

# Universidad Autónoma Gabriel René Moreno

## Facultad de Ingeniería en Ciencias de la Computación y Telecomunicaciones



### **“Tarea #1. Base de Conocimiento, Familias, Simpáticos, Bebedores”**

**DOCENTE:** Ing. Edwin Vargas Yapura

**MATERIA:** Programación Lógica y Funcional

**SIGLA:** INF318

#### **INTEGRANTES (Grupo 18):**

- Alice Loreny Cambara Ortiz
- Marcelo Camacho Moreno
- Jorge Paul Jiménez Saucedo
- Miguel Eduardo Sumi Mendoza

**Santa Cruz – Bolivia**

## **CONTENIDO**

<b>1. INTRODUCCIÓN.....</b>	<b>1</b>
<b>2. ANTECEDENTES.....</b>	<b>2</b>
<b>3. BC 'FAMILIAS' .....</b>	<b>3</b>
<b>4. BC 'PERSONAS SIMPÁTICAS' .....</b>	<b>6</b>
<b>5. BC 'PERSONAS BEBEDORAS' .....</b>	<b>7</b>
<b>6. BC PROPUESTOS .....</b>	<b>9</b>
<b>7. CONCLUSIONES .....</b>	<b>11</b>
<b>8. BIBLIOGRAFIA .....</b>	<b>11</b>
<b>9. ANEXOS .....</b>	<b>12</b>

## 1. Introducción

El lenguaje de programación Prolog se originó del trabajo hecho por Robert A. Kowalski en la Universidad de Edinburgh y Alain Colmatar en la Universidad de Aix-Marseille (Francia) en los años 70. La investigación de Kowolski en el área de deducción automatizada, llevó al desarrollo con Colmerauer al uso formal de lógica como un lenguaje de programación. Kowolski proporcionó la base teórica y Colmerauer inició la programación de Prolog. Colmeraur y Phillipe Roussel desarrollaron el primer intérprete, y David Warren de la Universidad de Edinburgh desarrolló el primer compilador Prolog.

La mayoría de las implementaciones comerciales de Prolog usan la misma sintaxis desarrollada en Edinburgh. Su nombre proviene de las palabras en inglés “Programming in Logic”. Desde su creación, Prolog ha crecido en popularidad en Europa, América y Japón.

Prolog es un lenguaje de programación coloquial, lo cual quiere decir que el ordenador y el programador sostienen una especie de conversación. El espera a que introduzcas los hechos (relación entre objetos) y reglas que definen el problema a resolver. PROLOG “Programming in Logic” Originado en Europa a principios de los 70’s por Alain Colmerauer (Universidad de Marsella, Francia). Los programas lógicos pueden ser entendidos y estudiados a partir de dos conceptos: verdad y deducción lógica.

Gran parte de su éxito se debe a su conveniencia por ser código abierto (modificable) y se obtiene fácilmente en Internet, además de su capacidad de deducción de respuestas para las consultas realizadas. Prolog es un lenguaje simple y fácil de programar, hasta para principiantes, pero sus motores de inferencia no siempre son eficientes. Sus aplicaciones varían desde sistemas ambientales hasta la resolución de funciones automatizadas.

## 2. Antecedentes

La programación lógica tiene sus raíces en el cálculo de predicados, que es una teoría matemática que permite, entre otras cosas, lograr que un computador pueda realizar inferencias, capacidad que es requisito para que un computador sea una "máquina inteligente". La realización del paradigma de la programación lógica es el lenguaje prolog.

**Notaciones:** en prolog se usa las reglas gramaticales DCG (Definite Clause Grammar, en castellano gramática de cláusulas definidas), la cual implementa de forma abreviada y legible al momento de manejar información. El lenguaje Prolog puede traducir automáticamente la siguiente regla:

```
se_gustan(A, B):- gusta(A, B), gusta(B, A).  
se_gustan A con B, si a le gusta B y si a B le gusta A.
```

Nombres de objetos y relaciones siempre son con minúsculas, variables con mayúsculas y al final de un hecho de ir un punto (carácter ".").

