.

### 4. (\*) Predicados y estructuras

- Escribe los siguientes hechos que utilizan la estructura nombre y el predicado **lector**:
  - *lector(nombre("Ana", "Garrido", "Aguirre"),mujer,31)*.
  - *lector(nombre("Marta", "Cantero", "Lasa"),mujer,20)*.
  - *lector(nombre("Rodrigo", "Luna", "Soto"),hombre,30)*.
  - *lector(nombre("Marta", "Siles", "Parra"),mujer,30)*.
  - Etc.
- Escribe como comentarios de Prolog las siguientes preguntas:
  - *¿Hay lectores?*
  - *¿Quiénes son lectores?*
  - *¿Qué lectores son mujeres? y ¿hombres?*
  - *¿Hay lectores con el mismo nombre?*
- Escribe una regla para contar los lectores que edad predeterminada.
  - Nota: utiliza el predicado **bagof** y un predicado auxiliar para **contar** los elementos de una lista.

### 5. (\*) Operaciones aritméticas.

- Escribe predicados que permitan calcular las siguientes operaciones aritméticas:
  - **Suma** de los números comprendidos entre dos dados.  
*? suma(1, 3, R)*  
*R = 6*

<sup>1</sup> Ejercicio adaptado del libro de Sterling E. y Shapiro E. "The Art of Prolog". Mit Press. 1994. Página 34.

<sup>2</sup> Imágenes de uso libre tomadas de pixabay.

- **Media aritmética** de los números comprendidos entre dos dados.  
? *mediaAritmética*(1, 3, R).  
R = 2

## 6. (\*) Operaciones con listas

- Codifica el predicado **doblar\_lista**(L,R) que permita duplicar cada elemento de la lista L.
  - Por ejemplo  
?- *doblar\_lista*([],R).  
R = [].
  - ?- *doblar\_lista*([a,b,c],R).  
R = [a, a, b, b, c, c].
- Codifica el predicado **eliminaRepetidos**(L,R) que elimine todos los elementos repetidos de una lista simple.
  - Por ejemplo  
?- *eliminarRepetidos* ([a,a,b,a,c,d,c,e,e,b],R).  
R = [a,b,c,d,e].
- Codifica un predicado denominado, **invertir**, para invertir todos los elementos de una lista que puede contener **sublistas**:
  - Por ejemplo  
?- *invertir*([1,2,3,4,5],R).  
R = [5, 4, 3, 2, 1].
  - ?- *invertir*([1,[2,3],[4,5]],R).  
R = [[5, 4], [3, 2], 1].
  - Observación: codifica los siguientes predicados auxiliares
    - **es\_lista(X)**: comprueba si X es una lista
    - **concatenar(L1,L2,L)**: L es el resultado de concatenar L1 y L2.

## 7. (\*) Soluciones múltiples

- Utiliza el predicado **localidad(Nombre, Provincia, Habitantes)** para definir hechos asociados a las siguientes localidades
  - *Localidades de la provincia de Córdoba*
    - *Aguilar de la frontera: 13.500 habitantes*
    - *Espiel: 2.400 habitantes*
    - *Montoro: 9.200 habitantes*
  - *Localidades de la provincia de Sevilla*
    - *Brenes: 12.700 habitantes*
    - *Lora del río: 18.700 habitantes*
    - *Marchena: 19.400 habitantes*
- Define el predicado **contarLocalidadesProvincia(Provincia,N)** para contar las localidades de una provincia
  - Por ejemplo  
?- *contarLocalidadesProvincia* ("Sevilla",N)  
N = 3.

- Define el predicado **sumarHabitantesProvincia(Provincia,N)** para sume los habitantes de las localidades de una provincia
  - Por ejemplo
    - ?- **sumarHabitantesProvincia("Sevilla",N)**
    - $N = 3.$
- Observación:
  - Utiliza el predicado **bagof**, **setof** o **findall**.
  - Define dos predicados auxiliares para **contar** o **sumar** los elementos de una lista.

