



# **Tema 1. Introducción al PROLOG**

---

- 1. ¿Qué es el PROLOG?**
- 2. Los elementos de PROLOG**
- 3. Un programa de ejemplo**
- 4. Ejercicios propuestos**



# 1. ¿Qué es el PROLOG?

## 1.1. Un poco de historia

- PROgramming in LOGic
- Desarrollado en Europa en la década de los años 70 (Edimburgo y Marsella)
  - Aplicaciones IA
  - En EEUU LISP
- Base del proyecto 5ª Generación en Japón
  - PROLOG hubiese sido su lenguaje máquina
- Estándar ISO desde 1996: basándose en la sintaxis de Edimburgo



## 1.2. PROLOG y la programación lógica

- Paradigmas de programación: imperativa, orientación a objetos, funcional y declarativa
- Programación declarativa frente a imperativa
  - Orientada a objetivos vs orientada a datos
  - Enfoque *what-to-do* frente a *how-to-do*
  - Sólo se especifican las propiedades del problema, no como alcanzar la solución
- Programación lógica
  - Conjunto de fórmula + motor de inferencias



## 1.2. PROLOG y la programación lógica

---

- PROLOG

- Programa Prolog: conjunto de cláusulas Horn
- Ejecución: inferencias lógicas (búsqueda)
- Puede que no se encuentre la solución óptima o de la forma más eficiente
- Influye el orden de las cláusulas
- Además, tiene cláusulas ejecutables



## 1.2. PROLOG y la programación lógica

- Resolución de problemas planteados en base a objetos y las relaciones entre ellos
  - Objetos: juan, libro
- Los hechos describirán propiedades de los objetos
  - tiene(juan, libro)
- Las reglas describirán relaciones entre objetos
  - Dos personas son hermanas si ambas son mujeres y tienen los mismos padres
- Programa en PROLOG
  - Declarar hechos sobre objetos (tuplas de relaciones)
  - Definir reglas (describir relaciones)
  - Hacer consultas sobre objetos o relaciones en la Base de Conocimiento



## 1.3. SWI-PROLOG

---

- Software de libre disposición
  - <http://www.swi-prolog.org/>
- Basado en la sintaxis de Edimburgo
- Tradicionalmente interpretado, ahora compilado
- Disponible manual de referencia



## 2. Elementos de Prolog

---

1. Hechos
2. Consultas
3. Variables
4. Conjunciones
5. Reglas



## 2.1. Hechos

---

- Un hecho describe una propiedad de un objeto
  - Ejemplo: el diamante es valioso
  - Hecho:  
valioso(diamante).
- El programador debe definir la interpretación de los objetos y las relaciones entre ellos:
  - Objetos: juan, libro
  - Relación:  
tiene(juan, libro).
- El orden de la relación es importante



## 2.1. Hechos

- Ejemplo:
  - Objetos: Juan, Maria, Cocido, Vino.
  - Relaciones: A <Objeto> le gusta <Objeto>
  - Hechos:
    - A Juan le gusta el cocido
    - A Juan le gusta el vino
    - A Maria le gusta el vino
  - PROLOG:
    - le\_gusta\_a(juan, cocido).
    - le\_gusta\_a(juan, vino).
    - le\_gusta\_a(maria, vino).



## 2.1. Hechos

---

- El nombre de los objetos y relaciones empiezan por minúscula
- Primero se escribe la relación y después los objetos a modo de argumento
- Se permite “\_” para separar caracteres
- Al final debe aparecer un punto

## 2.1. Hechos

- Los predicados pueden ser:

- Monádicos

valioso(diamante).

*El diamante es valioso*

mujer(ana).

*Ana es una mujer.*

- Poliádicos

tiene(juan, libro).

*Juan tiene el libro.*

da(juan, libro, maria).

*Juan da el libro a María.*

## 2.2. Consultas

- Comienzan por:  
?-
- Equivale a preguntar: “la consulta se deduce de la base de conocimiento (hechos + reglas).”
- Ante una consulta, PROLOG intenta hacer un *matching* sobre la base de conocimiento:
  - Mismo predicado
  - Mismo número de argumentos
  - Mismos argumentos: términos / variables (instanciadas)

## 2.2. Consultas

- Las respuestas a una consulta pueden ser:
  - Yes → *Se deduce de la BC*
  - No → *No se deduce; no es lo mismo que falso*
- ¿Qué responder a una consulta?
  - [ENTER] → Termina
  - ; → ¿Hay más respuestas?