**Hechos:** son relaciones que contienen 1 a o varios argumentos, los cuales prolog siempre los tomara como verdadero, se deberá tomar en cuenta nuestra oración (hecho) y que debe llevar el formato de predicado o predicados (sujeto).

Un conjunto de hechos en prolog son llamadas clausulas, y junto con reglas, forman lo que se conoce como una base de datos PROLOG.

```
hombre(pedro) . - dice que pedro es un hombre.
```

**Reglas:** es una sentencia condicional, con cabeza y cuerpo, declarando cosas que son ciertas dependiendo de una condición.

```
es_hombre(X) . -pregunta quienes son hombres.
```

**Variables:** son las letras para hacer preguntas en alguna regla, siempre escritas con mayúsculas dentro de las reglas.

**Operadores:** Son predicados predefinidos en PROLOG para las operaciones matemáticas básicas. Su sintaxis depende de la posición que ocupen, pudiendo ser infijos o prefijos. Por ejemplo el operador suma ("+"), podemos encontrarlo en forma prefija '+ (2,5)' o bien infija, '2 + 5'. También dispone de predicados de igualdad y desigualdad.

X = E igual	X > Y mayor
X \= Y distinto	X <= Y menor o igual
X < Y menor	X >= Y mayor o igual

### 3. Base de conocimientos 'Familias'

Son un conjunto de hechos y reglas que expresan cierto conocimiento de relaciones mediante la lógica básica, que no son otra cosa que una representación sintáctica concreta de cláusulas de Horn de primer orden, para una pequeña demostración utilizaremos la base de conocimiento del problema de las familias que nos ayudara a explicar el funcionamiento de PROLOG.

< Ejemplo 1 >Problema de las Familias:

Este se basa en un conjunto de personas con el cual estableceremos relaciones entre ellas y bajo este criterio representaremos un árbol genealógico de una familia cualquiera. (ver fig. 2)

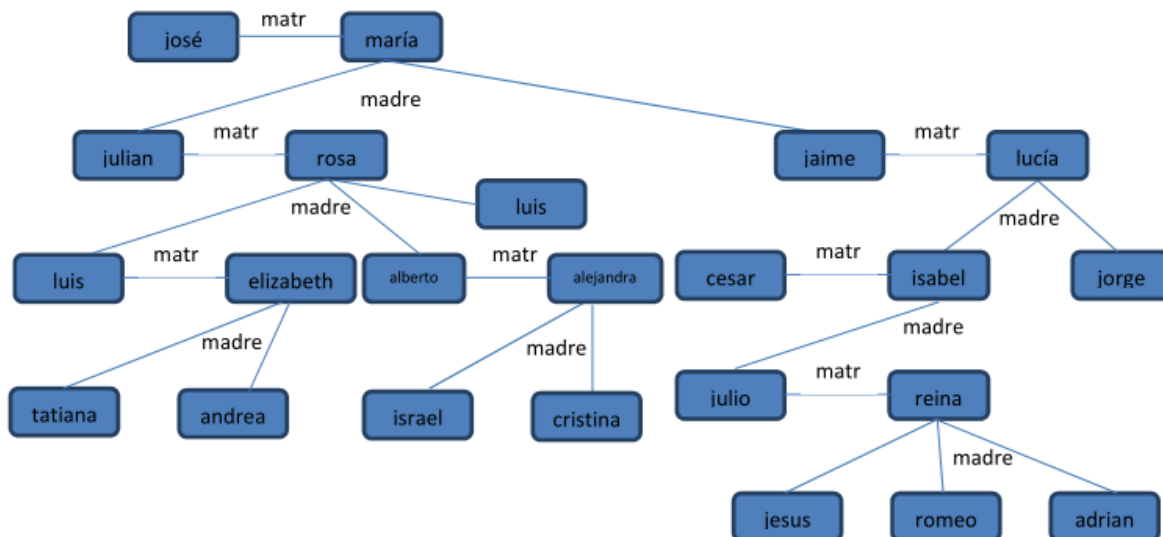




fig. 2 Árbol Genealógico, muestra la distribución familiar.

