Mg. Patricio Rojas Carrasco

Universidad central de chile – región de coquimbo

PROGRAMACIÓN LÓGICA DECLARATIVA: Manual Esencial de Prolog.

Contenido

[Introducción 2](#_Toc151552317)

[Comenzando con Prolog 3](#_Toc151552318)

[Formas mínimas en Prolog 4](#_Toc151552322)

[Relaciones en Prolog 4](#_Toc151552323)

[Base de conocimiento 5](#_Toc151552324)

[Términos 5](#_Toc151552325)

[Expresiones 7](#_Toc151552326)

[Ejemplos 9](#_Toc151552327)

[Referencias 9](#_Toc151552328)

# Introducción

¡Bienvenido al fascinante mundo de la programación lógica declarativa! Este manual, titulado "Programación Lógica Declarativa: Manual Esencial de Prolog," está diseñado para guiarte a través de los intrincados caminos de Prolog, un lenguaje de programación que destaca por su enfoque único en la lógica y la inferencia.

La programación con Prolog no se parece a la típica experiencia de codificación. Aquí, no solo le indicas a la máquina qué hacer, sino que también le proporcionas hechos y reglas para que pueda razonar y llegar a conclusiones por sí misma. Este enfoque declarativo hace que Prolog sea excepcionalmente poderoso para resolver problemas complejos en inteligencia artificial, procesamiento de lenguaje natural y otras áreas.

Si deseas profundizar te sugiero visitar la siguiente página web, es oficial de swi-prolog por lo que se encuentra actualizada. Gran parte del material contenido en este manual fue obtenido desde dicha web. <https://swish.swi-prolog.org/p/Tutorial%20de%20prolog.swinb>

Prepárate para un viaje emocionante en el que aprenderás a pensar de manera lógica, a formular reglas y consultas, y a construir soluciones elegantes a problemas complejos.

**Patricio Rojas Carrasco**

Mg. Ingeniería Matemática y Computación.

Ingeniero Civil en Electrónica

# Comenzando con Prolog

No consideraré la instalación de Prolog, ya que es algo muy sencillo, ya sea en Windows o Linux. Sin embargo, si procuraré replicar las actividades en ambos sistemas operativos, para que puedas elegir donde trabajar.

Desde Linux, para ejecutar Prolog debes abrir una Terminal, escribir la palabra clave “prolog” y presionar “Enter”. Debes visualizar una imagen como la Figura 1. Se aprecia un entorno simple, minimalista, donde predomina el prompt de Prolog, indicando que todo está bien.

|  |
| --- |
| Texto  Descripción generada automáticamente |
| Figura 1. Prolog en Linux. |
| Pero, si estas trabajando en ambiente Windows, una vez instalado SWI Prolog en tu computador, podrás identificar en el escritorio el ícono correspondiente a la figura 2. Al ejecutar el programa, visualizarás el entorno de trabajo, que es diferente al de Linux, y puedes identificarlo en la figura 3.  |  |  | | --- | --- | | Logotipo  Descripción generada automáticamente | Texto, Carta  Descripción generada automáticamente | | Figura 2. Icono de Prolog | Figura 3. Prolog en Windows | |

# Formas mínimas en Prolog

Las formas más sencillas que se pueden implementar en Prolog son:

1. Los átomos de información: Una cadena de texto que empieza por letra minúscula o un valor numérico con signo opcional. Ejemplos: perro, juan, viaje…
2. Las variables: Son cadenas de texto que empiezan por una letra mayúscula o por el signo \_(unsigne). Ejemplos: X, Y, Variable…

# Relaciones en Prolog

Las relaciones, se basan en la interacción de la información contenida en la base de conocimiento. Por ejmplo:

|  |  |
| --- | --- |
|  | En esta relación se denota una existencia de parentesco entre luis y carina. En este punto Prolog no conoce quien es el padre de quien, somos nosotros quienes le damos esa interpretación, Prolog solamente se encarga de crear una relación en su base del conocimiento. |

La sintaxis de la relación tiene las siguientes características:

* Los nombres de las personas empiezan por letras minúsculas indicando que son átomos de información y no variables.
* El nombre de la relación es también un átomo de información y no debe ser una cantidad numérica.
* Número de objetos relacionados: si son cero irán entre paréntesis y si son más de uno, irán separados por comas.

Los hechos terminan siempre en punto. (.)

# Base de conocimiento

La base de conocimiento se trata de un archivo de texto (.txt), que debe estar guardado en una carpeta.

Para que Prolog sepa donde se encuentra la base de conocimiento, se debe dar la dirección completa de la ubicación.

En la Figura 4, se aprecia la forma de indicar la ubicación del archivo correspondiente a la base de conocimiento. De forma similar, en la Figura 5 se presenta el equivalente para Linux. Para ambos casos, consideramos el archivo listado.pl

|  |  |
| --- | --- |
| Texto  Descripción generada automáticamente | Texto  Descripción generada automáticamente |
| Figura 4. Consulta de Base de conocimiento en Windows. | Figura 5. Consulta de Base de conocimiento en Linux. |

# Términos

Son los elementos que conforman el lenguaje en Prolog. Existen tres tipos de términos.

