# Lenguajes de programación

## Paradigma de la programación LÓGICA

***Introducción al lenguaje PROLOG***

## Programación lógica

- Perteneciente al grupo de los lenguajes declarativos.
- Planteamiento de la solución a un problema basado en un **QUE** y no en un **COMO**.
- Paradigma de mayor abstracción.
- Basado en el pensamiento lógico del ser humano (Lógica como área de estudio: Aristóteles, Boole).
- Formalmente basado en el **cálculo de predicados de primer orden (CP1).**

## Características generales

- Programas no determinísticos.
- No se utilizan declaraciones de tipos de datos.
- Ambiente de traducción basado en interpretación.
- Modularización. Parámetros de entrada y/o salida pero sin distinción en su declaración.

## Características generales

- Abstracción de control basada en la recursividad.
- Abstracción de datos basada en la estructura de la lista y en describir relaciones entre objetos.
- Internamente, se generan estructuras jerárquicas de búsqueda.
- Automatización del proceso de búsqueda de soluciones -> Mayor abstracción, pero menor eficiencia.

# Cálculo de predicados (lógica de primer orden)

- Mundo constituido por **objetos**.
- Los objetos poseen **propiedades**.
- Entre los objetos existen **relaciones**.
- Algunas relaciones son **funciones**

# Ejemplos y tipos de símbolos

- **Términos**
  - Objetos en el mundo
    - carros, fábricas, Michael Jordan, …
- **Predicados**
  - Unarios: propiedades de objetos
    - color, forma, altura, …
  - n-arios: relaciones entre objetos
    - mayor que, entre, enseñar, …
- **Funciones**
  - Mapeo de objetos a objetos
    - padre-de, suma, …

# Sintaxis CP1

Oración → OraciónAtómica
| Oración Conectivo Oración
| Cuantificador Variable ... Oración
| ¬ Oración
| ( Oración )

OraciónAtómica → Predicado(Término ....)
| Término = Término

Término → Función (Término ...)
| Constante | Variable

# Sintaxis CP1

Conectivo → $\wedge$ | $\vee$ | $\Leftrightarrow$ | $\Rightarrow$

Cuantificador → $\forall$ | $\exists$

Constante → $A$ | $X_1$ | $John$

Variable → $a$ | $x$ | $s$

Predicado → Antes | TieneColor | Lloviendo

Función → Madre | PielzqDe

## Ejemplos de Oraciones

- Mujer(Maria)
- Muerde(Fido)
- Quiere(Fido,Maria)
- $\forall x$ Izquierda(x,A)
- $\forall x$ [Perro(x) $\Rightarrow$ Muerde(x)]
- $\exists x$ [Mexicano(x) $\wedge \neg$Valiente(x)]
- $\forall x$ [Persona(x) $\Rightarrow \exists y$ Amigo(x,y)]

## Lógica de primer orden...

- Un programa es un conjunto de oraciones lógicas que modelan el conocimiento de un dominio.
- La ejecución consiste en la realización de pruebas de verdad por medio de deducciones o consecuencias de las reglas, dándole así un significado al programa.

## Aplicaciones

- Inteligencia artificial
- Bases de datos relacionales
- Procesamiento de lenguaje natural
- Sistemas basados en el conocimiento
- …

## Lenguaje PROLOG

- Lenguaje de propósito general cuyo nombre viene del francés **Pro**gramation et **Log**ique
- Surge a finales de los años 70´s.
- Originario de la Universidad de Marsella en Francia, con los antecedentes de la Universidad de Edimburgo.
- Sus creadores son Alan Colmenauer y Philipe Roussel, basados en los estudios previos de Robert Kowalski.

# Intérprete de PROLOG

- Obtener la versión gratuita de *SWI Prolog* en:

  http://www.swi-prolog.org/


- Checar el manual en línea que contiene el Help.

# SWI-Prolog

- SWI-Prolog ofrece un ambiente de software gratis para programar en Prolog.
- Se ofrece junto con la herramienta gráfica XPCE, para cubrir las necesidades de aplicaciones en el mundo real.
- SWI-Prolog es ampliamente usado en investigación y educación, así como en aplicaciones comerciales.

# Usando SWI-Prolog

- Ejecutarlo por medio del shorcut o con doble-click sobre un programa .pl
- Para cargar un programa desde la consola de SWI-Prolog, usar una de las siguientes:
  - `?- ['archivo.pl'].`
  - `?- consult('archivo.pl').`
  - Se asume que el archivo se encuentra en la carpeta de trabajo.
- Ejecutar consultas (*queries*) acerca del programa.