Continuando con el ejemplo anterior definimos los hechos y reglas y creamos un documento de texto en un block de notas lo guardamos este con el nombre "familia.pl" ("\*.pl" la extensión de prolog). En este comenzaremos a programar nuestra base de conocimiento.

Tenemos que tomar muy en cuenta algunas reglas:

- Las mayúsculas siempre representan variables, así que tenemos que tener mucho cuidado al escribir nuestro código en usar solo minúsculas para definir nuestros hechos y reglas.
- Al realizar consultas podemos usar una variable para obtener algún valor en particular o usar "\_" para consultas booleanas.

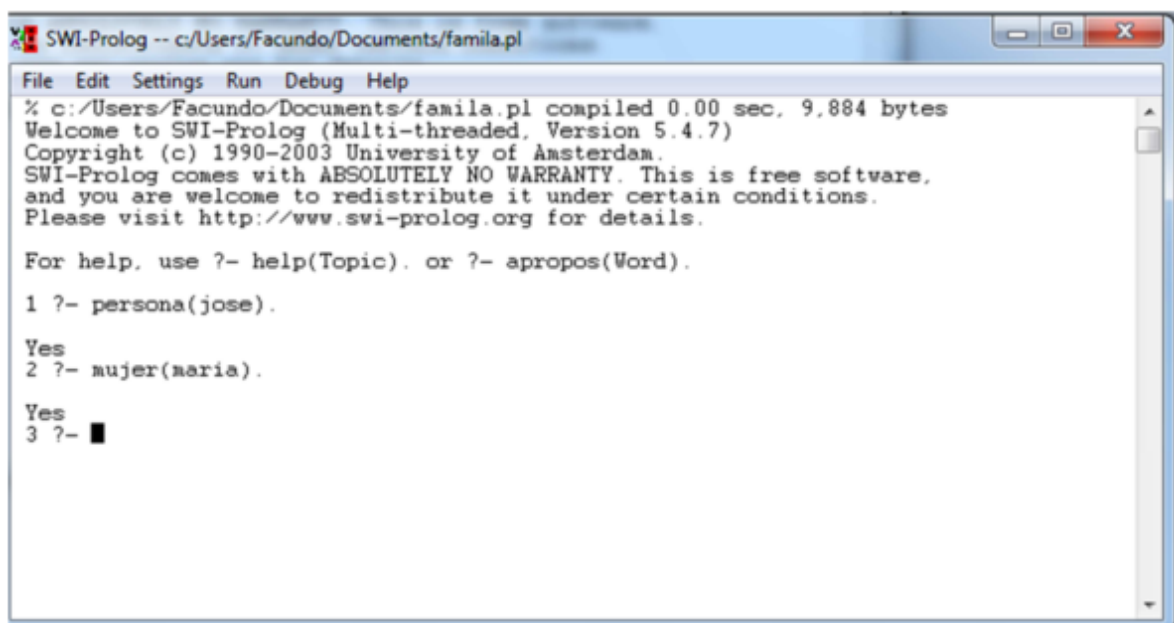
Aquí dejamos pequeños fragmentos de las líneas de código del programa. (Para el código completo vea “familia.pl” en anexo).

familia.pl		<b>Hechos</b>
hombre(jose).		
hombre(julian).		
mujer(maria).		
mujer(lucia).		
matr(jose,maria).		
matr(jaime,lucia).		
persona(X):-hombre(X);mujer(X).		<b>Reglas</b>
padre(X,Y):-madre(Z,Y),matr(X,Z).		
abuela(X,Y):-madre(Z,Y),madre(X,Z).		
hermana(X,Y):-mujer(X),madre(Z,Y),madre(Z,X).		

Nota: Para definir reglas y hechos en prolog se usan puras minúsculas ya que este reconoce las mayúsculas como variables, al terminar una línea de código se coloca un punto, punto y coma “;” es un “or” y la coma “,” es “and”.