**Constantes:** Átomo o Functor: Son nombres de objetos, propiedades o relaciones. Estos deben empezar en minúscula. En la Figura 6, puede ver unos ejemplos de átomos.

|  |
| --- |
|  |
| Figura 6. Ejemplos de átomos en Prolog. |

**Números:** Son valores que pueden ser enteros o decimales con signo, o sin signo. La Figura 7, ilustra algunos ejemplos.

|  |
| --- |
| Texto  Descripción generada automáticamente |
| Figura 7. Ejemplos de números en Prolog. |

**Variables:** Se representan por letras mayúsculas, números o el símbolo \_(guion bajo). Para que Prolog lo interpreta como una variable, debe comenzar con mayúscula o el símbolo \_ (guion bajo).

A continuación, en la Figura 8 se muestran algunos ejemplos de variables.

|  |
| --- |
| Texto  Descripción generada automáticamente |
| Figura 8. Ejemplos de variables en Prolog. |

En la línea 3 del ejemplo anterior, se muestra una **variable anónima,** esto implica que en cada instancia de esta variable se refiere a una variable distinta.

**Estructuras:** Son términos compuestos por otros términos, donde la sintaxis es la siguiente:

**Nombre\_atomo(termino1, termino2, …, terminoN)**

Los elementos denominados *termino*, se llaman **argumentos**. Además, al nombre del átomo se le llama **predicado**. En la Figura 9 se muestran algunos ejemplos.

|  |
| --- |
|  |
| Figura 9. Ejemplos de estructuras en Prolog. |

# Expresiones

**Operadores aritméticos:** Se pueden aplicar sobre números enteros o reales. En la tabla 1, se muestran los operadores aritméticos que se pueden utilizar.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Operador | Significado | | + | Suma (adición) | |  | Resta (substracción) | | \* | Multiplicación (producto) | | / y // | División real (cociente real) y División entera (cociente entero) | | ^ y \*\* | Potencia | | + | Signo positivo | |  | Signo negativo | |
| Tabla 1. Operadores aritméticos utilizados en Prolog |

Para realizar operaciones aritméticas, se debe utilizar el operador *is*. Para comprender su funcionamiento, en la Figura 9 se presentan algunos ejemplos. Se aprecia que este operador permite evaluar si el resultado de la operación es el esperado (True) o no lo es (False).

|  |  |
| --- | --- |
|  | Texto  Descripción generada automáticamente |
| Figura 9. Uso de operador is. Debe tener en consideración que Prolog, internamente, trabajo con números decimales. | |

Para realizar evaluaciones de operaciones aritméticas, se deben utilizar variables. La Figura 10, muestra algunos ejemplos.

|  |  |
| --- | --- |
|  |  |
| Figura 10. Ejemplos de operaciones aritméticas resueltas en Prolog. | |

Las expresiones aritméticas vistas anteriormente, son una forma simplificada de trabajo. Esto, es debido a que Prolog evalúa de la siguiente manera:

Esta estructura evita errores de digitación, que posteriormente impactarán en la evaluación. Para mayor claridad, en la Figura 11 se exponen algunos ejemplos.

|  |  |
| --- | --- |
|  |  |
| Figura 11. Se muestra otra forma de realizar las operaciones aritméticas. | |

Es importante el uso de adecuado de paréntesis, para resolver expresiones aritméticas más complejas, por ejemplo: . En la Figura 12, se aprecia el desarrollo de esta expresión.

|  |  |
| --- | --- |
|  |  |
| Figura 12. Uso de paréntesis, para resolver expresiones algebraicas. | |

# Ejemplos

Los siguientes ejemplos están desarrollados en SWI Prolog online (<https://swish.swi-prolog.org/p/Tutorial%20de%20prolog.swinb>), así como en los entornos de escritorio tanto en Windows como en Linux.

1. Se necesita establecer si existe una relación de pareja entre Vicente, Carolina, Marcelo, Juan, Luisa. Así como identificar quienes son celosos.

Primero debemos establecer la base de conocimiento, supondremos que nos han indicado las relaciones respectivas, y definiremos una función para buscar los posibles conflictos de relación (celos).

|  |  |  |
| --- | --- | --- |
|  |  | |
| Solución en SWI Prolog online. Debe tener en consideración que para ejecutar la consulta debe presionar la combinación Tab + Enter. | | |
|  | | |
|  |  | |
| Solución en SWI Prolog en Windows. Para resultados múltiples, después de la primera respuesta, debe agregar el caracter punto y como (;). | | |
|  | | |
|  | |  |
| Solución en SWI Prolog en Linux. Para resultados múltiples, después de la primera respuesta, debe agregar el caracter punto y como (;). | | |

1. Ejemplos para comprender el uso de arreglos en Prolog.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | |  | | |  |
| Uso de instrucción *member*. | | | | | | |
|  | | | | | | |
|  | | |  | | |  |
| Unificando listas, *append*. | | | | | | |
|  | | | | | | |
|  | | | | | | |
|  |  | | |  | | |
| Rotar una lista | | | | | | |
|  | | | | | | |
|  | | | | | | |
|  | |  | | |  | |
| Funciones recursivas | | | | | | |

Referencias

[1] <https://www.swi-prolog.org/>

[2] <https://swish.swi-prolog.org/p/Tutorial%20de%20prolog.swinb>

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamenteImagen que contiene Texto

Descripción generada automáticamente