## 8. (\*) Método de ordenación Mergesort

- Codifica un **predicado**, denominado **separar**, que reciba como parámetro una lista de números y los reparta en dos listas, dependiendo de que ocupen un "lugar o posición" par o impar.
  - Ejemplos
    - $separar([]) \rightarrow [[],[]]$
    - $separar([2]) \rightarrow [[2],[]]$
    - $separar([3,2]) \rightarrow [[3],[2]]$
    - $separar([1,2,3]) \rightarrow [[1,3],[2]]$
    - $separar([4,1,2,3]) \rightarrow [[4,2],[1,3]]$
- Codifica un predicado, denominada **unir**, que reciba como parámetros dos listas ordenadas de números y devuelva otra lista con los números ordenados:
  - Ejemplos
    - $unir([],[]) \rightarrow []$
    - $unir([1],[]) \rightarrow [1]$
    - $unir([], [1]) \rightarrow [1]$
    - $unir([1],[2]) \rightarrow [2]$
    - $unir([1,3],[2]) \rightarrow [1,2,3]$
    - $unir([1,3],[2,4,5]) \rightarrow [1,2,3,4,5]$
- Codifica un predicado que permita ordenar una lista de números utilizando el método **mergesort**.
  - Ejemplo
    - ? **mergesort([5,4,1,3,2], R)**
    - $R = [1,2,3,4,5]$
  - Pasos
    - Lista original: 5 4 1 3 2
    - División
      - ✓ Primera: 5 1 2 ; 4 3 ;
      - ✓ Segunda: 5 2 ; 1 ; ; 4 ; 3 ; ;
      - ✓ Tercera: 5 ; 2 ; ; 1 ; ; 4 ; 3 ; ;
    - Fusión:
      - ✓ Primera: 2 5 ; 1 ; ; 3 4 ;
      - ✓ Segunda: 1 2 5 ; 3 4 ;
      - ✓ Tercera: 1 2 3 4 5
  - Observación
    - Utiliza los predicados auxiliares **separar** y **unir** de los ejercicios anteriores.

## 9. (\*) Donantes de sangre

- Declara los hechos relativos a una base de datos de donantes que contiene la siguiente información:
  - *donante(persona(juan,campos,ruiz),a,positivo)*.
  - *donante(persona(ana,lara,silva),ab,negativo)*.
  - *donante(persona(luis,luna,pachecho),ab,negativo)*.
  - Nota: **persona** es una estructura.
- Escribe los hechos y las reglas que permitan comprobar si una persona **puede donar** sangre a otra teniendo en cuenta el grupo sanguíneo y el factor RH.
  - 0 -: donante universal.
  - 0 +: donante universal de los grupos positivos.
  - A -: puede donar a los grupos A y AB positivos y negativos.
  - A +: puede donar a los grupos A y AB positivos.
  - B -: puede donar a los grupos B y AB positivos y negativos.
  - B +: puede donar a los grupos B y AB positivos
  - AB -: puede donar a los grupos AB positivos y negativos
  - AB +: solamente puede donar a sí mismo.
- Define reglas para el predicado **contar\_por\_grupo\_y\_factor** que permita contar todos los donantes de un grupo sanguíneo y factor rh específicos.
  - Por ejemplo:  
?- *contar\_por\_grupo\_y\_factor(ab,negativo,N)*.  
N = 2
  - Nota: utilizar el predicado **bagof** y un predicado auxiliar para **contar** los elementos de una lista.
- Escribe una regla que permita hacer las siguientes acciones consecutivas
  1. Pedir por pantalla un grupo sanguíneo y un factor rh,
  2. Pedir por pantalla el nombre de un fichero,
  3. Y escribir en dicho fichero los nombres de todos los donantes que tengan el grupo sanguíneo y el factor rh indicados.

## 10. (\*) Un árbol binario ordenado es representado por una lista de la forma: [raíz, hijo izquierdo, hijo derecho] donde raíz es un átomo e hijo izquierdo e hijo derecho son árboles binarios.

- Define predicados para:
  - Escribir los elementos del árbol en orden prefijo, sufijo e infijo.
  - Determinar la profundidad del árbol.
  - Comprobar si un elemento está en el árbol.
  - Determinar el número de nodos del árbol.
  - Determinar el número de hojas del árbol.
    - Un nodo es una hoja si sus hijos izquierdo y derecho son árboles vacíos.

- ¿Cómo se pueden redirigir las salidas de los predicados anteriores hacia un fichero de escritura?

#### 11. (\*) **Números primos**

- Define el predicado **crear\_primos(N,L)** para crear una lista compuesta por los números primos menores o iguales que el número N.
- Por ejemplo:  
 $?- \text{crear\_primos}(10,L).$   
 $L = [2,3,5,7]$
- Nota: utiliza el predicado **primo(N)** explicado en el tema 9.

#### 12. (\*) **Ficheros y números primos**

- Escribe un programa que lea los números contenidos en un fichero y que escriba los números **primos** en otro fichero.