Después de definir los hechos y las reglas pasamos a realizar las consultas: ver fig.3 a la fig.5

*Fig.3 Consultas Booleanas (devuelven un true o false).*

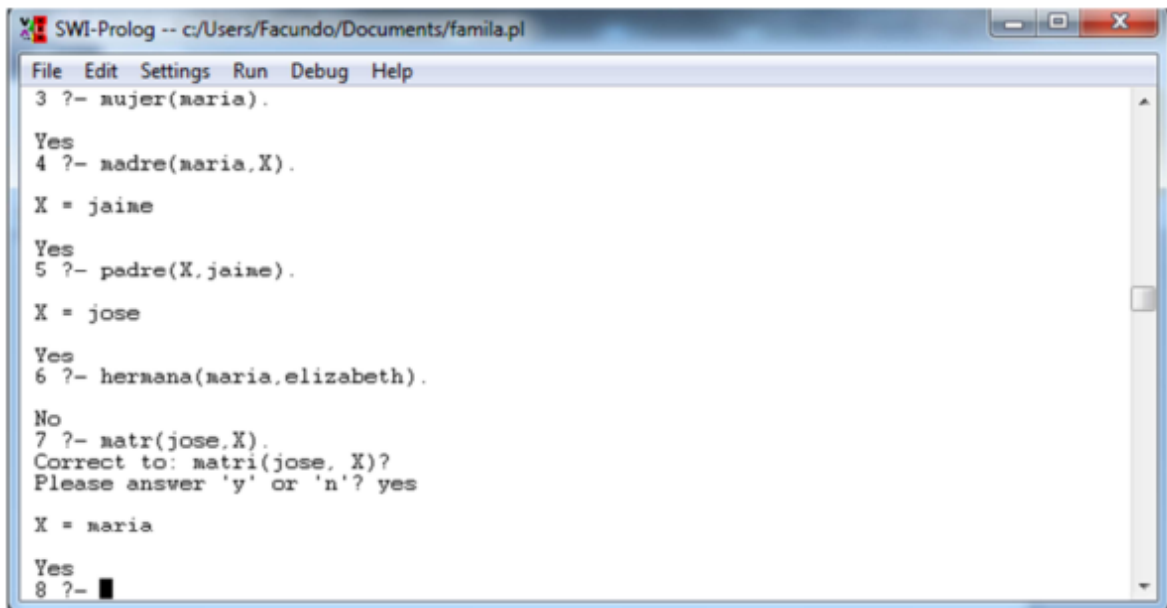


```
SWI-Prolog -- c:/Users/Facundo/Documents/familia.pl
File Edit Settings Run Debug Help
% c:/Users/Facundo/Documents/familia.pl compiled 0.00 sec, 9,884 bytes
Welcome to SWI-Prolog (Multi-threaded, Version 5.4.7)
Copyright (c) 1990-2003 University of Amsterdam.
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

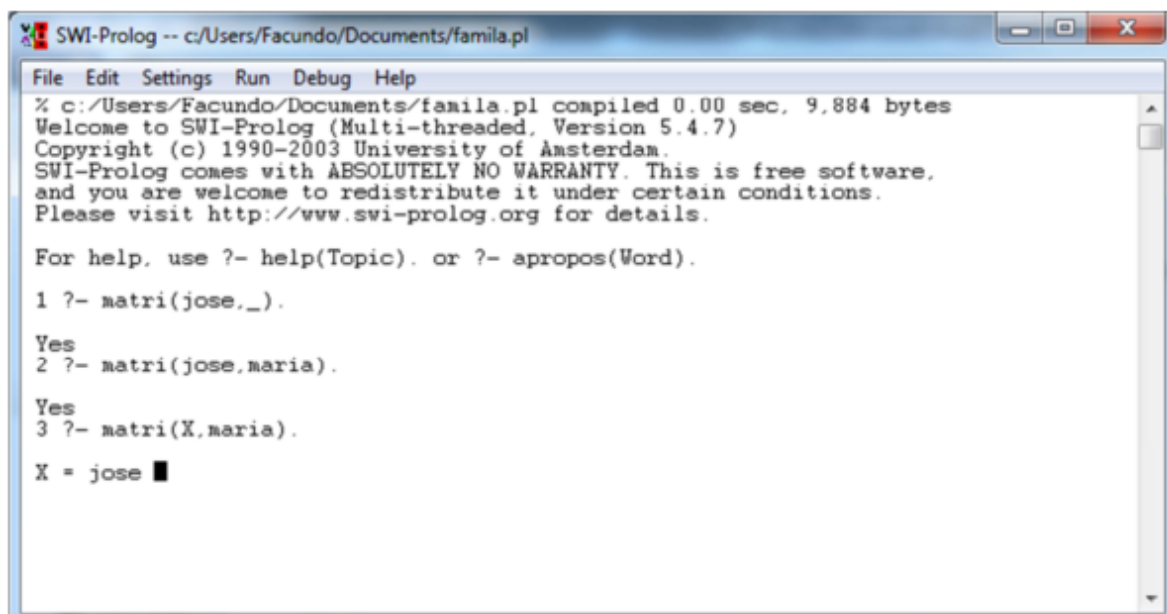
1 ?- persona(jose).
Yes
2 ?- mujer(maria).
Yes
3 ?- 
```