Base de conocimiento	Consultas	Respuestas
le_gusta_a(jose, maria).	?- le_gusta_a(maria,jose).	No
le_gusta_a(maria, libro).	?- le_gusta_a(maria, libro).	Yes
le_gusta_a(juan, coche).	?- le_gusta_a(juan, pescado).	No
le_gusta_a(jose, pescado).	?- le_gusta_a(jose, pescado).	Yes

## 2.2. Consultas

```
/* Los comentarios como en C */
```

```
/* Hechos: le_gusta_a(A,B) --> a A le_gusta B */
```

```
le_gusta_a(juan, maria).
```

```
le_gusta_a(pedro, coche).
```

```
le_gusta_a(maria, libro).
```

```
le_gusta_a(maria, juan).
```

```
le_gusta_a(jose, maria).
```

```
le_gusta_a(jose, coche).
```

```
le_gusta_a(jose, pescado).
```

## 2.3. Variables

- Para hacer preguntas genéricas necesitaremos variables

*¿Qué libros le gustan a María?*

*¿Qué cosas le gustan a María?*

*¿Hay algo que le guste tanto a María como a Juan?*

- Una variable permite hacer referencias a múltiples objetos en función del contexto
- Una variable puede estar libre o ligada
- En PROLOG todas las variables empiezan por Mayúsculas o “\_” (más adelante)

## 2.3. Variables

?- le\_gusta\_a(maria, X).

?- le\_gusta\_a(juan, X), le\_gusta\_a(maria, X).

- Aunque en PROLOG se suelen utilizar X, Y,... las variables pueden tener nombres mnemotécnicos:

?- le\_gusta\_a(maria, Persona).

- Ante una consulta se busca la primera instanciación (ligadura) de la variable que satisfaga el predicado (si hay alguna).
- En ese momento se añade una marca y si pedimos más respuestas ([;]) se consulta a partir de esa marca.



## 2.4. Conjunciones

- Permiten expresar múltiples objetivos que deben satisfacerse simultáneamente:

- ¿Se gustan Juan y María?

?- le\_gusta\_a(juan, maria), le\_gusta\_a(maria,juan).

?- le\_gusta\_a(jose, maria), le\_gusta\_a(maria, jose).

- Permitimos la CONJUNCIÓN LÓGICA entre (n) objetivos
- PROLOG intentará satisfacer TODOS los objetivos, en el orden dado (de izquierda a derecha).
- Tras satisfacer cada objetivo, marca el punto en la BC y las instanciaciones que ha hecho para llegar ahí.

## 2.4. Conjunciones

- Ejemplo: ¿Hay algo que le guste a Juan y a María?

?- le\_gusta\_a(juan, Algo), le\_gusta\_a(maria, Algo).

¿?

?- le\_gusta\_a(juan, Algo), le\_gusta\_a(jose, Algo).

¿?

- Las conjunciones implican instanciar y desinstanciar, para satisfacer o resatisfacer objetivos → reevaluación (con backtracking)

## 2.5. Reglas

- Establece dependencias entre hechos, de forma compacta: *afirmación general sobre objetos y sus relaciones*.
- Una regla tendrá:  
<cabeza> :- <cuerpo>  
<cuerpo> → hechos con conjunciones y variables

Ejemplo:

Juan es amigo de aquellos a quienes les gustan los coches.

En PROLOG:

es\_amigo(Juan,Y) :- le\_gusta\_a(Y, coche).

## 2.5. Reglas

- Al igual que conjunciones:  
instanciar/desinstanciar de izquierda a derecha.
- Ejemplos:  
?- es\_amigo\_de(juan, Alguien).  
  
?- es\_amigo\_de(jose, X).  
  
?- es\_amigo\_de(juan, jose).

### 3. Un primer programa de ejemplo

/\* Ejemplo1\_0. pl            → Los comentarios como en C \*/

/\* Hechos: le\_gusta\_a(A,B) --> a A le\_gusta B \*/

le\_gusta\_a(juan, maria).

le\_gusta\_a(pedro, coche).

le\_gusta\_a(maria, libro).

le\_gusta\_a(maria, juan).

le\_gusta\_a(jose, maria).

le\_gusta\_a(jose, coche).

le\_gusta\_a(jose, pescado).

/\* Reglas: es\_amigo\_de (Perso1, Perso2) --> Perso1 es amigo de  
Perso2 si a Perso2 le gustan los coches \*/

es\_amigo\_de(juan,X):- le\_gusta\_a(X, coche).

## 4. Ejercicios propuestos

- Queremos una BC sobre las personas de tu familia.

Definimos las personas (hombres y mujeres) que lo componen.

Definimos las relaciones de parentesco más sencillas:  
`es_hijo_de(X,Y)`.

Define reglas que permitan saber:

- Quién es abuelo/a de quién,
- Quién es padre/madre,
- Quién es hermana/hermano.