Programación Declarativa

Ingeniería Informática
Especialidad de Computación
Cuarto curso. Primer cuatrimestre

Escuela Politécnica Superior de Córdoba
Universidad de Córdoba

Curso académico: 2019 – 2020

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.*
  - *Ricardo es hijo de Juan y Carmen.*
  - *Alicia es hija de Rodrigo e Isabel.*
- 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. (*) Declara los siguientes hechos relativos a estudiantes y directores de trabajos de fin de grado
   - Utiliza el predicado **dirige**(director,estudiante)

        - *Miguel Lara Luna dirige los trabajos de fin de grado a los siguientes estudiantes:*
            - *Juan Santos Cruz*
            - *Ana Parra Soto*
        - *Laura Prado Silva dirige los trabajos de fin de grado a los siguientes estudiantes:*
            - *Isabel Duque Campos*
            - *Pablo Alba Blanco*
    - Define reglas que permitan comprobar los siguientes hechos:
        - Si una persona dirige algún trabajo de fin de grado. Utiliza el predicado **director**(Nombre).
        - Si dos estudiantes comparten el mismo director tarea, es decir, **comparten_director**(Estudiante1,Estudiante2).

4. (*) Lectores
    - Escribe un fichero denominado "lectores.pl" que contenga 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", "Duque", "Soto"),hombre,30).*
        - Etc.
    - Escribe como comentarios de Prolog las siguientes preguntas:
        - *¿Hay lectores?*
        - *¿Quiénes son lectores?*
        - *¿Qué lectores son mujeres? y ¿hombres?*
        - *¿Hay lectores con apellidos iguales?*
    - Escribe una regla para comprobar si unos apellidos están repetidos.
        - Nota: utiliza el predicado **bagof** y un predicado auxiliar para **contar** los elementos de una lista.

5. (*) Escribe predicados que permitan calcular las siguientes operaciones aritméticas:
    - Suma de los números comprendidos entre dos dados.
      ? suma(1,3,R).
      R = 6
    - Media aritmética de los números comprendidos entre dos dados.
      ? mediaAritmetica(1,3,R).
      R = 2

6. (*) Codifica el predicado **crearImpares(N,L)** que permite crear una lista compuesta por los N primeros números impares.
    - Por ejemplo:
      ?- crearImpares(4,L)
      L = [1,3,5,7].

7. (*) 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.

8. (*) **Películas**
   - Utiliza el predicado **pelicula(Título,País,Año)** para definir hechos asociados a las siguientes películas
     - *Ben Hur, Estados Unidos, 1959*
     - *Los santos inocentes, España, 1984*
     - *Tres colores: rojo, Francia, 1994*
   - Define el predicado **contar_peliculas(País,N)** para contar las películas de un país
     - Por ejemplo
       *?- **contar_películas**("España",N).*
       N = 1.
   - Observación:
     - Utiliza el predicado ***bagof*** o ***findall***
     - Define un predicado auxiliar para **contar** para contar los elementos de una lista.

9. (*) Método de ordenación **mergesort**.
   - 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 predicados auxiliares para

- la “división” (***split***): reparte los elementos de una lista en dos listas, dependiendo de que ocupen un "lugar" par o impar
- y para la fusión (***merge***): une de forma ordenada dos listas ordenadas

10. (*) Codifica los siguientes predicados sobre **listas numéricas**.
    - Media aritmética de una lista
    - Máximo de una lista.
    - Mínimo de una lista
    - Mediana de una lista ordenada
        - Si la lista no está ordenada entonces se debe ordenar.
        - Si una lista tiene un número par de elementos, se devolverá la media aritmética de los elementos centrales.

11. (*) **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.

12. (*) 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?

13. (*) **Números primos**
    - Define el predicado **primo(N)** para comprobar si el número N es primo o no.
        - Nota: un número es primo si no tiene divisores propios menores o iguales que su raíz cuadrada.
    - 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:
          ?- crear_primos(10,L).
          L = [2,3,5,7]

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