fig.4 Consultas que devuelven un valor cualquiera (funciones).



```
SWI-Prolog -- c:/Users/Facundo/Documents/famila.pl
File Edit Settings Run Debug Help
3 ?- mujer(maria).
Yes
4 ?- madre(maria,X).
X = jaime
Yes
5 ?- padre(X,jaime).
X = jose
Yes
6 ?- hermana(maria,elizabeth).
No
7 ?- matr(jose,X).
Correct to: matri(jose, X)?
Please answer 'y' or 'n'? yes
X = maria
Yes
8 ?- 
```

fig.5 muestra formas de consulta que se pueden realizar sea que importe un hecho, o que solo se quiera verificar si es o no cierto.



```
SWI-Prolog -- c:/Users/Facundo/Documents/famila.pl
File Edit Settings Run Debug Help
% c:/Users/Facundo/Documents/famila.pl compiled 0.00 sec, 9,884 bytes
Welcome to SWI-Prolog (Multi-threaded, Version 5.4.7)
Copyright (c) 1990-2003 University of Amsterdam.
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).
1 ?- matri(jose,_).
Yes
2 ?- matri(jose,maria).
Yes
3 ?- matri(X,maria).
X = jose
```

**Conclusión:** PROLOG es un lenguaje de programación lógico que utiliza el método de programación declarativa, a pesar de sus limitaciones es una herramienta de gran alcance y muy significativo teórico como práctico.

## 4. Base de conocimientos ‘Personas Simpáticas’

En este ejemplo queremos demostrar la afinidad de las personas hacia su pareja o con personas de su entorno (amigos).

```
hombre(jose).  
hombre(jaime).  
hombre(cesar).  
hombre(carlos).  
hombre(jorge).  
hombre(lion).  
hombre(jin).  
hombre(pedro).
```

```
mujer(maria).  
mujer(ines).  
mujer(elly).  
mujer(carla).  
mujer(laura).  
mujer(ana).  
mujer(tatiana).  
mujer(estefany).  
mujer(javiera).
```

```
matr(jose,maria).  
matr(jaime,elly).  
matr(cesar,carla).  
matr(jorge,ines).  
matr(carlos,laura).
```

```
hombreSimp(jose,maria).  
hombreSimp(jose,carla).  
hombreSimp(cesar,maria).  
hombreSimp(lion,javiera).  
hombreSimp(jin,maria).
```

```
mujerSimp(cesar,laura).  
mujerSimp(jose,elly).  
mujerSimp(jorge,ana).  
mujerSimp(jose,maria).  
mujerSimp(lion,javiera).
```

```
hombreSimpH(jin,jose).  
hombreSimpH(jose,jin).  
hombreSimpH(cesar,carlos).  
hombreSimpH(lion,jorge).
```

```
mujerSimpM(elly,ana).  
mujerSimpM(carla,javiera).  
mujerSimpM(maria,javiera).
```

```
amistad(jose,carlos).  
amistad(jorge,maria).  
amistad(lion,ines).  
amistad(jorge,carlos).
```



```

matriSimpMutuo(X,Y):-hombreSimp(X,Y),mujerSimp(X,Y),matr(X,Y).
homCAeSimpMujerCasada(X,Y):-hombre(X),hombreSimp(X,Y),matr(_,Y).
simpMutuoSolteros(X,Y):-hombreSimp(X,Y),mujerSimp(X,Y),not(matr(X,Y)).
matriNoSimpMuto(X,Y):-not(matriSimpMutuo(X,Y)).
homcasadsimpotrohombre(X,Y):-
matr(X,_),hombre(X),hombre(Y),hombreSimp(X,Y).
mujcasadsimpotramujer(X,Y):-matr(_,X),mujer(X),mujer(Y),mujerSimp(X,Y).
amigosencomun(X,Y,Z):-amistad(X,Z),amistad(Y,Z).