# Algunos Comandos Útiles

- `pwd.` – despliega el nombre del archivo de trabajo
- `ls.` – lista los archivos en el directorio de trabajo
- `edit(+Spec).` – edita archivos, predicados, etc. especificados en `Spec`.
- `apropos(+Key).` – busca todos los predicados que contienen `Key` en su nombre o descripción corta
- `help(Pred).` – despliega la ayuda sobre el predicado `Pred`.
- `listing.` – Lista el programa en memoria.
- `emacs.` – Ejecuta un editor tipo emacs (*recomendado*).

# Cláusulas de Horn

- Los programas en Prolog se componen de **cláusulas de Horn** que constituyen oraciones del tipo "Si es verdad el antecedente, entonces es verdad el consecuente".
- Forma LPO:

$$a_1 \wedge a_2 \wedge a_3 \wedge .... \wedge a_n \Rightarrow b$$

  - **b** es verdad si todas las $a_i$ son verdaderas.
  - Representa una implicación.
- No obstante, en Prolog se escribe el consecuente antes que el antecedente.

Cabeza → **b** :- $a_1$ , $a_2$ , $a_3$ , .... , $a_n$ . ← Cuerpo

# Un programa en Prolog se compone de...

- Declaraciones de **HECHOS** acerca de los objetos y sus relaciones.
  - Son cláusulas de Horn sin cuerpo.
- Declaración de **REGLAS** acerca de los objetos y sus relaciones.
  - Son cláusulas de Horn con cuerpo.
- **Consultas, interrogaciones o queries**.
  - Provocan el proceso de resolución y unificación de reglas y/o hechos para instanciar variables y dar resultados.
  - Son cláusulas de Horn sin cabeza.

# Hechos

- La sintaxis de un hecho es
  $pred(arg_1, arg_2, \ldots, arg_n)$
- Donde *pred* es el nombre de una relación
- Donde los argumentos pueden ser:
  - **Constantes**
    - Numéricas (reales o enteros)
    - Strings (delimitadas con apóstrofes)
    - Atómicas (representadas por símbolos que comienzan con minúsculas)
  - **Variables**
    - Con nombre (comienzan con mayúsculas)
    - Sin nombre (guion bajo '_')
  - **Estructuras** (normalmente funciones)

# Ejemplos de Hechos

- masculino(juan).
- femenino(maria).
- papa(juan, maria).
- hija(maria, juan).
- mama(maria, pedro).
- regalo(maria, juguete, pedro).
- suma(0,1,1).

## Consultas

- Comprobación de la veracidad de un hecho, o bien, relaciones entre objetos.
- Las consultas se denominan **metas.**
- Se plantean directamente a nivel del intérprete.
- Se responden siguiendo el principio de la correspondencia de patrones (“pattern matching”).

## Ejemplos de CONSULTAS

?- papa(juan, maria). --> SI
?- suma(0, 1, 1). --> SI
?- papa(juan, pedro). --> NO
?- suma(1, 2 ,3). --> NO

**Negación como falla**: se considera falso si no se puede probar (Suposición del Mundo Cerrado)

## Consultas cuantificadas

- Uso de variables para cuantificar la consulta.
- Ejemplos:

*?- masculino(X).* *X = juan*
*?- suma(X, Y, 1).* *X = 0, Y = 1*
*?- papa(juan, X).* *X = maria*

## Uso de conjunciones

- Ejemplos:

*?- papa(juan, X), femenino(X).*
*X = maria*

*?- papa(juan, X), mama(X, Y).*
*X =maria, Y = pedro*

# Definición de REGLAS

- Útiles cuando un hecho depende del cumplimiento de otros hechos (cláusula de Horn).

*Observar que las variables son locales a cada regla*

- **Ejemplos**:

*hijo(X, Y) :- papa(Y, X), masculino(X).*
*hijo(X, Y) :- mama(Y, X), masculino(X).*
*padre(X, Y) :- papa(X, Y); mama(X, Y).*
*hijo(X, Y):- padre(Y, X), masculino(X).*

**Disyunción (OR)**

# Ejemplo Completo

**Sintaxis**
Procedimientos: **hechos + reglas**
Variables: **mayúscula**
Predicados, constantes y funciones: **minúscula**

**Cláusulas:**

```
empleado(juan, 22, e1).
empleado(pedro, 19, e2).
empleado(rosa, 22, e3).
estudiante(rosa, informatica).
estudiante(alberto, farmacia).
estudiante_trabajador(X) :-
   estudiante(X,Y),
   empleado(X,Z,W).
```

**Consultas:**

Variable anónima

```
?- empleado(N,22,_).
N = juan ;
N = rosa
?- estudiante_trabajador(X).
X = rosa
```