```

**Conclusión:** Como otros lenguajes de programación PROLOG es adecuado para unos y no aptos para otros, su programación es orientada a objetos, manipula símbolos y de gran alcance para aplicaciones de procesamiento de lenguaje natural.

## 5. Base de conocimientos ‘Personas Bebedoras’

Este ejemplo está dirigido a personas que le gusta el consumo de bebidas alcohólicas y que frecuentan a algunos bares. Además hemos agregado otros gustos (comida) y algunos restaurantes.

```

hombre(juan).
hombre(jorge).
hombre(pedro).
hombre(francisco).
hombre(jose).
hombre(luis).

mujer(luisa).
mujer(maria).

cerveza(oriental).
cerveza(paceña).
cerveza(real).

comida(charke).
comida(bife).
comida(chuleta).
comida(chancho).

local(cochabamba).
local(cholita).
local(choca).

sirve(cochabamba,charke).
sirve(cochabamba,paceña).
sirve(cochabamba,chancho).
sirve(cholita,bife).
sirve(cholita,chuleta).
sirve(choca,bife).

bar(cabaña).
bar(homero).

bebedor(francisco).
bebedor(juan).

```

```

bebedor(jorge) .
bebedor(pedro) .
bebedor(luisa) .
bebedor(maria) .

gusta(juan,paceña) .
gusta(luisa,paceña) .
gusta(luisa,real) .
gusta(maria,paceña) .
gusta(maria,real) .
gusta(jorge,real) .
gusta(pedro,real) .

sirve(cabaña,paceña) .
sirve(homero,real) .
sirve(homero,paceña) .

frec(juan,cabaña) .
frec(luisa,cabaña) .
frec(maria,cabaña) .
frec(maria,homero) .
frec(jorge,homero) .
frec(pedro,homero) .

no_frecuente(X):-hombre(X),not(frec(X,_)) .

no_gusta(X):-hombre(X),not(gusta(X,_));mujer(X),not(gusta(X,_)) .

bebedores_frec(X):-frec(X,_),gusta(X,_) .

no_sirven(X,Y):-not(sirve(X,Y)),cerveza(_) .

barnofrec(X,Y):-bebedor(X),not(frec(X,Y)) .

acuantaspersgusta2cerv(X,Y,Z):-gusta(X,Y),gusta(X,Z) .

localcerv(X,Y):-cerveza(Y),sirve(X,Y),local(X) .

localcomida(X,Y):-local(X),sirve(X,Y),comida(Y) .

listacomidas(X,Y):-local(X),sirve(X,Y) .

```

**Conclusión:** Gracias a su facilidad de programar y su sencilla sintaxis gramatical y numérica, se pueden escribir rápidamente y con pocos errores, programas claramente legibles, demás cualquier usuario puede acceder a él si lo desea y sin problemas de entendimiento. También utiliza pocos comandos en comparación con otros lenguajes de programación.

## 6. Base de conocimientos propuestos

### A) universidad.pl

Este ejemplo está dirigido a personas que son estudiantes y docentes universitarios, de diferentes carreras y universidades.

```
hombre(julian).  
hombre(jaime).  
hombre(cesar).  
hombre(jorge).  
hombre(julio).  
hombre(daniel).  
hombre(fernando).
```

```
mujer(maria).  
mujer(lucia).  
mujer(isabel).  
mujer(rosario).  
mujer(carla).  
mujer(yobana).
```

```
estudia(informatica,julio).  
estudia(redes,jorge).  
estudia(sistemas,daniel).  
estudia(derecho,maria).  
estudia(contaduria,isabel).  
estudia(arquitectura,rosario).
```

```
docente(tomas,arquitectura).  
docente(juan,informatica).  
docente(mario,redes),
```

```
estudiaEnLa(uagrm,julian).  
estudiaEnLa(nur,jaime).  
estudiaEnLa(udi,cesar).  
estudiaEnLa(utepsa,daniel).  
estudiaEnLa(ucebol,jorge).  
estudiaEnLa(nur,julio).  
estudiaEnLa(ucebol,maria).  
estudiaEnLa(uagrm,lucia).  
estudiaEnLa(nur,isabel).  
estudiaEnLa(utepsa,rosario).
```

```
hombreNoEstudia(X):-hombre(X),not(estudiaEnLa(_,X)).
```

```
estanEnLaMismaU(X,Y):-estudiaEnLaEnLa(X,Z),estuEnLaEnLa(Y,Z).
```

```
compañeros(X,Y):-estanEnLaMismaU(X,Y),estudia(X,Z),estudia(Y,Z).
```

```
alumno(X,Y):-estudia(Z,X),docente(Y,Z).
```

## **B) abogados.pl**

Este ejemplo está dirigido a personas que son clientes, abogados, que tienen distintos asuntos legales.

hombre(julian).  
hombre(jaime).  
hombre(cesar).  
hombre(marco).  
hombre(daniel).

mujer(rosario).  
mujer(carla).  
mujer(yobana).

abogado(julian).  
abogado(cesar).

abogada(yobana).

cliente(jaime).  
Cliente (rosario).  
Cliente (carla).

Asunto(homicidio).  
asunto(robo).  
asunto(secuestro).

lleva(homicidio,cesar).  
lleva(robo,julian).  
lleva(secuestro,yobana).

tieneasunto(homicidio,rosario).  
tieneasunto(robo,jaime).  
tieneasunto(secuestro,carla).  
tieneasunto(homicidio,daniel).  
tieneasunto(robo,marco).

personasmujeresQNoSeaAbogada(Y):- mujer(Y),not(abogada(Y)).  
mismoasunto(X,Y):-tieneasunto(Z,X),tieneasunto(Z,Y).

## 7. Conclusiones

- Prolog es una herramienta sencilla, potente y útil para el desarrollo de sistemas expertos e inteligencia artificial.
- Tanto la programación lógica como funcional tiene sus ventajas y desventajas. Su crecimiento seguirá mientras exista necesidad de este tipo de sistemas. Prolog tiene la ventaja que el programa revisa la base de datos para encontrar soluciones a las preguntas, infiriendo el mismo la respuesta basándose en las reglas del programa. La desventaja que esto posee, es que la resolución automática no siempre es eficiente
- Prolog en la actualidad es muy utilizado en varias universidades del mundo por ser sencillo de utilizar, ya que desde niños sin base de programación pueden utilizarlo.

## 8. Bibliografía

[http://www.ecured.cu/index.php/Prolog\\_\(Lenguaje\\_de\\_programaci%C3%B3n\)](http://www.ecured.cu/index.php/Prolog_(Lenguaje_de_programaci%C3%B3n))

<http://es.wikipedia.org/wiki/Prolog>

<http://users.dcc.uchile.cl/~abassi/IA/Prolog.html>

<http://www.dccia.ua.es/logica/prolog/docs/prolog.pdf>

<http://www.slideshare.net/maxsp5566/practicas-prolog>

<http://www.slideshare.net/ajdgeniz/aprenda-prolog-en-n-diapositivas>

<http://www.monografias.com/trabajos5/prolog/prolog.shtml>

## 9. Anexos

### PROBLEMAS DE LAS FAMILIAS

En este ejemplo queremos determinar la descendencia de una familia cualquiera formando un árbol genealógico e implementando algunas reglas que creemos convenientes para este problema.

hombre(jose).  
hombre(julian).  
hombre(jaime).  
hombre(cesar).  
hombre(jorge).  
hombre(julio).  
hombre(jesus).  
hombre(adrian).  
hombre(romeo).  
hombre(pedro).  
hombre(luis).  
hombre(israel).  
hombre(alberto).

mujer(maria).  
mujer(lucia).  
mujer(isabel).  
mujer(rosa).  
mujer(alejandra).  
mujer(cristina).  
mujer(elizabeth).  
mujer(andrea).  
mujer(tatiana).  
mujer(reina).

matri(jose,maria).  
matri(jaime,lucia).  
matri(cesar,isabel).  
matri(julian,rosa).  
matri(luis,elizabeth).  
matri(alberto, alejandra).  
matri(julio, reina).

madre(maria,jaime).  
madre(maria,julian).  
madre(lucia,isabel).  
madre(lucia,jorge).  
madre(isabel,julio).  
madre(rosa,pedro).  
madre(rosa,luis).  
madre(rosa, alejandra).  
madre(alejandra,israel).  
madre(alejandra,cristina).  
madre(elizabeth,tatiana).

madre(elizabeth, andrea).  
madre(reina,jesus).  
madre(reina, adrian).  
madre(reina,romeo).

padre(X,Y):-madre(Z,Y), matri(X,Z).  
abuela(X,Y):-madre(Z,Y), madre(X,Z);padre(Z,Y), madre(X,Z).  
abuelo(X,Y):-abuela(Z,Y),matri(X,Z).  
persona(X):-hombre(X);mujer(X).  
hermana(X,Y):-mujer(X), madre(Z,Y),madre(Z,X).  
hermano(X,Y):-hombre(X), madre(Z,Y),madre(Z,X).  
soltera(X):-mujer(X),not(matri(\_,X)).  
soltero(X):-hombre(X),not(matri(X,\_)).  
sinhijos(X,Y):-matri(X,Y),not(madre(Y,\_)).  
primoshermanos(X,Y):-madre(Z,X),madre(W,Y),hermano(Z,W);padre(Z,X),padre(W,Y),  
hermano(Z,W);madre(Z,X),padre(W,Y),hermano(Z,W);padre(Z,X),madre(W,Y),hermano(Z,W).  
nieta(X,Y):-mujer(X),madre(Z,X),madre(Y,Z);madre(Z,X),padre(Y,Z);padre(Z,X),padre(Y,Z);padre(Z,X),madre(Y,  
Z).  
nieto(X,Y):-hombre(X),madre(Z,X),madre(Y,Z);madre(Z,X),padre(Y,Z);padre(Z,X),padre(Y,Z);padre(Z,X),madre(  
Y,Z).  
suegro(X,Y):-matri(Y,Z),padre(X,Z);matri(Z,Y),padre(X,Z).  
suegra(X,Y):-matri(Y,Z),madre(X,Z);matri(Z,Y),madre(X,Z).  
cuñado(X,Y):-hombre(X),matri(X,Z),hermana(Z,Y);matri(Y,Z),hermano(X,Z).  
cuñada(X,Y):-mujer(X),matri(Z,X),hermano(Z,Y).  
tio(X,Y):-madre(Z,Y),hermano(X,Z);padre(Z,Y),hermano(X,Z).  
tia(X,Y):-madre(Z,Y),hermana(X,Z);padre(Z,Y),hermana(X,Z).  
sobrina(X,Y):-mujer(X),padre(C,X),hermano(C,Y);padre(C,X),hermano(C,Y);madre(C,X),  
hermana(C,Y);madre(C,X),hermano(Y,C).  
sobrino(X,Y):-tio(Y,X);tia(Y,X),hombre(X).